



저작자표시-비영리-변경금지 2.0 대한민국

이용자는 아래의 조건을 따르는 경우에 한하여 자유롭게

- 이 저작물을 복제, 배포, 전송, 전시, 공연 및 방송할 수 있습니다.

다음과 같은 조건을 따라야 합니다:



저작자표시. 귀하는 원저작자를 표시하여야 합니다.



비영리. 귀하는 이 저작물을 영리 목적으로 이용할 수 없습니다.



변경금지. 귀하는 이 저작물을 개작, 변형 또는 가공할 수 없습니다.

- 귀하는, 이 저작물의 재이용이나 배포의 경우, 이 저작물에 적용된 이용허락조건을 명확하게 나타내어야 합니다.
- 저작권자로부터 별도의 허가를 받으면 이러한 조건들은 적용되지 않습니다.

저작권법에 따른 이용자의 권리는 위의 내용에 의하여 영향을 받지 않습니다.

이것은 [이용허락규약\(Legal Code\)](#)을 이해하기 쉽게 요약한 것입니다.

[Disclaimer](#)

**Doctor of Philosophy**

**Efficient Vision Transformers with Multi-Scale and  
Partial Attentions for Object Recognition**

**The Graduate School**

**of the University of Ulsan**

**Department of Electrical, Electronic and Computer Engineering**

**Xuan-Thuy Vo**

Efficient Vision Transformers with Multi-Scale and Partial Attentions for  
Object Recognition

Supervisor: Kang-Hyun Jo

A Dissertation

Submitted to  
the Graduate School of the University of Ulsan  
in Partial Fulfillment of the Requirements  
for the Degree of

Doctor of Philosophy

by

Xuan-Thuy Vo

Department of Electrical, Electronic and Computer Engineering

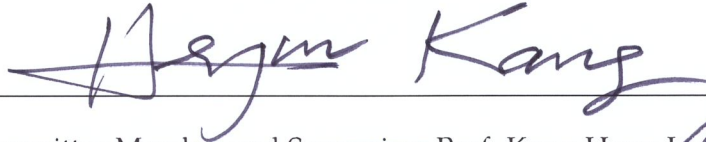
University of Ulsan

November 2023

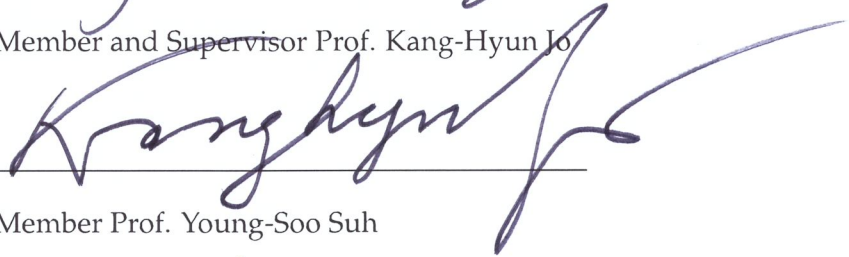
**Efficient Vision Transformers with Multi-Scale and Partial Attentions for  
Object Recognition**

This certifies that the dissertation  
of Xuan-Thuy Vo is approved:

Committee Chair Prof. Hee-Jun Kang



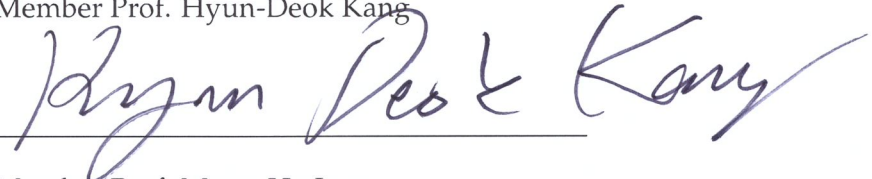
Committee Member and Supervisor Prof. Kang-Hyun Jo



Committee Member Prof. Young-Soo Suh



Committee Member Prof. Hyun-Deok Kang



Committee Member Prof. Moon-Ho Jeong



Department of Electrical, Electronic and Computer Engineering

University of Ulsan, South Korea

November 2023

*“The only way to discover the limits of the possible is to go beyond them into the impossible.”*

Arthur C. Clarke

## *Acknowledgements*

I would like to express my profound gratitude to my supervisor, Professor Kang-Hyun Jo, for affording me the invaluable opportunity to pursue my graduate studies under his expert guidance and unwavering support at the University of Ulsan. His mentorship, encouragement, and steadfast assistance have been instrumental in shaping my academic journey. Furthermore, I would like to express my sincere appreciation to the esteemed members of my thesis committee: Professor Hee-Jun Kang, Professor Young-Soo Suh, Professor Hyun-Deok Kang, and Professor Moon-Ho Jeong. Their insightful comments and constructive suggestions have played a pivotal role in enhancing the quality of this thesis.

I am grateful to all members of the Intelligent Systems Laboratory for the meaningful discussion and lesson that enhanced my knowledge and broadened my understanding of this research field, computer vision, and deep learning. Many thanks for the Brain Korea Scholarship program that allowed me to attend various conferences around the world and expand my horizons as a student and researcher. Last but not least, I reserve a special place in my heart for my parents, whose unwavering support and presence beside me throughout my studies in Korea have been a constant source of strength and motivation.

UNIVERSITY OF ULSAN

# ABSTRACT

Graduate School of the University of Ulsan

Department of Electrical, Electronic and Computer Engineering

Doctor of Philosophy

## **Efficient Vision Transformers with Multi-Scale and Partial Attentions for Object Recognition**

by Xuan-Thuy Vo

Recently, Vision Transformers have become dominant methods in processing visual data, achieving promising performances on image classification, object detection, segmentation, and multimodal foundation models. As a key component of the Transformer, self-attention has high flexibility in capturing long-range dependencies and great generalization capability. Modeling global token-to-token interactions in an input-adaptive manner defines a new paradigm in feature extraction.

With high general modeling capability and scalability to model and data size, global self-attention requires quadratic complexity with the token lengths and has weak inductive bias such as locality and relative positions between tokens. When transferring vision Transformers to downstream tasks, the model suffers a huge computational cost. Consequently, deploying the original Transformer models on real-world platforms results in high latency and energy consumption. This motivates us to develop efficient vision Transformers for object recognition that improve the efficiency of Transformer and augment inductive biases.

This research has three aims: (Aim 1) integrating self-attention layers into earlier stages of hierarchical backbone networks, (Aim 2) exchanging information across non-overlapped window self-attentions, and (Aim 3) identifying computation redundancy of sparse attention and proposing partial attention that learns spatial interactions more efficiently.

Aim 1 is presented in Chapter 3 entitled Efficient Multi-scale Spatial Interactions

(EMSNet). The EMSNet takes advantage of the hybrid network that adopts the merits of convolution and self-attention operations in hierarchical networks to achieve better visual representation. Each stage of the EMSNet efficiently models both short-range and long-range spatial interactions via the design of the multi-scale tokens. In each block, the efficient combination of the depthwise convolution, coordinate depthwise convolution, C-MHSA, and global multi-head self-attention (G-MHSA) are performed via channel splitting strategy, extracting wide ranges of frequencies and multi-order interactions.

Aim 2 is presented in Chapter 4 entitled Exchange Information across Non overlapped Local Self-Attentions via Mixing Abstract Tokens, called MAT Transformer. This method enlarges receptive fields and modeling capability of local self-attention, efficiently exchanging information across non-overlapped windows via Mixing Abstract Tokens (MAT). The intuitive idea of the MAT Transformer is to use learnable abstract tokens attached to windows. Via self-attention, each abstract token learns abstract information from each corresponding window. Hence, mixing all abstract tokens via a Transformer encoder helps to exchange information between local windows and results in global context modeling.

Aim 3 is presented in Chapter 5 entitled Efficient Vision Transformers with Partial Attention, named PartialFormer. In this chapter, we find out that there exist high similarities between attention weights and incur computation redundancy. To address this issue, this research proposes novel attention, called partial attention, that significantly reduces computation redundancy in Multi-head self-attention (MSA) and enhances the diversity of attention heads. Each query in our attention only interacts with a small set of relevant tokens.

Extensive experiments are conducted and evaluated with various tasks such as image classification, object detection, and object segmentation. As a result, our methods: EMSNet, MAT Transformer, and PartialFormer, achieve promising performances across tasks. For example, the MAT-2 achieves 79.0% Top-1 accuracy on ImageNet-1K and outperforms PVTv2-B0 by 8.5% under similar latency on a CPU device. The MAT-4 surpasses Swin-T by 1.8% mIoU with only 70% GFLOPs.



# Contents

|  |            |
|--|------------|
| <b>Acknowledgements</b>  | <b>iii</b> |
| <b>Abstract</b>  | <b>iv</b>  |
| <b>1 Introduction</b>  | <b>1</b>   |
| 1.1 Motivation and Background . . . . .                              | 1          |
| 1.2 Problem Description and Objective . . . . .                      | 3          |
| 1.3 Contributions . . . . .  | 5          |
| 1.4 Disposition . . . . .  | 7          |
| <b>2 Literature Review</b>   | <b>9</b>   |
| 2.1 Convolutional Neural Networks . . . . .                          | 9          |
| 2.2 Vision Transformer . . . . .                                     | 10         |
| 2.2.1 Global Vision Transformer . . . . .                            | 10         |
| 2.2.2 Local Vision Transformer . . . . .                             | 14         |
| 2.3 Hybrid Networks . . . . .  | 15         |
| 2.3.1 Convolution to Self-Attention Layers . . . . .                 | 15         |
| 2.3.2 Self-Attention Layers to Existing CNNs . . . . .               | 16         |
| 2.3.3 Combination of Convolution and Self-Attention Layers . . . . . | 16         |
| 2.4 Object Detection and Instance Segmentation . . . . .             | 17         |
| 2.4.1 Anchor-based detection and segmentation . . . . .              | 17         |
| 2.4.2 Query-based Detection and Segmentation . . . . .               | 20         |

|   |           |
|---|-----------|
| <b>3 Efficient Multi-scale Spatial Interactions</b>                   | <b>22</b> |
| 3.1 Introduction  | 22        |
| 3.2 Related Works   | 25        |
| 3.2.1 Convolutional Neural Networks                                   | 25        |
| 3.2.2 Vision Transformers   | 26        |
| 3.2.3 Hybrid models   | 27        |
| 3.3 Method  | 28        |
| 3.3.1 EMS Block   | 28        |
| Static operator   | 30        |
| Dynamic operator  | 31        |
| 3.3.2 C-MHSA Operation  | 32        |
| 3.3.3 EMS Variants  | 34        |
| 3.4 Experiments   | 35        |
| 3.4.1 ImageNet1k Image Classification                                 | 35        |
| 3.4.2 MS-COCO Object Detection and Segmentation                       | 36        |
| 3.5 Investigation of the EMS Block                                    | 38        |
| 3.5.1 Ablation studies  | 38        |
| 3.5.2 Comparison with local token mixers                              | 39        |
| 3.5.3 Fourier analysis  | 40        |
| 3.5.4 Interaction strength  | 41        |
| 3.6 Conclusion  | 41        |
| 3.7 Appendices  | 42        |
| 3.7.1 Analysis of Coordinate DWConv, Whiten Transformation and C-MHSA | 42        |
| Coordinate DWConv   | 42        |
| Whiten Transformation   | 42        |

|  |           |
|--|-----------|
| C-MHSA . . . . .   | 43        |
| 3.7.2 Interaction Strength . . . . .   | 44        |
| 3.7.3 Fourier analysis . . . . .   | 45        |
| 3.7.4 Additional Results . . . . .   | 45        |
| 3.7.5 Grad-CAM Visualization . . . . .   | 46        |
| <b>4 Exchanging Information across Non-overlapped Local Self-Attentions via<br/>Mixing Abstract Tokens</b> | <b>48</b> |
| 4.1 Introduction . . . . .   | 48        |
| 4.2 Related Works . . . . .  | 50        |
| 4.2.1 Vision Transformers. . . . .   | 50        |
| 4.2.2 Local self-attentions. . . . .   | 51        |
| 4.3 Methodology . . . . .  | 52        |
| 4.3.1 MAT Attention . . . . .  | 53        |
| Token Merging. . . . .   | 53        |
| Token Abstraction. . . . .   | 54        |
| Mixing Abstract Tokens. . . . .  | 54        |
| Token Fusion. . . . .  | 55        |
| 4.3.2 Bilinear PE . . . . .  | 55        |
| 4.3.3 Model Configuration . . . . .  | 56        |
| 4.4 Experiments . . . . .  | 57        |
| 4.4.1 Image Classification . . . . .   | 59        |
| Settings. . . . .  | 59        |
| Results. . . . .   | 59        |
| 4.4.2 Downstream tasks . . . . .   | 61        |
| Object detection & instance segmentation. . . . .  | 61        |

|   |           |
|---|-----------|
| Semantic Segmentation. . . . .                                | 62        |
| 4.4.3 Ablation Study . . . . .                                | 63        |
| 4.5 Conclusion . . . . .                                      | 64        |
| 4.6 Appendices . . . . .                                      | 64        |
| 4.6.1 Limitations & Future Works . . . . .                    | 64        |
| 4.6.2 Training Receipts . . . . .                             | 65        |
| Image Classification . . . . .                                | 65        |
| Object detection & Instance segmentation . . . . .            | 65        |
| Semantic Segmentation . . . . .                               | 67        |
| 4.6.3 Model Configuration . . . . .                           | 67        |
| <b>5 Efficient Vision Transformers with Partial Attention</b> | <b>69</b> |
| 5.1 Introduction . . . . .                                    | 69        |
| 5.2 Related Works . . . . .                                   | 73        |
| 5.3 Partial Vision Transformer . . . . .                      | 74        |
| 5.3.1 Overview of multi-head self-attention . . . . .         | 74        |
| 5.3.2 Partial Transformer . . . . .                           | 76        |
| Token Separation . . . . .                                    | 77        |
| Mixed Multi-Head Self-Attention . . . . .                     | 77        |
| Single-Query Attention . . . . .                              | 78        |
| The role of single query . . . . .                            | 79        |
| 5.3.3 Computational Complexity of Partial Attention . . . . . | 80        |
| 5.3.4 Model Configuration . . . . .                           | 80        |
| 5.4 Experiments and Results . . . . .                         | 81        |
| 5.4.1 ImageNet-1K Classification . . . . .                    | 81        |
| 5.4.2 MS-COCO Object Detection and Segmentation . . . . .     | 82        |

|  |           |
|--|-----------|
| 5.4.3 ADE20K Semantic Segmentation . . . . . | 86        |
| 5.5 Conclusion . . . . .                     | 87        |
| <b>6 Conclusion</b>                          | <b>88</b> |
| 6.1 Future Works . . . . .                   | 89        |
| <b>A Publications</b>                        | <b>91</b> |
| A.1 Journal . . . . .                        | 91        |
| A.2 Conference . . . . .                     | 91        |
| <b>Bibliography</b>                          | <b>95</b> |

# List of Figures

|     |   |    |
|-----|---|----|
| 1.1 | The general pipeline of object recognition. The visual features are learned via training CNNs or ViTs on large-scale datasets to capture the generalization of the input. After the model is finished, the trained weights are transferred to downstream tasks. . . . . | 4  |
| 2.1 | Feature Extraction Network Evolution. BN and LN are Batch Normalization and Layer Normalization. $c$ is the number of channels. $dwconv$ is depthwise convolution. . . . .  | 10 |
| 2.2 | Common overview of different networks in literature. . . . .  | 11 |
| 2.3 | Detailed architecture of ViT in (a) and Transformer block in (b). $N$ is the number of stacked blocks. . . . .  | 12 |
| 2.4 | (a) Attention mechanism of query-key interactions and (b) Transformer block with linear projections and detailed components of FFN. . . . .   | 13 |
| 2.5 | The main difference between convolution and self-attention. . . . .   | 13 |
| 2.6 | Various self-attention strategies. A reference point is denoted by a circle dot, attending to the shallow region. . . . .   | 15 |
| 2.7 | The common pipeline of the general object detection network. The pipeline includes two main components: Detection model and Anchor assignment & sampling. $\mathcal{L}_{cls}$ indicates classification loss. $\mathcal{L}_{loc}$ denotes localization loss. . . . .     | 17 |

- 2.8 Firstly, DETR utilizes the CNN network to perform feature extraction from the input image and produces a low spatial resolution feature map. Secondly, this feature map is flattened to the image feature vector, which is suitable for the Transformer computation (e.g., this vector is considered as a sequence of tokens). The positional encoding is added with the feature vector to serve as input of the Transformer. Thirdly, the Transformer encoder model the relationship between a token and other tokens, and outputs the global image context. Fourthly, the Transformer decoder reasons about the relations of learnable object queries and the global contextual feature. Fifthly, FFNs are feed-forward networks prediction, designed as classification and regression branches to directly generate the final prediction set of class scores and bounding box coordinates. . . . . 20
- 3.1 Network comparison between methods. **SSI**: Single-Scale Interaction, **MSI**: Multi-Scale Interaction, and **PE**: Patch Embedding. (a) ViT (Dosovitskiy et al., 2021) uses columnar style for feature extractor. (b) The recent works such as ResNet (He et al., 2016), ConvNext (Liu et al., 2022b), PVT (Wang et al., 2021a), Swin (Liu et al., 2021b) adopt the hierarchical structure and each stage performs single-scale interactions. (c) Efficient Multi-scale Spatial Interaction (EMS - Ours): At each stage, spatial interactions are performed on multiple feature scales generated based on different PEs with different patch sizes. . . . 23
- 3.2 The overall architecture of EMSNet. The network is separated into 4 stages and each stage includes 4 patch embedding layers ( $PE_{i\_1-i\_4}$ ) and  $L_i \times$  EMS Block. Each EMS block contains a Multi-scale Spatial Interaction (MSI) layer and a Channel Multi-Layer Perceptron (MLP) Interaction layer. Similar to hierarchical pyramid networks (He et al., 2016; Wang et al., 2021a; Liu et al., 2021b), the spatial dimension of the input image is progressively down-sampled across stages. . . . . 27

|      |   |    |
|------|---|----|
| 3.3  | The detailed structure of the EMS block. <b>DWConv</b> , <b>Coordinate DW-Conv</b> indicates depthwise convolution and coordinate depthwise convolution. <b>C-MHSA</b> , <b>G-MHSA</b> are Convolution-based Multi-head Self-Attention, Global Multi-head Self-Attention operations. $\delta$ is the sigmoid function and $\odot$ is element-wise matrix multiplication. $i \in \{1, 2, 3, 4\}$ indicates the index of the Patch Embedding (PE) with patch size $p$ . . . . . | 29 |
| 3.4  | The detailed architecture of the C-MHSA operation. $k$ is kernel size and $h$ is number of heads. The function $\sigma$ is softmax. . . . .   | 32 |
| 3.5  | Relative log amplitudes of the Fourier transformed feature maps of ViT-T (Dosovitskiy et al., 2021), ConvNeXt-T (Liu et al., 2022b), and our EMSNet-Tiny. $\Delta$ log amplitude measures the difference between log amplitude at scaled frequency $0.0\pi$ (center) and at $1.0\pi$ (boundary). . . . .  | 38 |
| 3.6  | Amplitude spectrum of ViT (Dosovitskiy et al., 2021), ConvNeXt (Liu et al., 2022b), and EMSNet. The ViT model tends to weaken the high-frequency component and the ConvNext model increases them. Otherwise, our model balances the range of frequencies. . . . .   | 39 |
| 3.7  | Interaction complexity of ResNet (He et al., 2016), Swin (Liu et al., 2021b), MobileNet (Sandler et al., 2018), MobileViT (Mehta and Rastegari, 2022), and EMSNet. . . . .  | 40 |
| 3.8  | Whiten transformation on the input feature $\mathbf{X}_s$ of the static branch: $\text{Conv}_{1 \times 1}(\mathbf{X}_s) - \text{Mean}(\text{Conv}_{1 \times 1}(\mathbf{X}_s))$ . The result is shown in the frequency domain. . . . .   | 43 |
| 3.9  | The attention maps of the C-MHSA operation in 5 <sup>th</sup> layer, 10 <sup>th</sup> layer, 15 <sup>th</sup> layer, and 20 <sup>th</sup> layer. . . . .  | 43 |
| 3.10 | The amplitude spectrum of the static and dynamic branches of the EMS-Tiny model. The operators in the static branch and C-MHSA tend to increase high-frequency components. G-MHSA tends to capture low-frequency components. Prop denotes Propagation operation. . . . .  | 44 |
| 3.11 | Grad-CAM activation map visualization of the EMSNet-Tiny, PoolFormerS12 (Yu et al., 2022), PVTv2-B1 (Wang et al., 2022b), and ResNet-34 (He et al., 2016). All the models are trained on ImageNet-1K. . . . .   | 47 |



|     |   |    |
|-----|---|----|
| 4.1 | Overall architecture of the MAT. Our architecture consists of a Stem Block and four stages inspired by the hierarchical network. Successive convolutions in Stem Block are used to down-sample the input image by a factor of 4 and change 3 channels to $C_1$ . In each stage, Bilinear PE (Patch Embedding) is introduced to select informative features of the patches based on input pixel locations and bilinear interpolation, and a stack of MAT Blocks is adopted. . . . .  | 51 |
| 4.2 | (a) Illustration of the MAT Block. (b) The detailed architecture of the MAT attention. The image tokens are partitioned into a set of windows and each learnable abstract token is merged with one corresponding window. Then, in each merged token, local-self attention learns spatial interactions of window tokens and the abstract token <i>versus</i> window tokens. Each abstract token learns abstract information from each window. Multi-head self-attention is used to mix learned abstract tokens, exchanging information across windows and resulting in global features. Finally, multi-head cross-attention using the image tokens as query and the mixed abstract tokens as key and value pairs propagate global information to the image features. . . . . | 52 |
| 4.3 | Bilinear PE. A grid of reference points $\mathbf{P}$ is taken from the input feature. The offsets are learned by the DWConv layer followed by $1 \times 1$ convolution, and conditioned on the input image. 9 reference points are shown in blue color as a simple example and sampled points are denoted by red points. Dash and solid lines describe the position and feature process, respectively. . . . .  | 56 |
| 4.4 | Accuracy-latency comparison between our MAT and representative networks. MAT achieves better trade-offs between Top-1 accuracy and efficiency. Latency (ms) is measured on the CPU Intel(R) Xeon(R) Gold 5220R @2.20GHz. . . . .  | 60 |
| 4.5 | Visualization of attention maps of the MAT model across blocks. In each input image, the first row shows where each abstraction token attends, and the second row illustrates the enhanced information of the image token via mixing abstract tokens globally and its aggregation. . . . .  | 61 |

|     |  |    |
|-----|--|----|
| 4.6 | Visualizations of attention maps across MAT blocks. As can be seen, different blocks capture different information of inputs. In earlier blocks, abstract tokens learn low-level features, and mixing learned abstract tokens results in larger focused regions. In later blocks, mixing abstract tokens produces high-level features, and the image tokens are enhanced. . . . .  | 68 |
| 5.1 | Performances of recent methods on ImageNet-1K. . . . .   | 70 |
| 5.2 | (a) Four attention maps in a self-attention layer are almost the same.<br>(b) Average cosine similarity between query points in the last four blocks of DeiT (Touvron et al., 2021) and our PartialFormer. . . . .   | 71 |
| 5.3 | Left: the overall architecture of PartialFormer. Right: Partial Transformer block. $L_1$ to $L_4$ are the number of stacked partial Transformer blocks. CPVT (Chu et al., 2023) is adopted as positional encoding. MLP denotes a multi-layer perceptron, consisting of two fully connected layers and GELU inserted between them. . . . .  | 74 |
| 5.4 | The structure of partial attention. Image tokens are separated into foreground and background tokens based on the evaluation of context scores. Mixed Multi-Head Self-Attention is proposed to capture informative features from the foreground set and Single-Query Attention squeezes information from background tokens to partially focus on background regions. Learnable query $\mathbf{Q}_A$ serves as a bridge between foreground and background tokens. Note that linear projections in two attentions are shared to further reduce the cost. . . . . | 75 |
| 5.5 | (a) Detailed structure of Mixed Multi-head Self-Attention and (b) Detailed structure of Single-Query Attention (SQA). . . . .  | 78 |

# List of Tables

|     |   |    |
|-----|---|----|
| 1.1 | Comparison of prevalent token mixers in deep visual models. . . . .   | 3  |
| 3.1 | Three EMS models with different width and depth settings . . . . .  | 34 |
| 3.2 | Image classification results of lightweight networks for various types<br>of models on ImageNet-1K . . . . .  | 36 |
| 3.3 | Object detection results on MS-COCO (Lin et al., 2014) and the detec-<br>tor RetinaNet (Lin et al., 2017b) and the neck FPN . . . . .   | 37 |
| 3.4 | The performance of the EMSNet on MS-COCO instance segmentation<br>with Mask R-CNN (He et al., 2017a), the neck FPN . . . . .  | 37 |
| 3.5 | Ablation study of the EMS block . . . . .   | 38 |
| 3.6 | Comparison of the local token mixers . . . . .  | 39 |
| 3.7 | Ablation study on channel splitting of the static branch . . . . .  | 45 |
| 3.8 | Latency of lightweight models . . . . .   | 46 |
| 3.9 | Latency of each operator in the EMS block . . . . .   | 46 |
| 4.1 | Comparison of local self-attentions. Our MAT is quite different from<br>recent methods in both cross-window designs and implementation.<br>$H, W$ indicates the height and width of the feature map. $[k \times H, W \times k]$<br>denotes the model computes self-attention inside cross-shaped win-<br>dows in parallel. $DWConv()$ is depthwise convolution with special<br>initial weights. . . . . | 49 |
| 4.2 | Positions of the MAT blocks. ✓ indicates the MAT blocks are used in<br>this stage and ✗ denotes there is no spatial token mixer in this stage. . .  | 57 |

|      |   |    |
|------|---|----|
| 4.3  | Detailed configurations of 5 MAT models. #dim is the number of base channels and this value is duplicated in the next stage. #blocks is the number of stacked MAT blocks. #heads is the number of heads in self-attention and cross-attention layers. . . . .   | 57 |
| 4.4  | Comparison of the MAT variants and recent methods on ImageNet-1K validation set. P denotes the number of parameters and G indicates GFLOPs. . . . .   | 58 |
| 4.5  | Object detection performances with detector SSD (Liu et al., 2016) . . . .  | 62 |
| 4.6  | Object detection performances with detector RetinaNet (Lin et al., 2017b) (1×schedule) . . . . .  | 62 |
| 4.7  | Object detection and instance segmentation performances with detector Mask R-CNN (He et al., 2017b) (1×schedule) on MS-COCO (Lin et al., 2014) validation set. All the backbones are pre-trained on ImageNet-1K (Russakovsky et al., 2015). $AP^{bb}$ and $AP^m$ indicate bounding box AP, and mask AP. . . . . | 63 |
| 4.8  | Semantic Segmentation performances with Semantic FPN (Kirillov et al., 2019) on ADE20K (Zhou et al., 2019) val set. All the models are trained for 80K iterations. . . . .  | 63 |
| 4.9  | Ablation study of the MAT attention on ImageNet-1K. Throughput is measured on one GPU Tesla V100. . . . .   | 64 |
| 4.10 | Detailed training settings for image classification MAT models. . . . .   | 66 |
| 4.11 | Detailed training settings for object detection and instance segmentation. . . . .  | 66 |
| 4.12 | Detailed model configurations of 5 MAT variants. <i>exp</i> indicates the expansion ratio in MLP. <i>h</i> denotes the number of heads in MAT Attention (MATAttn). DWconv is depth-wise convolution. . . . .  | 66 |
| 5.1  | Detailed configurations of five PartialFormers. . . . .   | 81 |

|     |  |    |
|-----|--|----|
| 5.2 | Classification performances of our PartialFormer (B0-B2) and recent methods on ImageNet-1K. All the models are trained for 300 epochs, except for CMT-Ti <sup>†</sup> and EffNet-B3 <sup>†</sup> trained for 800 epochs and 350 epochs, respectively. * denotes the downscaled versions of Swin Transformer (Liu et al., 2021b). | 83 |
| 5.3 | Comparison of PartialFormer-B3, B4 with recent methods on ImageNet-1K.   | 84 |
| 5.4 | Object detection and instance segmentation performances on MS-COCO validation set using RetinaNet (Lin et al., 2017b) and Mask R-CNN (He et al., 2017a). GFLOPs are calculated with the input size 1280×800.   | 85 |
| 5.5 | Object detection performances using SSDLite (Liu et al., 2016).  | 85 |
| 5.6 | 2D keypoint detection using SimpleBaseline (Xiao, Wu, and Wei, 2018).  | 86 |
| 5.7 | ADE20K semantic segmentation performances with Semantic FPN (Kirillov et al., 2019) and UperNet (Xiao et al., 2018). GFLOPs are measured with the input size 2048×512.   | 86 |

# List of Abbreviations

|                |  |
|----------------|--|
| <b>CNN</b>     | <b>Convolutional Neural Network</b>                      |
| <b>MLP</b>     | <b>Multi-Layer Perceptrons</b>                           |
| <b>EMSNet</b>  | <b>Efficient Multi-scale Spatial Interaction Network</b> |
| <b>MHSA</b>    | <b>Multi-Head Self-Attention</b>                         |
| <b>C-MHSA</b>  | <b>Convolution-based Multi-Head Self-Attention</b>       |
| <b>Im2Col</b>  | <b>Images to Column</b>                                  |
| <b>MAT</b>     | <b>Mixing Abstract Tokens</b>                            |
| <b>mIoU</b>    | <b>mean Intersection Over Union</b>                      |
| <b>BN</b>      | <b>Batch-Normalization</b>                               |
| <b>LN</b>      | <b>Layer Normalization</b>                               |
| <b>DETR</b>    | <b>DEtection TRansformer</b>                             |
| <b>FC</b>      | <b>Fully-Connected Layers</b>                            |
| <b>FFN</b>     | <b>Feed-Forward Networks</b>                             |
| <b>DWConv</b>  | <b>DepthWise Convolution</b>                             |
| <b>RPN</b>     | <b>Region Proposal Network</b>                           |
| <b>PE</b>      | <b>Patch Embedding</b>                                   |
| <b>SSI</b>     | <b>Single-Scale Interaction</b>                          |
| <b>MSI</b>     | <b>Multi-Scale Interaction</b>                           |
| <b>MM</b>      | <b>Matrix Multiplication</b>                             |
| <b>MS-COCO</b> | <b>MicroSoft Common Objects in COntext</b>               |
| <b>CUDA</b>    | <b>Compute Unified Device Architecture</b>               |
| <b>CPU</b>     | <b>Central Processing Unit</b>                           |
| <b>GPU</b>     | <b>Graphic Processing Unit</b>                           |
| <b>ONNX</b>    | <b>Open Neural Network Exchange</b>                      |
| <b>FLOP</b>    | <b>FLloating-point OPeration</b>                         |
| <b>FPS</b>     | <b>Frame Per Second</b>                                  |
| <b>mAP</b>     | <b>mean Average Precision</b>                            |

|                      |   |
|----------------------|---|
| <b>NMS</b>           | <b>Non-Maximum Suppression</b>                    |
| <b>FPN</b>           | <b>Feature Pyramid Network</b>                    |
| <b>SSD</b>           | <b>Single Shot Detector</b>                       |
| <b>RCNN</b>          | <b>Region-based Convolution Neural Network</b>    |
| <b>ReLU</b>          | <b>Rectified Linear Unit</b>                      |
| <b>GELU</b>          | <b>Gaussian Error Linear Unit</b>                 |
| <b>PVT</b>           | <b>Pyramid Vision Transformer</b>                 |
| <b>Swin</b>          | <b>Shifted Window self-attention</b>              |
| <b>Grad-CAM</b>      | <b>Gradient-weighted Class Activation Mapping</b> |
| <b>PartialFormer</b> | <b>Partial Vision TransFormer</b>                 |
| <b>MMSA</b>          | <b>Mixed Multi-head Self-Attention</b>            |
| <b>SQA</b>           | <b>Single-Query Attention</b>                     |

*Dedicated to  
my family*



## Chapter 1

# Introduction

### 1.1 Motivation and Background

Nowadays, Transformer models are widely used in modern networks such as state-of-the-art image classification models, object recognition, and multimodal foundation models. With strong modeling capabilities, Transformer models have outperformed Convolutional Neural Networks (CNNs), Multi-Layer Perceptron Mixers (MLPMixers), and recurrent networks by a large margin. However, in visual tasks, the main drawback of the Transformer originates from matrix multiplication between query and key matrices. It creates a lot of calculations and makes the models impractical in real-world applications. Existing methods attempt to mitigate this issue by introducing sparse attention such as spatial reduction attention and local self-attention. Although these methods are effective, the model still produces redundant information in attention patterns. Hence, diving deeper into Transformer models is the main goal of this research.

In the view of understanding involved visual data, the model compresses high dimensions of image data to lower spaces and keeps informative features through processing layer-by-layer of the model. The way the model compresses and extracts the features relies on what the image encompasses. As we interpret data, one point in the image contains two components: content (intensity values)  $c \in \mathbb{R}^3$  and geometric information  $g \in \mathbb{R}^2$ . The image is interpreted as  $I \in \mathbb{R}^{5 \times N}$ , where  $N = H \times W$  is the number of pixels in the image. With the formulation of convolution, CNNs aggregate information of local windows to the center of the local windows in a sliding manner and also capture the relative position  $g_{i-j}$  inside local windows. Generally speaking, CNN models (Krizhevsky, Sutskever, and Hinton,

2017; He et al., 2016; Liu et al., 2022b) can extract helpful features that the image contains and result in translation equivariance and locality. Otherwise, the Transformer invented by (Vaswani et al., 2017) views a sentence as a sequence of words (tokens) to compute word-to-word relationships and dynamically aggregate these features by global multi-head self-attention blocks for machine translation. With the success of Transformer in both general modeling capabilities and scalable models, Vision Transformer (ViT) (Dosovitskiy et al., 2021) tries to adapt self-attention operation in computer vision. Each image is separated into a sequence of patches (tokens), and the model learns an affinity matrix of token-to-token similarity. The ViT only considers content-to-content relationships from the input images or input features and can fail to capture positional information. The lack of geometric  $g_{i-j}$  results in weak inductive biases. The model needs a lot of data to compensate for the absence of geometric information  $g_{i-j}$ .

In terms of model complexity, the convolution operation is more efficient than the self-attention block. To extract global features, CNN-based models stack a series of convolution layers with residual connections that create a large computational cost. At the heart of Transformer, self-attention operation requires quadratic complexity with the lengths of input tokens and the model is not acceptable to adapt self-attention operation at earlier layers. Especially for downstream tasks, these networks perform predictions on the input features with high resolution. With the bottleneck computation of ViT, many methods try to reduce the cost  $O(N^2)$  to  $O(N)$  (Mehta and Rastegari, 2023), sub-sample the query, key, and value matrices (Wang et al., 2021a; Wang et al., 2022b), and compute attention in local windows (Liu et al., 2021b; Liu et al., 2022a). These methods are called sparse attention. Another line of research is to enhance the weak inductive biases of the transformer. The affinity matrix is supplemented with positional information such as absolute positional embedding (Vaswani et al., 2017), relative positional embedding (Liu et al., 2021b; Dai et al., 2021b; Min et al., 2022; Chu et al., 2021a). Other works (Mehta and Rastegari, 2022; Mehta and Rastegari, 2023; Li et al., 2022a; Li et al., 2022b) attempt to combine the strengths of convolution and self-attention operations to build hybrid networks. They inherit the strong inductive biases of CNNs and the strong modeling of ViTs, and deliver better performance than pure CNNs and ViTs.

In summary, Table 1.1 shows the difference of prevalent operations in deep visual

| Token Mixers          | Complexity<br>$\mathcal{O}()$ | #Params  | Properties                 |                            |                       |
|-----------------------|-------------------------------|----------|----------------------------|----------------------------|-----------------------|
|                       |                               |          | Input-dependent<br>weights | Global<br>receptive fields | Relative<br>Positions |
| Convolution           | $k^2HWC^2$                    | $k^2C^2$ | ✗                          | ✗                          | ✓                     |
| Depthwise Convolution | $k^2HWC$                      | $k^2C$   | ✗                          | ✗                          | ✓                     |
| Spatial MLP           | $H^2W^2C$                     | $H^2W^2$ | ✗                          | ✓                          | ✗                     |
| Self-Attention        | $HWC^2 + H^2W^2C$             | $4C^2$   | ✓                          | ✓                          | ✗                     |
| Window Self-Attention | $HWw^2C + HWC^2$              | $4C^2$   | ✓                          | ✗                          | ✓                     |

$k$ : kernel size;  $w$ : window size;  $H, W, C$ : Height, Width, and number of channels

TABLE 1.1: Comparison of prevalent token mixers in deep visual models.

models in both model costs and feature representation. Unifying three properties into one layer can extract better visual features. However, it is a straightforward way and causes high computational costs. Therefore, properly combining the best from prevalent token mixers is necessary and requires a lot of attempts to achieve this target.

Although hybrid networks attain trade-offs between accuracy and computational cost. Its mechanism only relies on single-scale spatial interactions. Hence, performing interactions on multi-scale tokens is not a trivial task. In window attention, the communication across windows is implemented by unfriendly operations and its receptive fields are limited. To tackle this issue, one part of this research is to improve the efficiency of window attentions and also enlarge receptive fields. In sparse attention methods, each query attends to all spatial locations/irrelevant regions. This strategy produces a lot of calculations and impairs performance due to lost information in downed key and value features. Therefore, forcing the tokens to attend to important regions is one part of this research.

## 1.2 Problem Description and Objective

Vision Transformer (Dosovitskiy et al., 2021) embeds the input tokens into a sequence of patches and each patch dimension has  $16 \times 16$  pixels. ViT captures patch-to-patch interactions via global self-attention layers. Although ViT without inductive biases can learn long-range dependencies from the input token and result in strong modeling capabilities, self-attention layers have quadratic complexity with token lengths. When transferring trained ViT to object recognition, the model suffers a huge of computation cost. This motivates us to design efficient vision Transformers for object recognition by reducing the computation redundancy in attention patterns

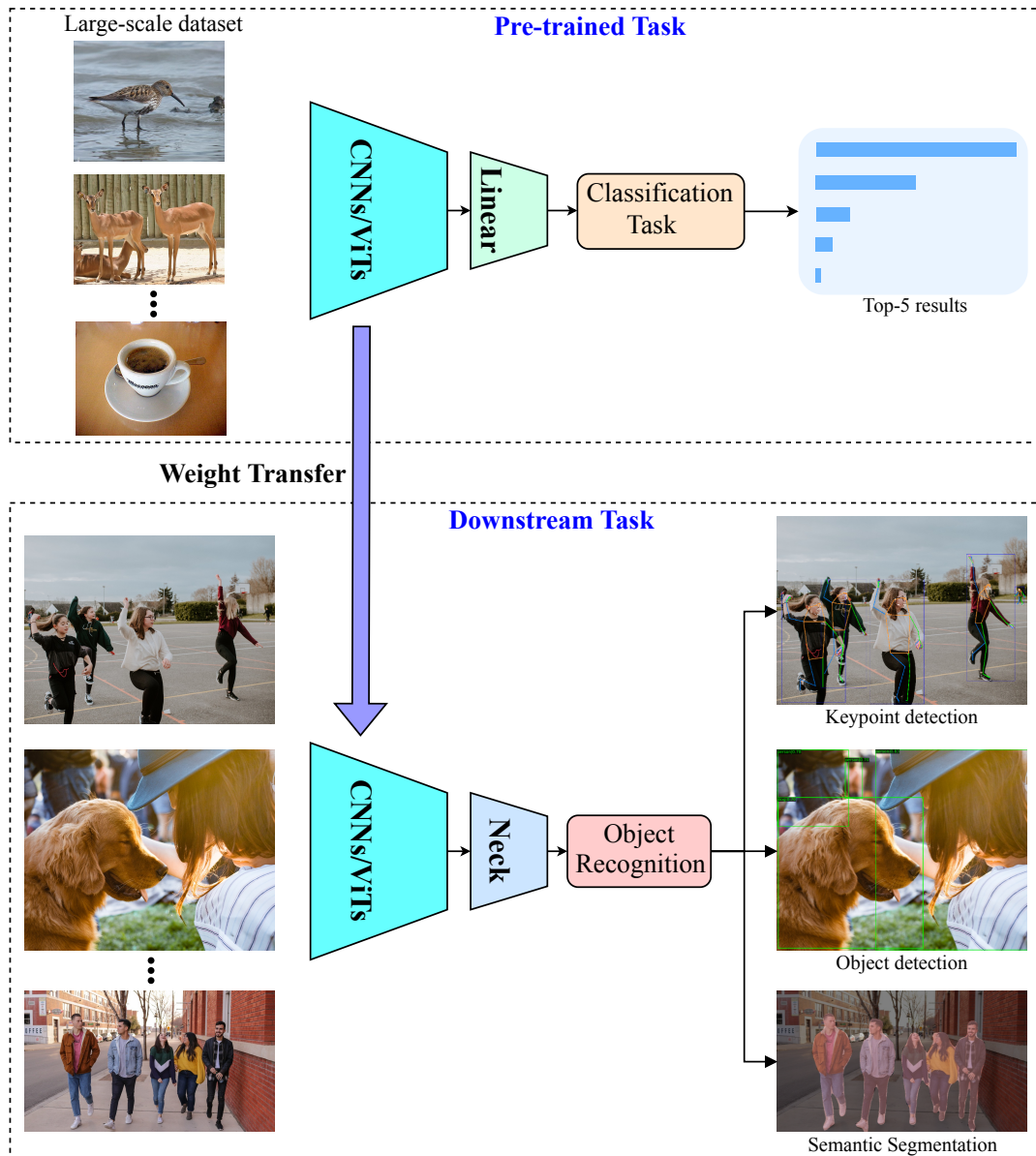


FIGURE 1.1: The general pipeline of object recognition. The visual features are learned via training CNNs or ViTs on large-scale datasets to capture the generalization of the input. After the model is finished, the trained weights are transferred to downstream tasks.

and augmenting the diversity of features. The general pipeline of object recognition is shown in Figure 1.1. General features are learned on large-scale datasets via training CNNs/ViTs models. Then, learned weights are transferred to downstream tasks (object detection, semantic segmentation, and keypoint detection) to evaluate the quality of pre-trained weights and also deploy for real-work applications.

Firstly, the efficient combination of self-attention and convolution is conducted in this research. Current methods (Pan et al., 2022c; Pan, Cai, and Zhuang, 2022; Li et al., 2022a; Lin et al., 2023) employ convolution in earlier stages and self-attention

in later stages to decrease model costs. The self-attention only learns global features on a single scale (Dosovitskiy et al., 2021; Touvron et al., 2021; Liu et al., 2021b; Dong et al., 2022; Zhang et al., 2023), and hence, it limits network capacities. Therefore, this research enhances modeling capability by performing spatial token interactions on multi-scale features while still preserving the efficiency of the networks.

Secondly, the improvement of local self-attention is described in this research. Although local self-attentions (Vaswani et al., 2021; Liu et al., 2021b; Dong et al., 2022; Pan et al., 2023) have translation-equivariance and locality similar to convolution, their receptive fields are limited, and leading to weak modeling capability. The main reason is that self-attention is computed within non-overlapped windows. To overcome this issue, common methods need further operations to communicate the information across windows such as window shifting (Liu et al., 2021b), and sliding (Vaswani et al., 2021). These operations are memory unfriendly, not well supported, and optimized by modern deep-learning frameworks. Alternatively, this research exchanges information across non-overlapped windows via efficiently mixing abstract tokens. Our design is efficient and easy to implement, only containing matrix multiplications.

Thirdly, existing sparse attention methods (Wang et al., 2021a; Wang et al., 2022b; Liu et al., 2021b; Dong et al., 2022; Tu et al., 2022; Zhang et al., 2023) treat tokens equally and heads independently. Attendance of each query to all spatial locations/irrelevant regions produces a lot of calculations and lost information. The work of this research aims to improve the attention areas by forcing the model to learn relevant features, and also significantly cut down a lot of computations.

### 1.3 Contributions

The work in this research explores various approaches to the development of efficient vision Transformers for image classification and downstream tasks. The first study addresses the integration of self-attention layers in the earlier stages of the backbone. The main contributions of this study are summarized as follows:

- At each stage of EMSNet, token mixers are performed on multi-scale features with progressive shrinking resolutions. Local and global mixers are adopted

for features with high and low resolutions. This design brings two benefits: (1) extracting both short-range and long-range interactions in one layer and hence, enhancing the general modeling capacities of the model; (2) easily performing spatial interactions of global MHSA on the coarse features at earlier stages and hence, avoiding the large cost of the MHSA in lower layers. Each stage of the EMSNet has a pyramid structure and can benefit downstream tasks that require multi-scale interactions.

- The EMS block combines the strengths of the proposed coordinate convolution, local self-attention, and global self-attention via a simple channel-splitting strategy and patch embeddings with various patch sizes. Convolution and local self-attention are performed on the features with small patch sizes to capture high-frequency components. Global self-attention is used for the feature with a large patch size to extract low-frequency components. Therefore, the EMS block can learn a wide range of frequencies from the input. In another aspect, leveraging the sparseness of the input tokens into feature learning can enhance interaction complexity. Existing works capture low- and high-order interactions while the EMSNet forces the model to focus on multi-order interactions.
- For local self-attention, C-MHSA is proposed to compute self-attention inside the windows. The C-MHSA unifies the insightful properties of convolution (relative position) and local self-attention (input-dependent attention weights) into one operation. In our implementation, the input feature is partitioned into overlapped windows via Im2Col operation, while Swin Transformer has non-overlapped windows and needs cyclic-shifted window operation to communicate information across windows. Previous local self-attentions only have input-dependent attention weights, while C-MHSA has both input-dependent attention weights and learnable kernels. Hence, our design has stronger generalization capacities.

The second study tackles limited receptive fields and weak modeling capability of window attention. The contributions of this study are described as follows:

- The Mixing Abstract Tokens (MAT) block is proposed to perform cross-window attention through friendly implementation.

- A bilinear patch embedding between two stages is proposed to downsample the number of image tokens in a data-dependent manner and increase the number of channels.
- As a result, this method achieves promising performances across tasks. For example, the MAT-2 achieves 79.0% Top-1 accuracy on ImageNet-1K and outperforms PVTv2-B0 by 8.5% under similar latency on a CPU device. The MAT-4 surpasses Swin-T by 1.8% mIoU with only 70% GFLOPs.

The third study investigates the computation redundancy of self-attention and proposes a solution to reduce a lot of costs while keeping high performances of vision Transformers. The main contributions of this work are defined as follows:

- Investigations of computation redundancy and feature diversity of spatial reduction attention and local self-attention.
- This work proposes intuitive attention, called partial attention, that significantly reduces computation redundancy in multi-head self-attention and enhances the diversity of attention heads. In our attention, each relevant query only attends to a small set of relevant keys and values.
- Efficient single-query attention is introduced to force one unique query to attend to the background set. Single-query attention only results in linear complexity with background token length. Therefore, a lot of computational costs are cut down while still keeping the global context modeling of the Transformer.

## 1.4 Disposition

The organization of this thesis is described as follows:

Chapter 2 reviews existing methods related to convolutional neural networks, vision Transformers, and hybrid networks. We also discuss the applications of convolution neural networks, vision Transformers, and hybrid networks such as object detection, instance segmentation, and semantic segmentation.

Chapter 3 describes an efficient combination of convolution and self-attention via a design of multi-scale tokens at each stage. It includes an overview of the architecture, efficient multi-scale spatial interaction block, convolution-based multi-head self-attention, network configurations, and experimental results. The last part analyzes the behavioral learning of the proposed method in terms of the Fourier spectrum, and interaction strengths of multi-scale sparse token mixers.

Chapter 4 discusses an improvement of local self-attention via mixing abstract tokens learned from local windows. This chapter presents the drawbacks of window attention, its solution for enlarging receptive fields, the proposed method, and model configuration. Furthermore, extensive experiments verify the effectiveness of the proposed method.

Chapter 5 investigates the computation redundancy of global and local self-attention operations, and explains the proposed partial attention for achieving efficient vision Transformers. This chapter contains parts related to the proposed attention, including an overview of self-attention, network architecture, partial attention, and model configurations. The last of this chapter shows experimental results across object recognition on benchmark datasets.

Finally, Chapter 6 concludes this research and presents possible directions for future works.



## Chapter 2

# Literature Review

This chapter discusses the development of efficient networks from classical Convolutional Neural Networks (CNNs) to modern hybrid networks and their application to dense prediction tasks.

In the following, the first part covers existing CNN methods; the second part describes the vision Transformer and its variant; section 3 analyzes the combination of convolution and self-attention; and the final section discusses the application of backbone when transferring trained models to other tasks.

### 2.1 Convolutional Neural Networks

In 2012s, AlexNet (Krizhevsky, Sutskever, and Hinton, 2017) successfully trained the CNN model on large-scale ImageNet dataset (Russakovsky et al., 2015) and established a new record in the ranking of classification research. From this milestone, many works were proposed with more innovative designs. VGG (Simonyan and Zisserman, 2014) stacks a sequence of plain  $3 \times 3$  convolution layers to 16 or 19 layers, shown in Figure 2.1(a). ResNet (He et al., 2016) investigates the vanishing gradient problem when stacking plain layers with more than 20 layers. From the observation, ResNet proposed a residual connection that can help network optimization and extend the network deeper. With the success of these designs, ResNet has become the dominant method for vision tasks. Other works improve the ResNet model with dense residual connection (Huang et al., 2017), Inception module (Szegedy et al., 2015), deformable convolution (Zhu et al., 2019), depthwise separable convolution (Howard et al., 2017; Sandler et al., 2018), and integration with attention (Hu, Shen,

## Feature Extraction Network Evolution

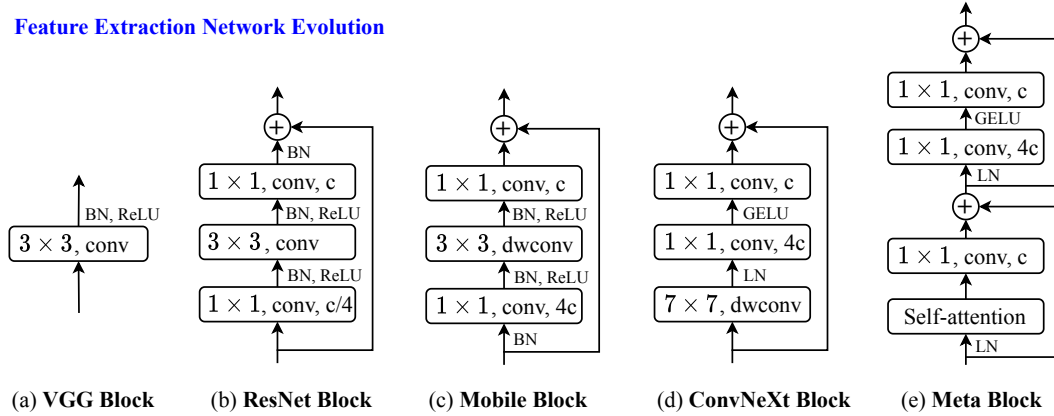


FIGURE 2.1: Feature Extraction Network Evolution. BN and LN are Batch Normalization and Layer Normalization.  $c$  is the number of channels. dwconv is depthwise convolution.

and Sun, 2018; Cao et al., 2019). Figure 2.1 illustrates the development of feature extract blocks in the literature. A pair of Batch Normalization and ReLU is compatible with convolutional blocks while Layer Normalization and GELU perform well on self-attention blocks. ConNeXt (Liu et al., 2022b) simplifies the design of the self-attention block and uses  $7 \times 7$  depthwise convolution as a token mixer. With this replacement, ConvNeXt achieves on-par performance with Swin Transformer (Liu et al., 2021b) and opens a new way in the investigation of large-kernel convolution.

The common point of the CNN-based models is that the network extracts the features in a hierarchical manner, shown in Figure 2.2(b). Earlier layers capture low-level features with high-frequency components and the latter layers extract high-level features with low-frequency patterns. Although vanilla convolution has strong inductive biases and efficient implementation, their receptive fields are limited to kernel sizes and the model requires a lot of layers to promote network ability.

## 2.2 Vision Transformer

### 2.2.1 Global Vision Transformer

After the success of the original Transformer (Vaswani et al., 2017) in the language field, the significant shift of the Transformer towards incorporating the vision, audio, and foundation models has achieved noteworthy attention. DETR (Carion et al., 2020) was the first method that successfully applies the Transformer encoder,

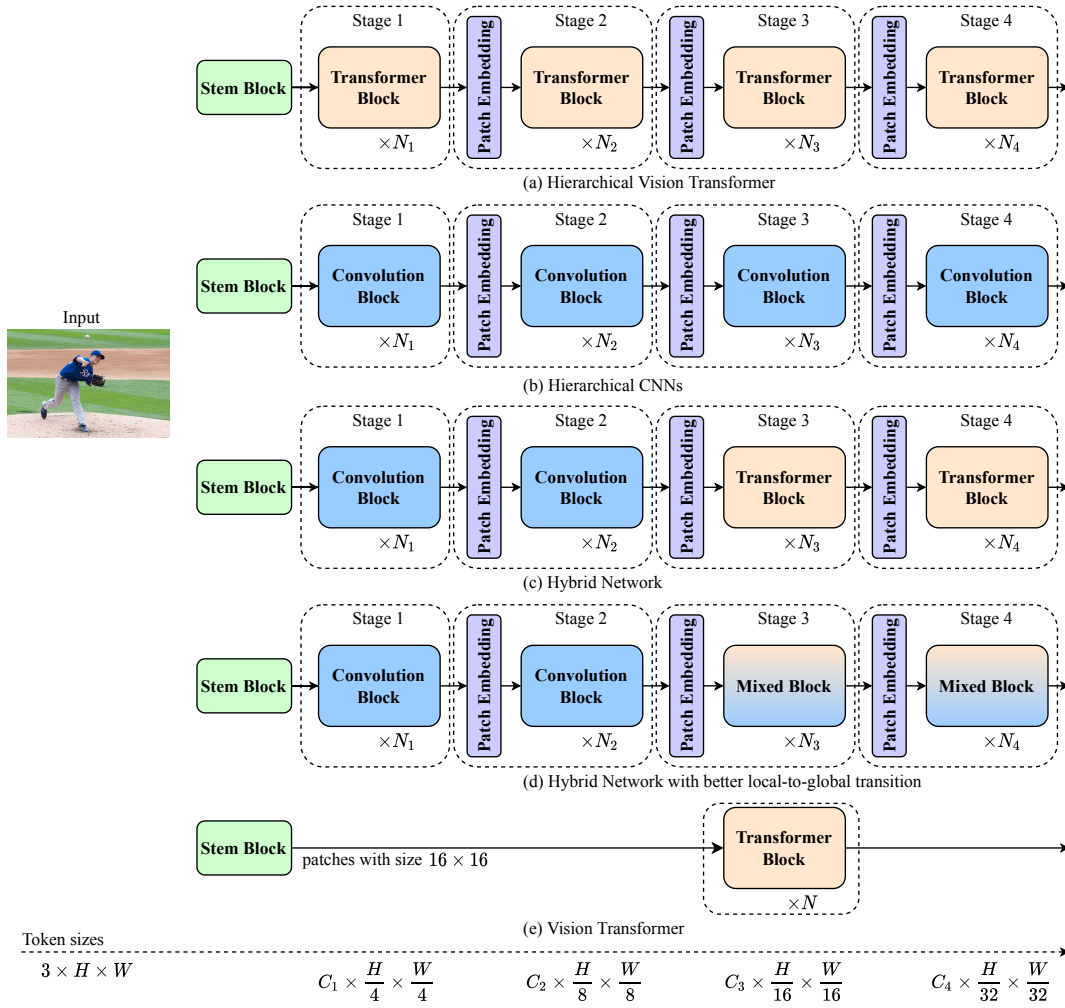


FIGURE 2.2: Common overview of different networks in literature.

and decoder to the object detection task and opens new views in modeling image and video data. In 2021s, ViT (Dosovitskiy et al., 2021; Touvron et al., 2021) fully adopts the Transformer encoder to image classification and achieves promising performances compared to CNNs counterparts. The overview of ViT is sketched in Figure 2.2(e), only containing one stage. The detailed architecture of ViT is shown in Figure 2.3(a). To process visual data with high dimensions, ViT splits images into a sequence of patches and considers one patch with size  $16 \times 16$  as one token. And using Transformer encoders learn interactions between tokens that produce global features. As described in Figure 2.3(b), the Transformer block consists of two main parts: spatial mixing (Norm1 and multi-head self-attention), and channel mixing (Norm2 and 2 fully-connected layers in MLP Mixer). In recent years, modern networks have focused on multi-head self-attention part to improve the cost and also feature learning.

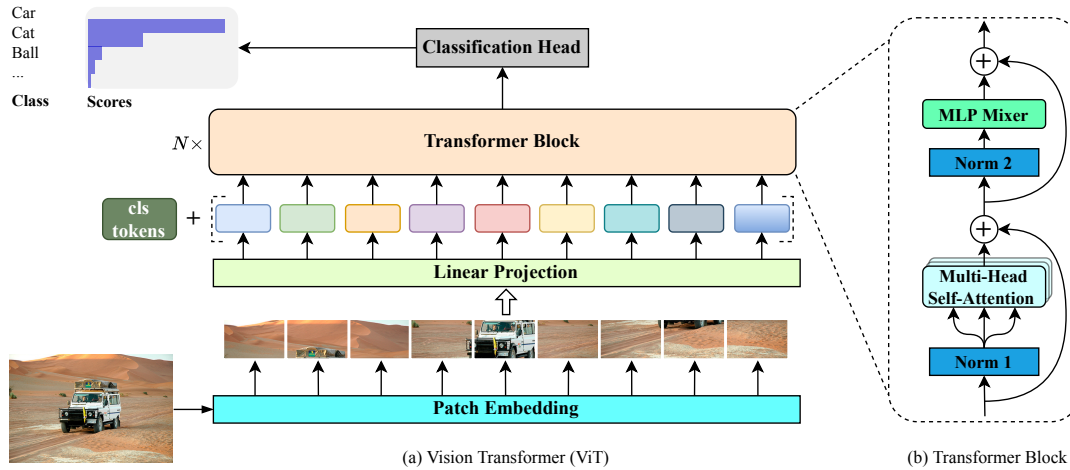


FIGURE 2.3: Detailed architecture of ViT in (a) and Transformer block in (b).  $N$  is the number of stacked blocks.

As a heart of ViT, multi-head self-attention without inductive biases results in long-range dependencies from a sequence of patches. The illustration of multi-head self-attention is shown in Figure 2.4. The core idea of self-attention operation is to compute pair-wise relation between query and key tokens via dot-product. The  $\text{softmax}()$  is performed on each row of the pair-wise matrix to produce attention weights. Figure 2.5 illustrates learnable weights of convolution and attention weights of self-attention operations. Convolution has local receptive fields while self-attention produces global receptive fields and the attention weights generalize better than convolution. Although self-attention has high flexibility in capturing global context features, its mechanism results in quadratic complexity with token length -  $\mathcal{O}(N^2)$ . Existing methods introduce sparse attention to reduce the cost of self-attention. In literature, there are two kinds of sparse attention: spatial reduction attention (Wang et al., 2021a; Wang et al., 2022b; Zhang and Yang, 2021; Zhang and Yang, 2022; Xia et al., 2022) and local-self-attention (Vaswani et al., 2021; Liu et al., 2021b; Dong et al., 2022; Tu et al., 2022).

PVT (Wang et al., 2021a; Wang et al., 2022b) leverages the benefits of ViT into downstream tasks by designing hierarchical backbones and introducing spatial reduction attention (SRA) to reduce the complexity of self-attention. The common hierarchical ViT is illustrated in Figure 2.2(a), which is similar to the design of hierarchical CNNs. DAT (Xia et al., 2022) replaces SRA with deformable attention. To augment weak inductive biases of ViT, several methods (Chen et al., 2022b; Wu et

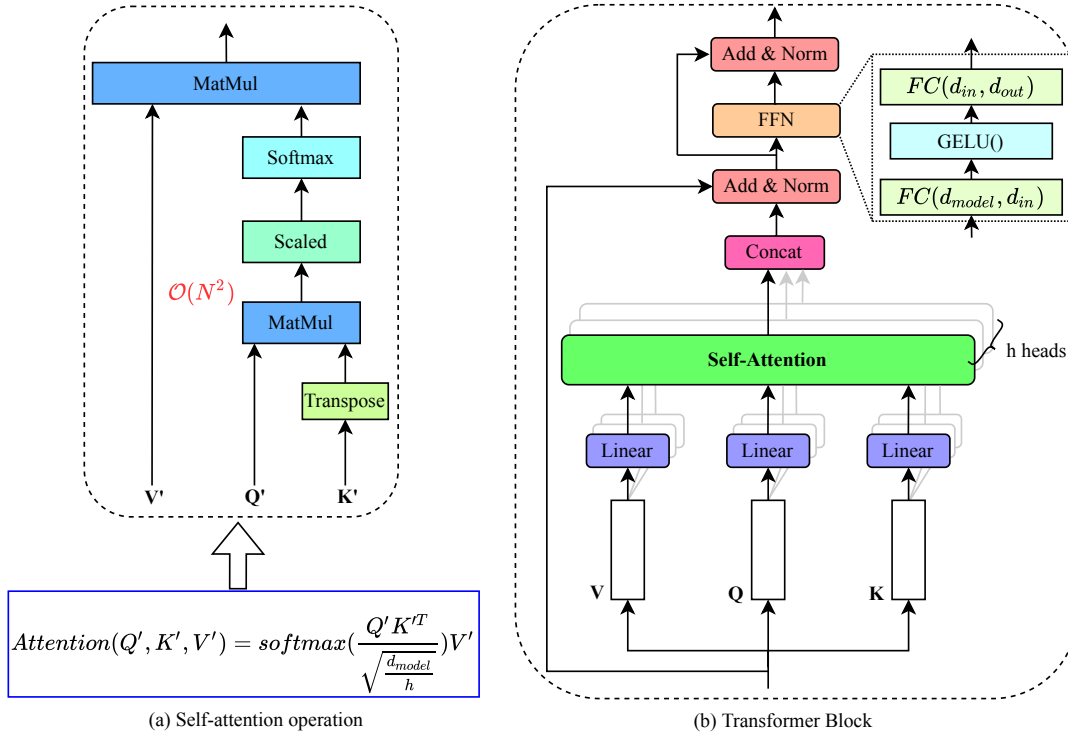


FIGURE 2.4: (a) Attention mechanism of query-key interactions and (b) Transformer block with linear projections and detailed components of FFN.

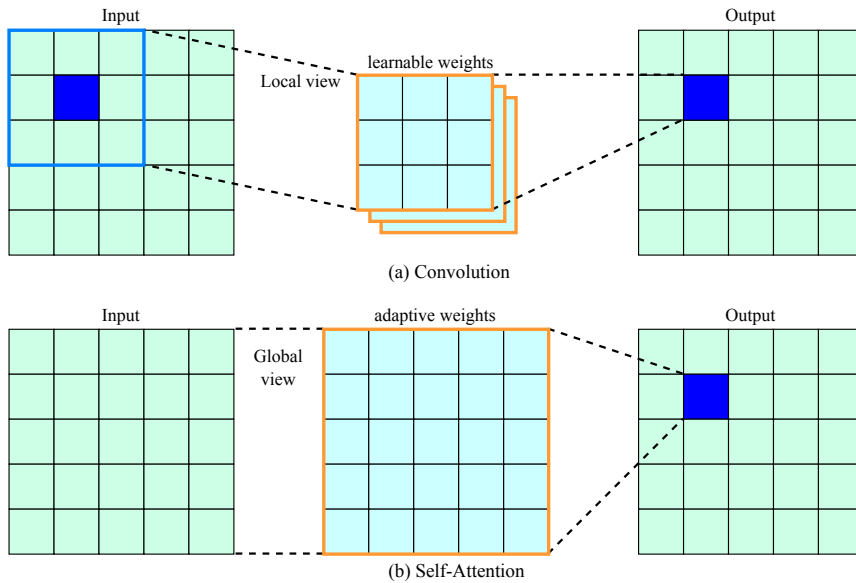


FIGURE 2.5: The main difference between convolution and self-attention.

al., 2021; Guo et al., 2022a; Wang et al., 2022b; Zhang and Yang, 2022) internally employ convolution into self-attention layers and achieve great improvements without pretraining on large-scale datasets. Based on the relative position of convolution, Twins (Chu et al., 2021b), CPVT (Chu et al., 2023), and CSWin Transformer (Dong

et al., 2022) substitute absolute positional encoding in ViT by convolution. Other research externally combines the merits of convolution and self-attentions to build hybrid models (Mehta and Rastegari, 2022; Mehta and Rastegari, 2023; Zhang et al., 2023; Li et al., 2023; Li et al., 2022a; Chen et al., 2022a; Pan et al., 2022a; Hatamizadeh et al., 2023a; Si et al., 2022).

### 2.2.2 Local Vision Transformer

The key idea of local vision Transformers is to limit self-attention to local windows and require further operations/layers to exchange information across windows that result in larger receptive fields. HaloNet (Vaswani et al., 2021) separates the query feature into non-overlapped windows, and the key & value features into overlapped windows using sliding `torch.roll()` - `Im2Column` with zero-padding to slightly enlarge receptive fields. This implementation results in high memory access and high latency. Swin Transformer Liu et al., 2021b partitions feature maps into  $7 \times 7$  windows and self-attention is computed within these regions. As windows are non-overlapped, further operations are needed to communicate between them such as window shifting (Liu et al., 2021b; Liu et al., 2022a), window expanding (Dong et al., 2022), window shuffling (Huang et al., 2021), and window sliding (Chen et al., 2022a; Pan et al., 2023). Figure 2.6 illustrates how information is exchanged across windows. Slide-Transformer (Pan et al., 2023) initialize the weights of `DWConv` with special values to shift the features towards different locations. CSWin Transformer (Dong et al., 2022) and Pale Transformer (Wu et al., 2022) partition the input features into cross-shaped windows and apply self-attentions on these windows, resulting in larger receptive fields. CrossFormer (Wang et al., 2022c) replaces shifted window attentions in Swin Transformer with long-distance attention where the features are shuffled via `reshape()`.

Instead of internally exchanging information across windows, MixFormer (Chen et al., 2022a) applies overlapped `DWConv` with window self-attention in the parallel scheme for externally modeling cross-window relations. MOAT (Yang et al., 2023), EMO (Zhang et al., 2023) implements non-overlapped local-self-attention and `DWConv` in sequential manners.

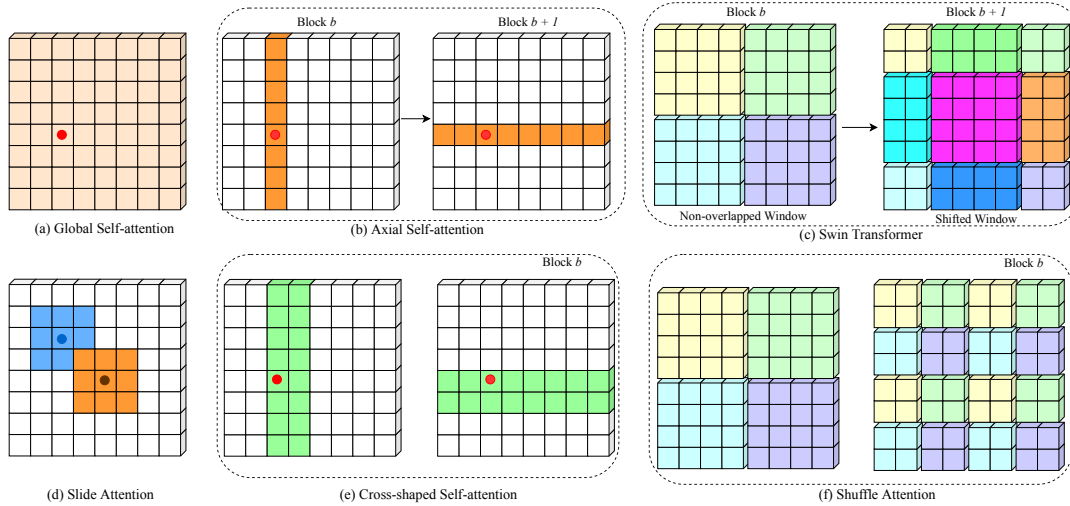


FIGURE 2.6: Various self-attention strategies. A reference point is denoted by a circle dot, attending to the shallow region.

## 2.3 Hybrid Networks

Vanilla convolution has locality and translation equivalence while self-attention operation models long-range spatial dependencies from the input data. Leveraging these merits into one network can improve performance and model optimization. Two examples of hybrid networks are shown in Figure 2.2(c) and (d). Earlier stages tend to learn low-level information easily extracted by convolution block and Later stages capture high-level features via token-to-token interactions of Transformer blocks.

### 2.3.1 Convolution to Self-Attention Layers

Convolution can model content-to-content relations and geometric information (relative position) inside each window. Self-attention ignores the orders of the image tokens and results in weak inductive biases. Hence, many studies (Wang et al., 2021a; Wang et al., 2022b; Graham et al., 2021; Liu et al., 2021b; Chu et al., 2021b; Chu et al., 2023; Dai et al., 2021b; Guo et al., 2022a; Dong et al., 2022; Wang et al., 2022c) are proposed to compensate for the lack of geometric modeling. PVT (Wang et al., 2021a) uses depthwise convolution that can keep the order of tokens and also subsample key and value pairs. PVTv2 (Wang et al., 2022b) adopts convolution in MLP layers to learn local features. To faster converge ViT models, LeViT (Graham et al., 2021) stacks a sequence of convolution in the stem block. Swin Transformer (Liu

et al., 2021b) injects relative positions of windows into attention weights as attention biases. Twins (Chu et al., 2021b), CPVT (Chu et al., 2023) also investigates the role of inductive biases in Transformer models and proposes conditional position encoding using convolution that puts between blocks. Inspired by Swin, CoAtNet (Dai et al., 2021b) integrates relative positions between tokens into global self-attention. CMT (Guo et al., 2022a), and CSWin(Dong et al., 2022) also employs  $3 \times 3$  convolutions in linear projections (Guo et al., 2022a), in value branch (Dong et al., 2022).

### 2.3.2 Self-Attention Layers to Existing CNNs

Another line of hybrid networks is to insert Transformer blocks into stages of existing CNNs. MobileViT (Mehta and Rastegari, 2022) attaches self-attention layers to stages 3, and 4 of MobileNetv2 (Sandler et al., 2018) and achieves outstanding improvements compared to the baseline. MobileViTv2 (Mehta and Rastegari, 2023) further reduces the cost of MobileViT by introducing separable self-attention. TopFormer (Zhang et al., 2022) concatenates the multi-scale feature maps and then, uses Transformer blocks for concatenated features. With this scheme, this kind of model achieves better trade-offs between accuracy and speed while capturing better feature representations.

### 2.3.3 Combination of Convolution and Self-Attention Layers

This research combines the best of convolution and self-attention to build hybrid networks. A lot of works are introduced in this branch and attain state-of-the-art performances across visual tasks. EdgeViT (Pan et al., 2022b) proposes local-to-global blocks and can be deployed on mobile devices. Local blocks are implemented by depthwise convolution and global blocks capture long-range dependencies by using self-attention layers. To reduce the cost, before computing attention, EdgeViT samples query, key, and value tokens. Next-ViT (Li et al., 2022a) utilizes group convolution in easier stages and original self-attentions in later stages. MobileFormer (Chen et al., 2022b) proposes group self-attention layers that work with mobile blocks in parallel and surpass existing lightweight networks by a large margin. MOAT (Yang et al., 2023) rethinks the combination of convolution and self-attention and introduces efficient blocks that reorganize the position of convolution



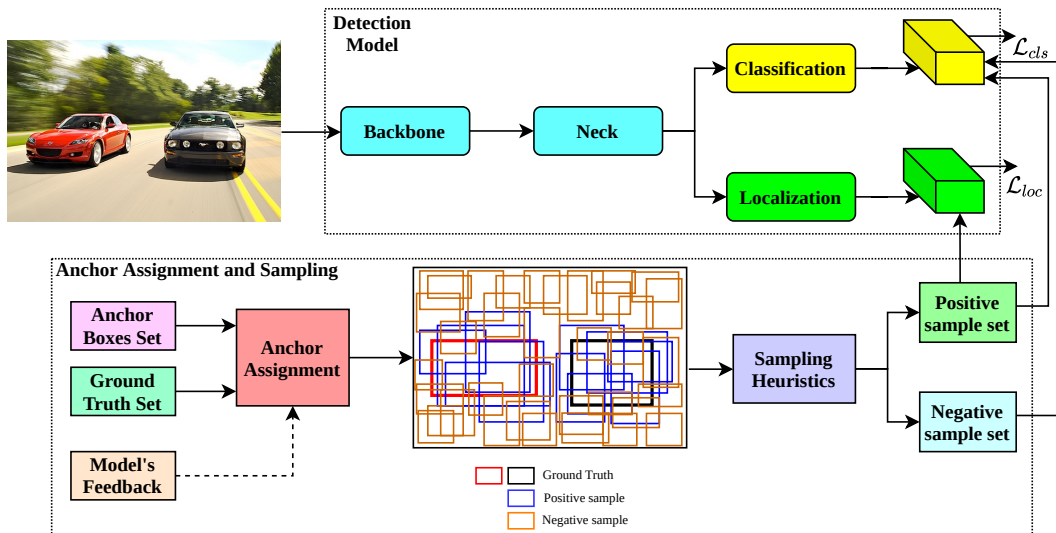


FIGURE 2.7: The common pipeline of the general object detection network. The pipeline includes two main components: Detection model and Anchor assignment & sampling.  $\mathcal{L}_{cls}$  indicates classification loss.  $\mathcal{L}_{loc}$  denotes localization loss.

and self-attention layers. EMO (Zhang et al., 2023) combines window attention and  $3 \times 3$  depthwise convolution on top of MLP blocks and achieves promising accuracy and speed. iFormer (Si et al., 2022) balances the range of frequencies: convolution and max-pooling learn high-frequency components, and self-attention captures low-frequency components. Recent method (Lin et al., 2023) extracts more local information in earlier stages by using multi-scale convolution such as modulation (Yang et al., 2022b).

## 2.4 Object Detection and Instance Segmentation

In this subsection, we briefly describe representative methods in object detection and instance segmentation. Based on prior knowledge (anchor generation, regression variables), there are two types of detectors: anchor-based object detection/segmentation and query-based object detection/segmentation. The common pipeline of the object detection network is illustrated in Figure 2.7.

### 2.4.1 Anchor-based detection and segmentation

Many advanced detectors have been dominated by anchor-based methods categorized into three groups: two-stage object detection, one-stage object detection, and

center-based object detection.

**Two-stage object detection.** The family of RCNN (Girshick et al., 2014; Girshick, 2015; Ren et al., 2015) has been pioneering works in two-stage anchor-based methods. RCNN applies a selective search algorithm for generating many region proposals (about 2000 region proposals for each image). Then, the region-wise CNNs extract features and classify each region proposal using SVM. Instead of forwarding the region proposals to CNNs, Fast R-CNN directly feeds the input image to the CNN to create the feature map. Then, they generate the region proposals from the feature map using selective search and reshape them into a fixed size ( $7 \times 7$ ) utilizing the RoI pooling layer. The classification scores and regressed bounding box for each proposal are predicted through stacked fully connected layers from pooled features. Although Fast R-CNN reduces training and testing time, region proposals generated by the selective search are a matter in Fast R-CNN architecture since selective search is a slow and time-consuming step. To overcome this problem, Faster R-CNN introduces an anchor generation mechanism to create dense anchor boxes (prior bounding boxes). In general ways, multiple anchors of different scales and aspect ratios are placed to each feature map location to encompass all objects with various sizes and shapes. Faster R-CNN includes two stages: Region Proposal Network (RPN) and region-wise RCNN. In the first stage, RPN uses two CNN sub-networks to predict objectness scores and regressed offsets from the set of anchor boxes. The main goal of RPN is to reduce the number of negative samples by eliminating low-quality bounding boxes via Non-Maximum Suppression (NMS), i.e., the suppressed boxes have low objectness scores. To train RPN, anchor boxes are separated into two sets: a set of negative samples and a set of positive samples. The bounding box regression only refines bounding boxes of positive samples. In the second stage, the RCNN network further processes filtered bounding boxes in RPN to get final detection results in which RoI/RoIAlign is used to crop refined bounding boxes before feeding to classification and regression networks.

Based on object detection models, instance segmentation methods are grouped into two kinds: top-down methods and bottom-up methods. Top-down methods (He et al., 2017a; Bolya et al., 2019; Liu et al., 2021a; Chen et al., 2019a) heavily rely on the bounding box predictions of detection task, while bottom-up methods (Xie et al., 2020; Wang et al., 2020a; Wang et al., 2020b) learn affinity embedding of the same

instances and different instances and require extra post-processing to distinguish object instances.

**One-stage object detection.** Without RPN, one-stage detectors directly predict object classification scores and regression offsets at each spatial location from assigned dense anchor boxes, which balances accuracy and speed. The representative methods of one-stage detectors are SSD (Liu et al., 2016) and its variants (Fu et al., 2017), YOLO family (Redmon and Farhadi, 2018; Bochkovskiy, Wang, and Liao, 2020; Wang, Bochkovskiy, and Liao, 2021), and RetinaNet (Lin et al., 2017b). SSD places anchor boxes on multiple feature map with different scale and then, directly predicts object categories and box offsets. RetinaNet improves the one-stage network in many aspects, such as applying a feature pyramid (Lin et al., 2017a) for solving scale imbalance in which anchor boxes are densely tiled on each feature map; proposing Focal loss to handle foreground/background imbalance; and designing classification and regression sub-networks. Nowadays, one-stage object detectors achieve similar performance with two-stage methods but higher testing speed than two-stage detectors.

**Center-based object detection.** Recently, many researchers have great attention to anchor-free methods due to their high efficiency and flexibility. Anchor-free object detections directly output object categories and bounding box regression without designing anchor boxes.

**Center-based object detection.** Center-based methods consider the center point or center region of the object as a strict criterion to define positive and negative samples. During training, these methods regress the distance offsets from positive samples to four sides of the object boundary. YOLO (Redmon et al., 2016) separates the input image into an  $S \times S$  grid. If the center of an object belongs to a grid cell, that grid cell is used to detect that object. GA-RPN (Wang et al., 2019) determines the pixels inside the center region of a ground truth box as positive samples and then predicts the anchor location and shape. FSAF (Zhu, He, and Savvides, 2019) defines the center region of an object as positive according to the prediction of the anchor-free branch with a feature selection module integrated into the detection head. FoveaBox (Kong et al., 2020) defines the region inside the middle part of an object as a positive area and then predicts four distance offsets from each cell inside the positive area to the object boundary. FCOS (Tian et al., 2019) considers anchor boxes as anchor

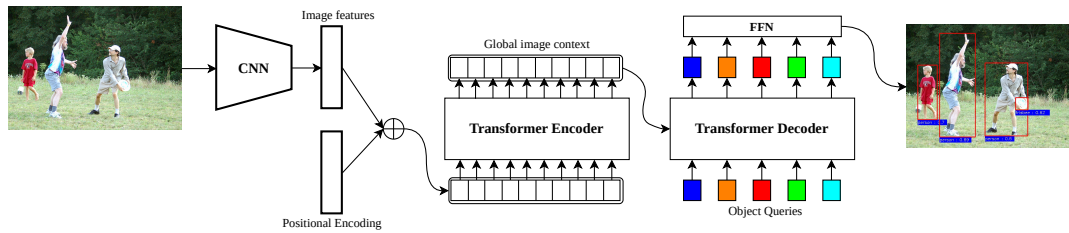


FIGURE 2.8: Firstly, DETR utilizes the CNN network to perform feature extraction from the input image and produces a low spatial resolution feature map. Secondly, this feature map is flattened to the image feature vector, which is suitable for the Transformer computation (e.g., this vector is considered as a sequence of tokens). The positional encoding is added with the feature vector to serve as input of the Transformer. Thirdly, the Transformer encodes the relationship between a token and other tokens, and outputs the global image context. Fourthly, the Transformer decoder reasons about the relations of learnable object queries and the global contextual feature. Fifthly, FFNs are feed-forward networks prediction, designed as classification and regression branches to directly generate the final prediction set of class scores and bounding box coordinates.

points, eliminating hyperparameter selections of anchor boxes such as how many anchor boxes are tiled per spatial location, scale, and aspect ratio. If an anchor point falls into the object region, this point is assigned as a positive sample and utilized to regress distance offsets from this point to each side of the object boundary.

## 2.4.2 Query-based Detection and Segmentation

Inspired by the success of the attention mechanism in visual tasks, in recent years, many researchers have adapted and facilitated Transformer (Vaswani et al., 2017) architecture to object detection, which achieves significant improvements in both global computation and performance, and establishes new state-of-the-art detectors on the challenging benchmark (Lin et al., 2014). Transformer architecture originally was designed for a sequence-to-sequence machine translation, which became the de-facto standard method in most natural language processing. The core element of the Transformer is the self-attention block that models long-range dependencies in data. This promising property brings many advantages to solving visual tasks such as general modeling capacity (relation of pixel-to-pixel, pixel-to-object, object-to-object), self-attention to complement CNNs, powerful operation because of adaptive computation, unified modeling between vision and language, and scalability in both model and data. This promising property brings many advantages to solving

visual tasks such as general modeling capacity (relation of pixel-to-pixel, pixel-to-object, object-to-object), self-attention to complement CNNs, powerful operation because of adaptive computation, unified modeling between vision and language, and scalability in both model and data.

DETR (Carion et al., 2020) is the first end-to-end method that performs interaction learning through Transformer operation to reason about detection results without any specific hand-crafted assumptions. The overall network of DETR is shown and described in Figure 2.8. In recent years, many methods have improved DETR architecture in various aspects such as efficient self-attention design (Zhu et al., 2021; Dai et al., 2021a), object query improvement (Gao et al., 2021; Meng et al., 2021; Yao et al., 2021; Wang et al., 2021b), and Transformer encoder-decoder improvement (Sun et al., 2021b; Fang et al., 2021).

In this research, the feature extractors are trained on large-scale datasets to learn general visual features. After training is finished, the learned weights are fine-tuned for object recognition by using existing object detection/segmentation methods, such as RetinaNet (Lin et al., 2017b), Mask R-CNN (He et al., 2017a), FCOS (Tian et al., 2019), DETR (Carion et al., 2020), to clarify the quality of proposed vision Transformers and deploy unified models for practical devices.

## Chapter 3

# Efficient Multi-scale Spatial Interactions

### 3.1 Introduction

An image can be interpreted as a set of pixels, and each pixel contains content (intensity values) -  $x \in \mathbb{R}^3$  and geometric information -  $w \in \mathbb{R}^2$ . The way the model extracts the features relies on how content-to-content interactions ( $x_i$  vs.  $x_j$ ) and relative geometric ( $w_{i-j}$ ) between  $x_i$  and  $x_j$  are encoded. The convolution layer models content-to-content relationships inside local windows via sliding kernels and also captures the relative position between them. With this scheme, the convolution layer has strong inductive biases such as locality and translation equivalence. To extract long-range spatial interactions, CNN-based models require more stacked convolution layers. Otherwise, as the heart of Vision Transformer (Dosovitskiy et al., 2021), self-attention operations capture long-range patch-to-patch interactions through dot product between query and key features but fail to encode relative positions. The lack of relative positions  $w_{i-j}$  results in weak inductive biases and the ViT-based models need a lot of data to compensate for the absence of relative positions. Other methods supplement absolute or relative positional encoding into Transformer blocks to mimic the feature learning of the convolution layer.

With the success of the ViT model, modern networks focus on the improvements of self-attention layers in both spatial interactions and computational cost. For example, bringing innovative designs of CNNs can enhance network capacities. PVT (Wang et al., 2021a) adopts the hierarchical design of CNNs (He et al., 2016) into

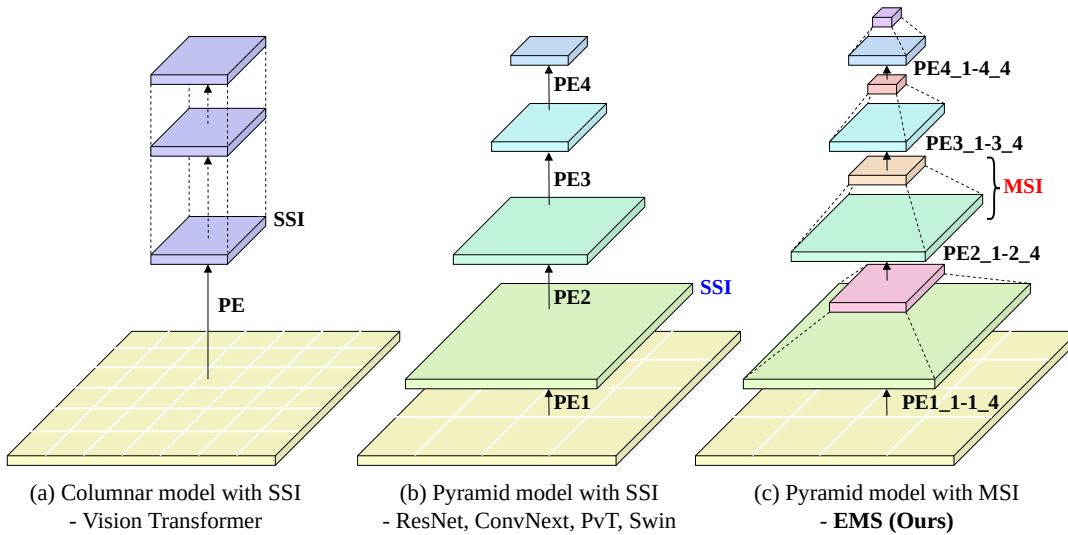


FIGURE 3.1: Network comparison between methods. **SSI**: Single-Scale Interaction, **MSI**: Multi-Scale Interaction, and **PE**: Patch Embedding. (a) ViT (Dosovitskiy et al., 2021) uses columnar style for feature extractor. (b) The recent works such as ResNet (He et al., 2016), ConvNext (Liu et al., 2022b), PVT (Wang et al., 2021a), Swin (Liu et al., 2021b) adopt the hierarchical structure and each stage performs single-scale interactions. (c) Efficient Multi-scale Spatial Interaction (EMS - Ours): At each stage, spatial interactions are performed on multiple feature scales generated based on different PEs with different patch sizes.

Vision Transformer (Dosovitskiy et al., 2021), and achieves great performance on upstream and downstream tasks. Compared to the columnar style in ViT, the pyramid network is separated into four stages and the spatial sizes across stages are progressively decreased. However, each stage only learns token interactions on a single scale and limits the ability of feature representations. To address this issue, at each stage of the pyramid model, we extend the feature learning from single-scale to multi-scale interactions (MSI) to capture multiple views of the input. Figure 3.1 shows the network comparison between existing methods and the proposed method.

In the network budgets, ViT-based models are parameter-efficient but FLOPs-inefficient. Global self-attention layers do not require more parameters since the attention weights are dependently generated with the input instead of input-independent in convolution. Originating from query-key interactions, the self-attention layer has quadratic complexity with spatial dimensions. With this drawback, many works attempt to relax the complexity of global self-attention such as decreasing the size of query, key and value matrices (Wang et al., 2021a; Wang et al., 2022b), limiting self-attention operation inside window (Liu et al., 2021b), inserting Transformer

blocks to latter stages to build hybrid networks (Mehta and Rastegari, 2022; Mehta and Rastegari, 2023; Wadekar and Chaurasia, 2022; Vaswani et al., 2021). Following these lines of research, this paper introduces a new and efficient self-attention operation called Convolution-based Multi-head Self-attention (C-MHSA). C-MHSA brings self-attention into convolution instead of convolution into self-attention operation. The benefits of the C-MHSA are that: (1) Attention weight generation is input-dependent like self-attention; (2) Relative position is preserved via unfold and fold operations like convolution; (3) The computational cost has linear complexity with input tokens.

To efficiently extract proper features from the input, this work attempts to adopt various operators on multi-scale features, and each operator is performed on each specific feature. Specifically, through the channel-splitting mechanism, multiple features are created and processed by multiple spatial mixers followed by patch embedding layers with different patch sizes. For the features with large scales, depthwise convolution, and C-MHSA operations are used to learn the high-frequency component. And multi-head self-attention (MHSA) operation is performed on the features with the smallest scales to learn patch-to-patch interactions. This placement can avoid the quadratic complexity problem and enlarge the general capability of the hybrid model. Since different operators are used on different branches in parallel, the network can effectively capture the information from the input, e.g., a wide range of frequencies and multi-order interactions.

Briefly, the key contributions of this paper are summarized as follows:

1. At each stage of EMSNet, token mixers are performed on multi-scale features with progressive shrinking resolutions. Local and global mixers are adopted for features with high and low resolutions. This design brings two benefits: (1) extracting both short-range and long-range interactions in one layer and hence, enhancing the general modeling capacities of the model; (2) easily performing spatial interactions of global MHSA on the coarse features at earlier stages and hence, avoiding the large cost of the MHSA in lower layers. Each stage of the EMSNet has a pyramid structure and can benefit downstream tasks that require multi-scale interactions.



2. The EMS block combines the strengths of the proposed coordinate convolution, local self-attention, and global self-attention via a simple channel-splitting strategy and patch embeddings with various patch sizes. Convolution and local self-attention are performed on the features with small patch sizes to capture high-frequency components. Global self-attention is used for the feature with a large patch size to extract low-frequency components. Therefore, the EMS block can learn a wide range of frequencies from the input. In another aspect, leveraging the sparseness of the input tokens into feature learning can enhance interaction complexity. Existing works capture low- and high-order interactions while the EMSNet forces the model to focus on multi-order interactions.
3. For local self-attention, C-MHSA is proposed to compute self-attention inside the windows. The C-MHSA unifies the insightful properties of convolution (relative position) and local self-attention (input-dependent attention weights) into one operation. In our implementation, the input feature is partitioned into overlapped windows via Im2Col operation, while Swin Transformer has non-overlapped windows and needs cyclic-shifted window operation to communicate information across windows. Previous local self-attentions only have input-dependent attention weights, while C-MHSA has both input-dependent attention weights and learnable kernels. Hence, our design has stronger generalization capacities.

In the following, the paper is organized as follows: Section 3.2 describes representative methods related to our work; Section 3.3 analyzes the proposed EMSNet network; Section 3.4 shows the implementation detail and experimental results; Section 3.5 discusses the insightful properties of our designs.

## 3.2 Related Works

### 3.2.1 Convolutional Neural Networks.

In 2012s, AlexNet (Krizhevsky, Sutskever, and Hinton, 2017) successfully trained the CNN model on large-scale ImageNet dataset (Russakovsky et al., 2015) and established a new record in the ranking of classification research. From this milestone,

many works were proposed with more innovative designs. VGG (Simonyan and Zisserman, 2014) stacks a sequence of plain  $3 \times 3$  convolution layers to 16 or 19 layers. ResNet (He et al., 2016) investigates the vanishing gradient problem when stacking plain layers with more than 20 layers. From the observation, ResNet proposed a residual connection that can help network optimization and extend the network deeper. With the success of these designs, ResNet has become the dominant method for vision tasks. Other works improve the ResNet model with dense residual connection, Inception module (Szegedy et al., 2015), deformable convolution. The common point of the CNN-based models is that the network extracts the features in a hierarchical manner. Earlier layers capture low-level features with high-frequency components and the latter layers extract high-level features with low-frequency patterns. Alternatively, with EMS block, this paper can balance the ranges of frequencies across layers and serve as the bridge between CNNs and ViT models.

### 3.2.2 Vision Transformers

Transformer architecture was originally developed for natural language processing and became a de-facto model for language tasks. In 2021s, ViT (Dosovitskiy et al., 2021) applies the Transformer network for vision tasks (Luo et al., 2023; Sun et al., 2023; Jing, Meng, and Hou, 2023) and set a new view to extract features via the query-and-key mechanism without considering geometric information. Based on the strong modeling capacity and generalization of ViT, recent methods focus on the improvement of token mixers in the model efficiency and adaptation, and leverage the design of convolution into the self-attention layer. TNT (Han et al., 2021) performs spatial interactions on multiple patch sizes. Methods in (Wang et al., 2021a; Wang et al., 2022b; Heo et al., 2021) develop the hierarchical backbone and reduce the cost of self-attention layers by sampling key and value matrices. Swin (Liu et al., 2021b) computes spatial interactions inside non-overlapped windows and requires additional designs to exchange information across windows. Differently, this paper brings hierarchical design into stages and enhances modeling capability. The proposed C-MHSA with learnable and adaptive weights computes attention in overlapped windows and does not need extra designs.

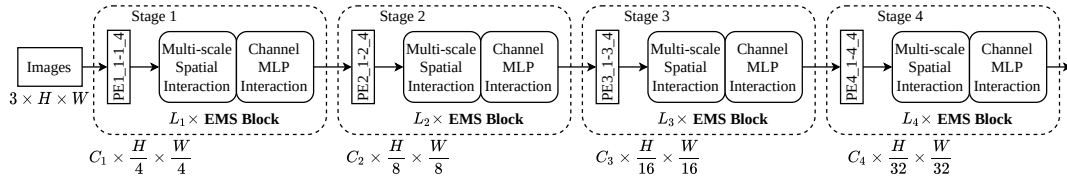


FIGURE 3.2: The overall architecture of EMSNet. The network is separated into 4 stages and each stage includes 4 patch embedding layers (PE<sub>i\_1-i\_4</sub>) and  $L_i \times$  EMS Block. Each EMS block contains a Multi-scale Spatial Interaction (MSI) layer and a Channel Multi-Layer Perceptron (MLP) Interaction layer. Similar to hierarchical pyramid networks (He et al., 2016; Wang et al., 2021a; Liu et al., 2021b), the spatial dimension of the input image is progressively down-sampled across stages.

### 3.2.3 Hybrid models

With the combination of convolution and Transformer block, hybrid models achieve state-of-the-art results on vision tasks. MobileViT (Mehta and Rastegari, 2022; Mehta and Rastegari, 2023; Wadekar and Chaurasia, 2022) inserts Transformer blocks into stages 3, and 4 of the MobileNetv2 (Sandler et al., 2018). Another line of research is to replace self-attention operation in Transformer blocks with other token mixers. PoolFormer (Yu et al., 2022) utilizes non-learnable  $3 \times 3$  max pooling as a mixer. EdgeViT (Pan et al., 2022b) adopts depthwise convolution and self-attention as the local and global mixer in a serial way. Inspired by the success in Swin (Liu et al., 2021b), ConvNext (Liu et al., 2022b) replaces  $7 \times 7$  windows with  $7 \times 7$  depthwise convolution and achieves similar performances with Swin. VAN (Guo et al., 2022b) and ParC-Net (Zhang, Hu, and Wang, 2022) extend the idea of ConvNext by enlarging kernel sizes. With the paradigm of input-dependent weights, several methods (Jin et al., 2023; Wang et al., 2023) bring the dynamic property of self-attention to convolution and achieve promising performance on various tasks. Inspired by hybrid models, this paper efficiently combines the strengths of convolution and self-attention operations by performing spatial interactions on features with various scales and the channel-splitting mechanism.

### 3.3 Method

Existing works (He et al., 2016; Wang et al., 2021a; Wang et al., 2022b; Liu et al., 2021b) divide the network into four stages, and the size of the feature map is progressively shrunk across stages. Each stage processes a single-scale feature map through a patch embedding layer with a fixed patch size and a stack of token mixer blocks that exchange information among these patches. However, interacting spatial information on a single-scale feature map can limit the general modeling capacities of the model.

In this paper, we perform spatial interactions on large ranges of scales that can benefit downstream tasks such as object detection and segmentation. The hierarchical pyramid network of the EMSNet is described in Figure 3.2, and has four stages similar to (He et al., 2016; Wang et al., 2021a; Liu et al., 2021b). Given the input image with dimension  $3 \times H \times W$ , we first generate multi-scale feature maps with dimension  $\{3p_i^2 \times \frac{H}{p_i} \times \frac{W}{p_i} | i \in \{1, 2, 3, 4\}\}$  using multiple patch embedding layers with patch sizes  $\{p_1, p_2, p_3, p_4\}$ . Inside each patch embedding, a linear projection layer is used to control the width of the model. Then, various spatial mixers in the EMS block are applied to these multi-scale feature maps. Similarly, in stages 2-4, the spatial dimension of the feature maps is reduced by 8, 16, and 32 pixels with respect to the input image, and the EMS blocks model short-range and long-range interactions from generated multi-scale feature maps.

#### 3.3.1 EMS Block

The detailed architecture of the EMS block is shown in Figure 3.3. Performing the spatial interactions on original feature maps causes high computational cost and inefficient parameters. Inspired by the Inception module (Szegedy et al., 2015) with parallel branches, the original input is split into sub-tensors along the channel dimension. Rather than using linear transformations to split the original input in Inception, this work directly creates split tensors by slice operation with more efficiency.

Specifically, given the input  $\mathbf{X} \in \mathbb{R}^{C_i \times H_i \times W_i}$ , two sub-features:  $\mathbf{X}_s \in \mathbb{R}^{C_s \times H_i \times W_i}$

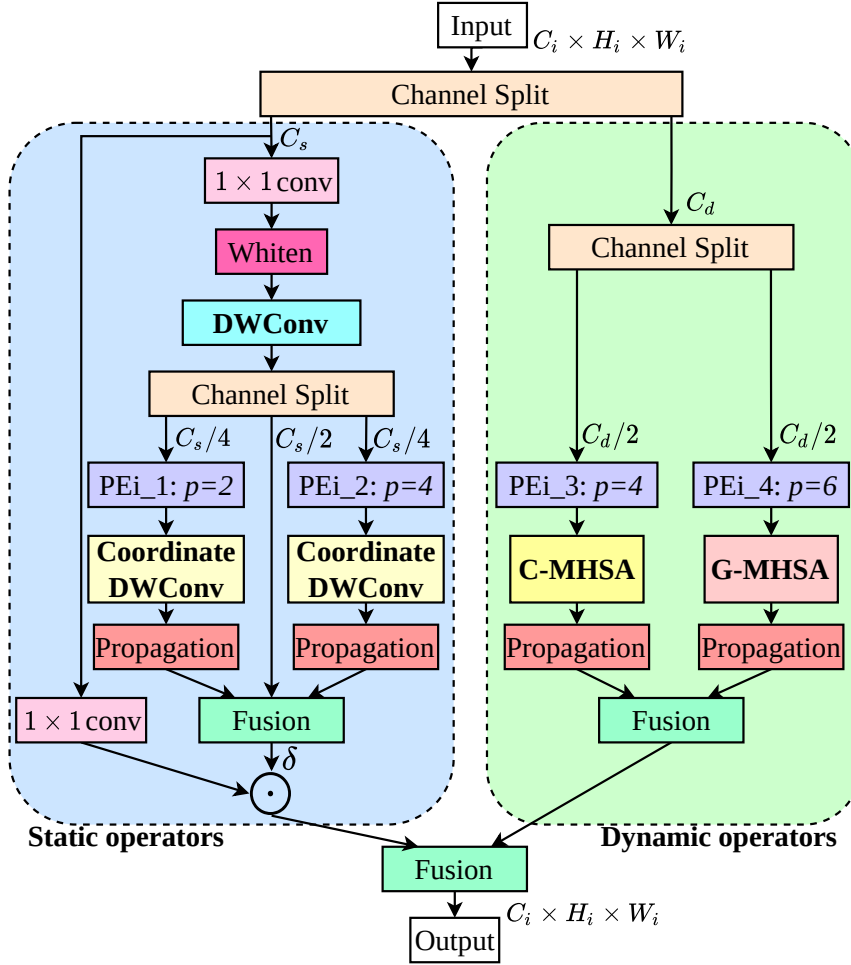


FIGURE 3.3: The detailed structure of the EMS block. **DWConv**, **Coordinate DWConv** indicates depthwise convolution and coordinate depthwise convolution. **C-MHSA**, **G-MHSA** are Convolution-based Multi-head Self-Attention, Global Multi-head Self-Attention operations.  $\delta$  is the sigmoid function and  $\odot$  is element-wise matrix multiplication.  $i \in \{1, 2, 3, 4\}$  indicates the index of the Patch Embedding (PE) with patch size  $p$ .

and  $\mathbf{X}_d \in \mathbb{R}^{C_d \times H_i \times W_i}$  are generated, with  $C_s + C_d = C_i$ . The two features are processed through two branches with two spatial mixers: (1) Static operator (convolution) and (2) Dynamic operator (self-attention). The difference between static and dynamic operators is the way we generate the model weights in input-independent and input-dependent manners. The static and dynamic branches are to learn short-range and long-range spatial interactions between patch tokens that are crucial for visual representation. Both operators can help each other (Liu et al., 2021b), and leveraging this property in learning visual features can result in better generalization performance.

### Static operator

The goal of the static operator is to make the model focus on high-frequency components and medium-order interactions. Directly modeling local fine-grained features from the original input can impair the performance of the model and general capacities. To eliminate the global components of the input features, whiten transformation is used as:

$$\mathbf{X}_w = \text{Conv}_{1 \times 1}(\mathbf{X}_s) - \text{mean}(\text{Conv}_{1 \times 1}(\mathbf{X}_s)), \quad (3.1)$$

where  $\mathbf{X}_s$  are the split feature from the original input  $\mathbf{X}$ , and  $\text{mean}$  denotes mean operation computing along the channel axis (*Its analysis is described in Appendices*). Then, the disentangled feature  $\mathbf{X}_w$  is fed into the depthwise convolution (DWConv) to create the mixed feature  $\mathbf{X}_m$ , shown in Figure 3.3.

To capture the information of the input with different levels, we divide the main features  $\mathbf{X}_m$  into three features:  $\mathbf{X}_{m1} \in \mathbb{R}^{C_s/4 \times H \times W}$ ,  $\mathbf{X}_{m2} \in \mathbb{R}^{C_s/2 \times H \times W}$ , and  $\mathbf{X}_{m3} \in \mathbb{R}^{C_s/4 \times H \times W}$ . On each branch, a stack of Patch Embedding (PE) with patch size  $p$ , Coordinate depthwise convolution, and Propagation Layers is adopted. The use of the PE with various patch sizes and convolution on each sampled feature can learn multi-scale spatial interactions and enhance the network's modeling ability. Technically, the PE equally samples a sub-set of the static feature across spatial space and one output token represents  $p \times p$  window. As a result, the sparse feature is generated, and the convolution aggregates the information of delegate tokens. After exchanging the information among sampled tokens, the Propagation layer propagates this information back to the neighbor pixels. Consequently, the outputs of multi-scale patch mixers are calculated as:

$$\mathbf{Y}_{m1} = \text{Prop}(\text{CoordDWConv}(\text{PEi}_1(\mathbf{X}_{m1}))), \quad (3.2)$$

$$\mathbf{Y}_{m2} = \text{Prop}(\text{CoordDWConv}(\text{PEi}_2(\mathbf{X}_{m3}))), \quad (3.3)$$

And one output feature is computed by concatenating the output of local mixers as:

$$\mathbf{Y}_c = \text{Conv}_{1 \times 1}(\text{Concat}(\mathbf{Y}_{m1}, \mathbf{X}_{m2}, \mathbf{Y}_{m2})), \quad (3.4)$$

where  $\text{CoordDWConv}$ ,  $\text{Prop}$  are coordinate depthwise convolution and propagation

operation. In this paper, PE layers are implemented by a max-pooling with kernel size  $p$ , stride  $p$ , and a  $1 \times 1$  convolution to map the input to a higher dimension. We also decompose 2D DWconv into horizontal DWConv and vertical DWConv called CoordinateDWConv. This strategy can preserve the property of the convolution and reduce the computational cost by a large number of FLOPs (*proof in Appendices*).

To efficiently distribute the mixed information of represented tokens to its local neighborhood, depthwise transposed convolutions with kernel size  $p$ , and stride  $p$  are utilized as propagation. Then, the features of multi-scale spatial interaction branches are fused through concatenation, and  $1 \times 1$  conv mixes the fused feature along the channel dimension. Inspired by gated aggregation, element-wise matrix multiplication between the mixed feature after sigmoid and the input feature after linear projection is performed to obtain the final output feature. Formally, this process is addressed as:

$$\mathbf{Y}_s = \text{Conv}_{1 \times 1}(\mathbf{X}_s) \odot \delta(\mathbf{Y}_c), \quad (3.5)$$

where  $\delta$  is sigmoid function and  $\odot$  is element-wise matrix multiplication.

### Dynamic operator

Given the input feature  $\mathbf{X}_d \in \mathbb{R}^{C_d \times H \times W}$ , we efficiently separate the feature  $\mathbf{X}_d$  into two parts:  $\mathbf{X}_{d1} \in \mathbb{R}^{C_d/2 \times H \times W}$  and  $\mathbf{X}_{d2} \in \mathbb{R}^{C_d/2 \times H \times W}$ . Then, two parts are processed by convolution-based multi-head self-attention addressed in the next section and global multi-head self-attention (G-MHSA).

The vanilla (G-MHSA) (Dosovitskiy et al., 2021) operation is used to model long-range spatial interactions among all patch tokens that can produce global features with low-frequency patterns. The main bottleneck of G-MHSA comes from matrix multiplication between the query and key matrices in the self-attention operation that requires quadratic complexity with the spatial size. Using G-MHSA blocks at earlier stages can cause high computation costs. To take full advantage of the strong modeling capacities of self-attention, similar to the static operators, we adopt a sequence of the PE, G-MHSA, and Propagation layers. With this design, the G-MHSA layer is performed on the sparse feature with the smaller sizes (stride 6 with respect to the input feature) and the cost is largely reduced. Finally, the global information

modeled in the sampled patches is propagated to its neighbor patches via the Propagation layer implemented by lightweight depthwise transposed convolution. The output of this branch can be computed as

$$\mathbf{Y}_{d2} = \text{Prop}(\text{G-MHSA}(\text{PEi}_4(\mathbf{X}_{d2}))), \quad (3.6)$$

Similarly, the C-MHSA branch can be defined as

$$\mathbf{Y}_{d1} = \text{Prop}(\text{C-MHSA}(\text{PEi}_3(\mathbf{X}_{d1}))). \quad (3.7)$$

Finally, the outputs of the dynamic operators are fused along the channel dimension:

$$\mathbf{Y}_d = \text{Conv}_{1 \times 1}(\text{Concat}(\mathbf{Y}_{d1}, \mathbf{Y}_{d2})). \quad (3.8)$$

### 3.3.2 C-MHSA Operation

In recent years, many methods have attempted to limit self-attention inside local windows through special operators. Swin (Liu et al., 2021b) uses window partition and window reverse block to perform local self-attention. However, two complex operators are unfriendly deployed on real hardware (Pan et al., 2022b). To overcome this problem, this paper proposes a C-MHSA operation based on simple operators and brings the designs of self-attention to convolution.

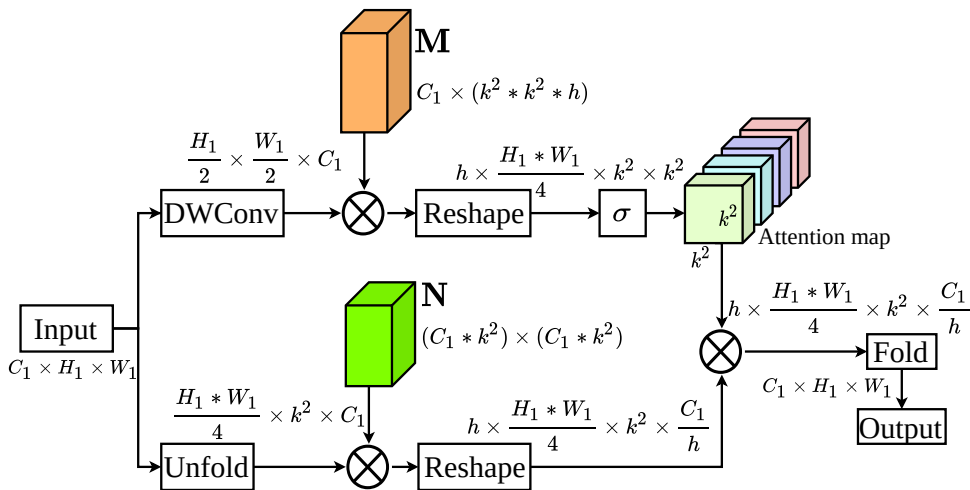


FIGURE 3.4: The detailed architecture of the C-MHSA operation.  $k$  is kernel size and  $h$  is number of heads. The function  $\sigma$  is softmax.



The vanilla convolution is equivalent to a process of Unfold, Matrix Multiplication, and Fold operators. The key design of the C-MHSA is to unify insightful properties of convolution (relative position via Unfold and Fold operators), and local self-attention (attention weights conditioned on the input) into one operation. With this combination, the C-MHSA operation can improve the feature learning of the model. The detailed analysis of the C-MHSA is shown in Figure 3.4.

Mathematically, given the feature  $\mathbf{X}_{d1} \in \mathbb{R}^{C_d/2 \times H \times W}$ , the patch embedding is used to split the feature  $\mathbf{X}_{d1}$  into a sequence of patches and in this paper, we set the patch size to 4 for reducing the cost. After processing the patch embedding, the feature  $\mathbf{X}'_{d1}$  has the dimension  $C_1 \times H_1 \times W_1$ . In the C-MHSA operation, there are two main processes: (1) Generating an attention map; (2) Unfolding, matrix multiplication between the attention map and unfolded input, and Folding.

To generate the attention map  $\mathbf{A}$ , firstly, the Depthwise convolution (DWConv) with kernel size  $3 \times 3$  and stride 2 is employed to shrink the size of the input by two times in overlapped window. The matrix multiplication (MM) between  $\mathbf{M} \in \mathbb{R}^{C_1 \times (k^2 * k^2 * h)}$  and the shrunk features is computed to model the similarity map that one position of the output map is generated based on aggregating information from all positions inside  $k \times k$  window. Then, the softmax function  $\sigma$  is applied to compute the attention map:

$$\mathbf{A} = \sigma(\text{MM}(\mathbf{M}, \text{DWConv}(\mathbf{X}'_{d1}))). \quad (3.9)$$

For the second process, the unfold operation with kernel size  $k$ , stride 2, and the linear projection  $\mathbf{N}$  are adopted. To perform context aggregation like the G-MHSA operation, matrix multiplication between the projected input and attention matrix is employed to generate the output feature after folding. Formally, the output of the C-MHSA is defined as:

$$\mathbf{Y}_{d1} = \text{Fold}(\text{MM}(\mathbf{A}, \text{MM}(\mathbf{N}, \text{Unfold}(\mathbf{X}'_{d1}))))). \quad (3.10)$$

With these designs, the C-MHSA inherits both input-dependent attention weights and learnable kernels. Therefore, it brings stronger generalization capacities.

For computing attention maps, the G-MHSA operation requires quadratic complexity with spatial sizes, causing heavy computation. While the proposed C-MHSA operation has quadratic complexity with a small kernel size ( $3 \times 3$ ), it verifies the efficiency of our method.

### 3.3.3 EMS Variants

Based on the proposed EMS Block, we design EMS variants described in Table 3.1 (G is GFLOPs, #p(M) is a number of parameters (millions)).  $[C_1, C_2, C_3, C_4]$  and  $[L_1, L_2, L_3, L_4]$  are the number of channels and number of EMS blocks across four stages, respectively. Following exiting works (Mehta and Rastegari, 2022; Wang et al., 2021a; Liu et al., 2021b; Yu et al., 2022), the number of EMS blocks and channel dimensions is gradually increasing. The Channel MLP Interaction contains two  $1 \times 1$  convolution layers. The first convolution is used to expand the number of channels by the MLP ratio addressed in Table 3.1 and the second convolution maps the number of channels to the original input. In the first stage, a stem block (PE1) including two  $3 \times 3$  convolution layers with kernel size 3, and stride 2 is employed to down-sample the input image by 4 times. In other stages, patch embedding layers used in static and dynamic branches are a stack of max-pooling with kernel size  $p_i$ , stride  $p_i$  and  $1 \times 1$  convolution for the static process, and a stack of adaptive pooling with kernel size  $p_i$ , stride  $p_i$  and  $1 \times 1$  convolution for the dynamic process. A set of four patch sizes in the EMS block are configured to  $p_i \in \{2, 4, 4, 6\}$ . The channel split ratios, and kernel sizes in static operators of the EMS block are analyzed in Appendices.

TABLE 3.1: Three EMS models with different width and depth settings

| Model         | $[C_1, C_2, C_3, C_4]$ | $[L_1, L_2, L_3, L_4]$ | MLP       | G   | #p(M) |
|---------------|------------------------|------------------------|-----------|-----|-------|
| EMSNet-XXTiny | [32, 64, 128, 192]     | [2, 2, 4, 2]           | [8,8,4,4] | 0.5 | 2.5   |
| EMSNet-XTiny  | [32, 64, 96, 128]      | [3, 3, 10, 2]          | [8,8,4,4] | 0.7 | 3.0   |
| EMSNet-Tiny   | [64, 96, 128, 256]     | [3, 3, 10, 2]          | [8,8,4,4] | 1.9 | 5.4   |

## 3.4 Experiments

In this section, we verify the effectiveness of the proposed EMSNet by conducting extensive experiments on image classification, object detection, and instance segmentation tasks.

### 3.4.1 ImageNet1k Image Classification

**Settings.** For the classification task, we only train and evaluate the EMSNet on the large-scale dataset ImageNet-1K (Russakovsky et al., 2015) without using pre-trained models on ImageNet-22K. The ImageNet-1K dataset contains 1.2M training images and 50K images for validation.

Following standard settings of existing works (Liu et al., 2021b; Wang et al., 2021a; Yu et al., 2022), all the EMS models are trained for 300 epochs from scratch and the image size is  $224 \times 224$ . The batch size for one GPU is 256 and the total batch size is 512 for two GPU Tesla V100. The optimizer is AdamW configured with a weight decay of 0.03, and a momentum of 0.9. The basic learning rate is  $1e^{-3}$  and changed based on a cosine learning scheduler with 5 warmup epochs. For fair comparisons, we employ data augmentation and regularization techniques used in Swin (Liu et al., 2021b) such as Label Smoothing, Mixup, Cutmix, RandAugment, and stochastic depth. All the experiments are conducted by Pytorch framework with the codebase Timm and automatic mixed precision for faster training.

**Results.** Table 3.2 shows the performance comparison of lightweight models on the validation set of ImageNet-1K (G is GFLOPs, #p(M) is number of parameters (millions), and Top-1 denotes Top-1 accuracy (%)). Type indicates used token mixers in the network including convolution (Conv), Attention (Attn), and the combination of Conv and Attn (Hybrid). As a result, with similar parameters and GFLOPs, the EMSNet outperforms state-of-the-art lightweight models by a clear margin. For the smallest version, the EMSNet-XXTiny achieves 73.1 % Top-1 accuracy that is greater than MobileViTv1-XXS (Mehta and Rastegari, 2022) by 4.1%, MobileViTv2-0.5 (Mehta and Rastegari, 2023) by 2.9%, PVTv2-B0 (Wang et al., 2022b) by 2.6%, and current method MobileViTv3-0.5 by 0.8%. For the EMSNet-XTiny with 3.0M parameters and 0.7 GFLOPs, we establish the new record on the ImageNet-1K dataset and

TABLE 3.2: Image classification results of lightweight networks for various types of models on ImageNet-1K

| Method  | Type   | Size | #p(M) | G   | Top-1 (%)   |
|---|--------|------|-------|-----|-------------|
| MobileViTv1-XXS (Mehta and Rastegari, 2022)   | Hybrid | 256  | 1.3   | 0.4 | 69.0        |
| MobileViTv2-0.5 (Mehta and Rastegari, 2023)   | Hybrid | 256  | 1.4   | 0.5 | 70.2        |
| PVTv2-B0 (Wang et al., 2022b)                 | Attn   | 224  | 3.7   | 0.6 | 70.5        |
| MobileViTv3-0.5 (Wadekar and Chaurasia, 2022) | Hybrid | 256  | 1.4   | 0.5 | 72.3        |
| ResNet18 (He et al., 2016)                    | Conv   | 224  | 11.7  | 1.8 | 69.8        |
| TNT-Ti (Han et al., 2021)                     | Attn   | 224  | 6.1   | 1.4 | 73.9        |
| EdgeViT-XXS (Pan et al., 2022b)               | Hybrid | 256  | 4.1   | 0.6 | 74.4        |
| Swin-0.7G (Liu et al., 2021b)                 | Attn   | 224  | 4.4   | 0.7 | 74.4        |
| PiT-Ti (Heo et al., 2021)                     | Attn   | 224  | 4.9   | 0.7 | 74.6        |
| MobileNetv2-1.4 (Sandler et al., 2018)        | Conv   | 224  | 6.9   | 0.6 | 74.7        |
| MobileViTv1-XS (Mehta and Rastegari, 2022)    | Hybrid | 256  | 2.3   | 1.0 | 74.8        |
| ConvNeXt-XTiny (Liu et al., 2022b)            | Conv   | 224  | 4.4   | 0.7 | 75.1        |
| PVTv1-Tiny (Wang et al., 2021a)               | Attn   | 224  | 13.2  | 1.9 | 75.1        |
| VAN-B0 (Guo et al., 2022b)                    | Hybrid | 224  | 4.1   | 0.9 | 75.4        |
| MobileViTv2-0.75 (Mehta and Rastegari, 2023)  | Hybrid | 256  | 2.9   | 1.0 | 75.6        |
| PoolFormerS12 (Yu et al., 2022)               | Hybrid | 224  | 11.9  | 1.8 | 77.2        |
| Swin-1G (Liu et al., 2021b)                   | Attn   | 224  | 7.3   | 1.0 | 77.3        |
| EdgeViT-XS (Pan et al., 2022b)                | Hybrid | 256  | 6.7   | 1.1 | 77.5        |
| MobileViTv1-S (Mehta and Rastegari, 2022)     | Hybrid | 256  | 5.6   | 4.0 | 78.4        |
| ParC-Net-S (Zhang, Hu, and Wang, 2022)        | Conv   | 256  | 5.0   | 3.5 | 78.6        |
| PVTv2-B1 (Wang et al., 2022b)                 | Attn   | 224  | 13.1  | 2.1 | 78.7        |
| EMSNet-XXTiny (Ours)                          | Hybrid | 224  | 2.5   | 0.5 | 73.1        |
| EMSNet-XTiny (Ours)                           | Hybrid | 224  | 3.0   | 0.7 | 77.1        |
| EMSNet-Tiny (Ours)                            | Hybrid | 224  | 5.4   | 1.9 | <b>79.3</b> |

surpass strong models such as EdgeViT-XXS (Pan et al., 2022b) by 2.7%, MobileViTv2-0.75 (Mehta and Rastegari, 2023) by 1.5%, ConvNeXt-XTiny (Liu et al., 2022b) by 2%, Swin-0.7G (Liu et al., 2021b) by 2.7%, and similar Top-1 accuracy with Swin-1G with only 70% GFLOPs. Surprisingly, when only changing the channel number of the EMS-XTiny, we achieve 79.3% Top-1 accuracy that outperforms PVTv1-Tiny (Wang et al., 2021a) by 4.2% with only 41% parameters, PVTv2-B1 (Wang et al., 2022b) by 0.6% with only 41% parameters, ParC-Net-S (Zhang, Hu, and Wang, 2022) by 0.7% with only 54% GFLOPs.

### 3.4.2 MS-COCO Object Detection and Segmentation

**Settings.** The EMSNet variants are evaluated on MS-COCO object detection and segmentation. The MS-COCO dataset (Lin et al., 2014) contains 115K training images, 5k validation images, and 20k testing images with 80 categories. We replace the backbone ResNet-50 in detector RetinaNet (Lin et al., 2017b) and segmentor Mask R-CNN (He et al., 2017a) with our feature extractor EMSNet. Except for the hyperparameters in the EMSNet, all other settings are kept similar to the RetinaNet, and

TABLE 3.3: Object detection results on MS-COCO (Lin et al., 2014) and the detector RetinaNet (Lin et al., 2017b) and the neck FPN

| Backbone                        | #params (M) | GFLOPs | $AP^b$ | $AP_{50}^b$ | $AP_{75}^b$ |
|---------------------------------|-------------|--------|--------|-------------|-------------|
| ResNet-18 (He et al., 2016)     | 21.3        | 188.7  | 31.7   | 49.6        | 33.4        |
| ResNet-50 (He et al., 2016)     | 37.7        | 250.3  | 36.3   | 55.4        | 39.1        |
| PVTv1-Tiny (Wang et al., 2021a) | 23.0        | 183.3  | 36.6   | 56.6        | 38.8        |
| PVTv2-B0 (Wang et al., 2022b)   | 13.0        | 160.4  | 37.1   | 57.2        | 39.2        |
| EdgeViT-XXS (Pan et al., 2022b) | 13.1        | -      | 38.7   | 59.0        | 41.0        |
| EMSNet-XXTiny                   | 11.7        | 162.1  | 37.3   | 57.3        | 39.4        |
| EMSNet-XTiny                    | 12.4        | 167.9  | 39.0   | 59.1        | 41.4        |
| EMSNet-Tiny                     | 14.7        | 190.3  | 41.2   | 61.3        | 44.2        |

Mask R-CNN. Specifically, the model is trained for 12 epochs with a batch size of 4 and the image is resized to  $1333 \times 800$ . The optimizer is AdamW with a learning rate of  $1e^{-4}$ , betas (0.9, 0.999), weight decay 0.05. We use the codebase MMDetection for experiments and the metrics used here are mAP for box and mask results.

TABLE 3.4: The performance of the EMSNet on MS-COCO instance segmentation with Mask R-CNN (He et al., 2017a), the neck FPN

| Backbone                        | #params (M) | GFLOPs | $AP^m$ | $AP_{50}^m$ | $AP_{75}^m$ |
|---------------------------------|-------------|--------|--------|-------------|-------------|
| ResNet-18 (He et al., 2016)     | 31          | 207    | 31.2   | 51.0        | 32.7        |
| ResNet-50 (He et al., 2016)     | 44          | 260    | 34.4   | 55.1        | 36.7        |
| PVTv1-Tiny (Wang et al., 2021a) | 33          | 208    | 35.1   | 57.6        | 37.3        |
| PVTv2-B0 (Wang et al., 2022b)   | 24          | 179    | 36.2   | 57.8        | 38.6        |
| EMSNet-XTiny                    | 23          | 186    | 37.1   | 58.5        | 40.0        |
| EMSNet-Tiny                     | 25          | 209    | 39.0   | 62.1        | 41.9        |

**Results.** The object detection performance is shown in Table 3.3. The pyramid hierarchical network with the EMS blocks can balance the high-level and low-level features, and benefit downstream tasks. As a result, with only 11.7M parameters and 162.1 GFLOPs, the EMSNet-XXTiny outperforms pyramid-based model PVTv1-Tiny (Wang et al., 2021a) by 0.6%  $AP^b$ , with 50.9% parameters and 40.2% GFLOPs. It demonstrates the effectiveness of our method.

Table 3.4 describes the segmentation results on the MS-COCO dataset. With a budget of 23M parameters and 186 GFLOPs, the EMSNet-XTiny outperforms both CNN-based model ResNet (He et al., 2016) and ViT-based model PVT (Wang et al., 2021a; Wang et al., 2022b) by large margins but smaller complexity.

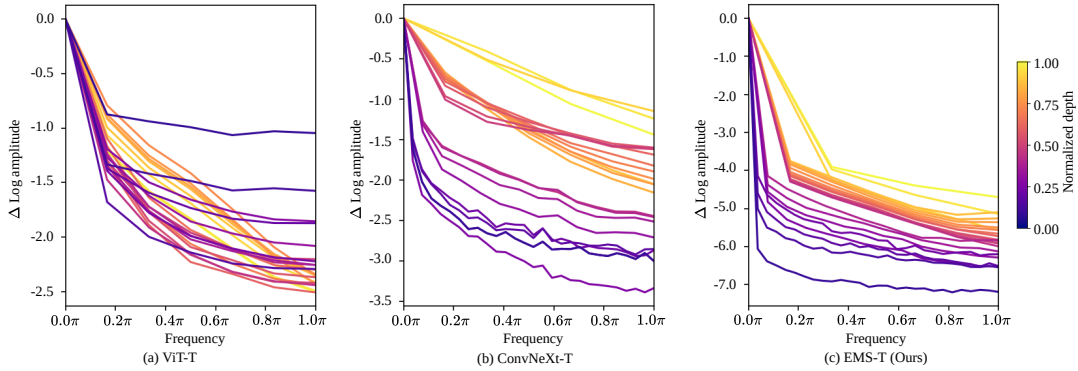


FIGURE 3.5: Relative log amplitudes of the Fourier transformed feature maps of ViT-T (Dosovitskiy et al., 2021), ConvNeXt-T (Liu et al., 2022b), and our EMSNet-Tiny.  $\Delta$  log amplitude measures the difference between log amplitude at scaled frequency  $0.0\pi$  (center) and at  $1.0\pi$  (boundary).

### 3.5 Investigation of the EMS Block

To further analyze the behavioral learning of the EMS blocks, we provide ablation studies, comparison with other token mixers, the Fourier spectrum of the EMSNet, and the interaction strength of multi-scale sparse token mixers following (Deng et al., 2022).

TABLE 3.5: Ablation study of the EMS block

| Operations     |                         | #params (M) | GFLOPs | Top-1 |
|----------------|-------------------------|-------------|--------|-------|
| Static branch  | Baseline                | 2.09        | 0.48   | 70.2  |
|                | +Whiten                 | 2.24        | 0.51   | 70.5  |
|                | +PEi:p=2, CoordDW, Prop | 2.41        | 0.53   | 70.9  |
|                | +PEi:p=4, CoordDW, Prop | 2.38        | 0.53   | 71.1  |
|                | +Gated Aggregation      | 2.68        | 0.59   | 71.9  |
| Dynamic branch | +C-MHSA                 | 3.10        | 0.55   | 72.7  |
|                | +G-MHSA                 | 2.56        | 0.54   | 73.1  |

#### 3.5.1 Ablation studies

Table 3.5 shows the ablation study of the EMSNet-XXTiny model that addresses the effectiveness of each operator in the EMS block. Combining all operators into one block gets better performance with acceptable increases in the number of parameters and GFLOPs.

TABLE 3.6: Comparison of the local token mixers

| Method                         | Token Mixer            | #p(M) | G   | Latency (ms) <sup>†</sup> |      | Top-1 | Top-1 V2 |
|--------------------------------|------------------------|-------|-----|---------------------------|------|-------|----------|
|                                |                        |       |     | CPU                       | GPU  |       |          |
| PVTv2 (Wang et al., 2022b)     | Spatial Reduction Attn | 3.7   | 0.6 | 67.3                      | 0.46 | 70.5  | 58.5     |
| Swin (Liu et al., 2021b)       | Window+Shifted Attn    | 4.4   | 0.7 | 67.3                      | 0.76 | 74.4  | 62.5     |
| ConvNeXt (Liu et al., 2022b)   | 7 × 7 DWConv           | 4.4   | 0.7 | 37.6                      | 0.78 | 75.1  | 63.3     |
| HaloNet (Vaswani et al., 2021) | Local Attention        | 4.4   | 0.7 | 83.7                      | 1.03 | 75.8  | 63.9     |
| EMSNet-XTiny                   | Ours                   | 3.0   | 0.7 | 70.3                      | 0.57 | 77.1  | 65.1     |

<sup>†</sup>The latency is tested on the CPU Intel(R) Core(TM) i5-6600@3.30GHz and the GPU Tesla V100

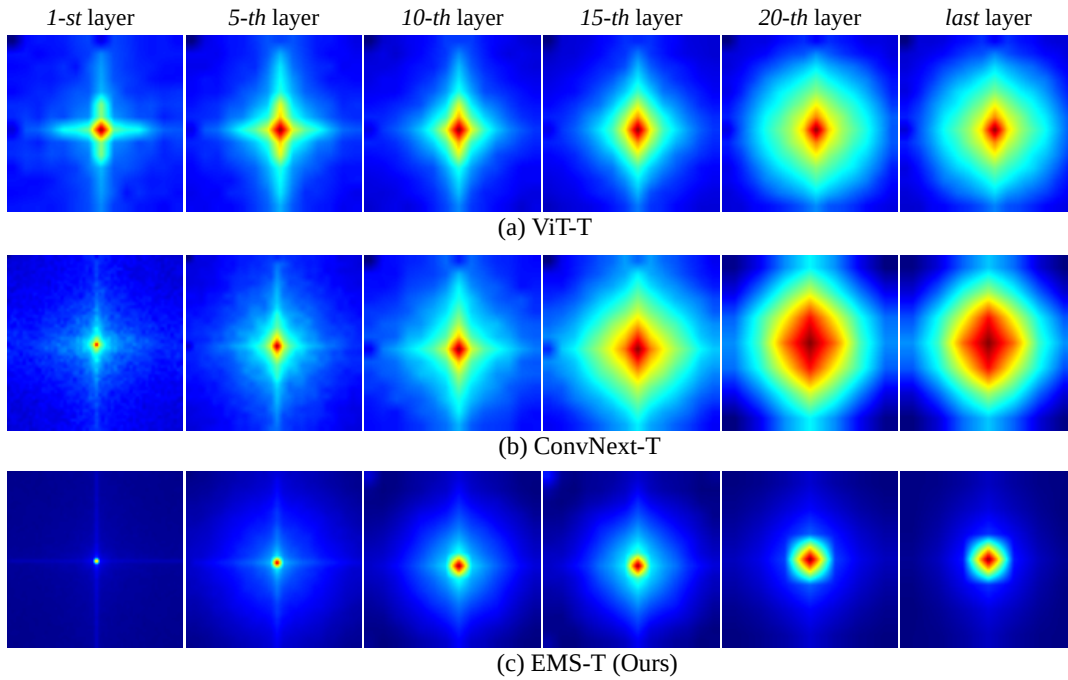


FIGURE 3.6: Amplitude spectrum of ViT (Dosovitskiy et al., 2021), ConvNeXt (Liu et al., 2022b), and EMSNet. The ViT model tends to weaken the high-frequency component and the ConvNext model increases them. Otherwise, our model balances the range of frequencies.

### 3.5.2 Comparison with local token mixers

Table 3.6 reports the comparisons between EMSNet and recent methods about token mixers, accuracy, and latency. Among methods, the EMSNet has lower latency than Swin (Liu et al., 2021b), ConvNeXt (Liu et al., 2022b), HaloNet (Vaswani et al., 2021) on the GPU device while achieving better accuracy. On the CPU device, the EMSNet runs similar speed to PVTv2, Swin, and HaloNet. We also provide performances on ImageNet-v2 matched frequency (Top-1 V2) to address overfitting evaluation. As a result, the EMSNet outperforms PVTv2 by 6.6%, Swin-XT by 2.6%, ConvNeXt by 1.8%, and HaloNet by 1.2%.

### 3.5.3 Fourier analysis

Figure 3.5 and Figure 3.6 describe the relative log amplitudes and amplitude spectrum of the Fourier transformed features of ViT-T (Dosovitskiy et al., 2021), ConvNeXt-T (Liu et al., 2022b), and EMSNet-Tiny. For each network, the feature maps in all blocks (except the MLP Head layer) are transformed to the frequency domain using the 2D Fourier Transform. And only the half-diagonal components of shift Fourier features are visualized because of the conjugate symmetry property of the Fourier Transform. As a result, the EMSNet has the opposite behavior with the ViT-based models and CNN-based models. The ViT model tends to extract low-frequency components (global information). Otherwise, the ConvNeXt model tends to increase the high-frequency components (local information). With the design of the multi-scale spatial interaction block, the EMSNet captures both low-and-high frequency components, resulting in balancing the range of frequencies. This phenomenon can be explained as: in the EMS Block, the static branch focuses on learning high-frequency information, and the dynamic branch model global features with low-frequency patterns. Hence, unifying static and dynamic branches into one layer can achieve better visual representation.

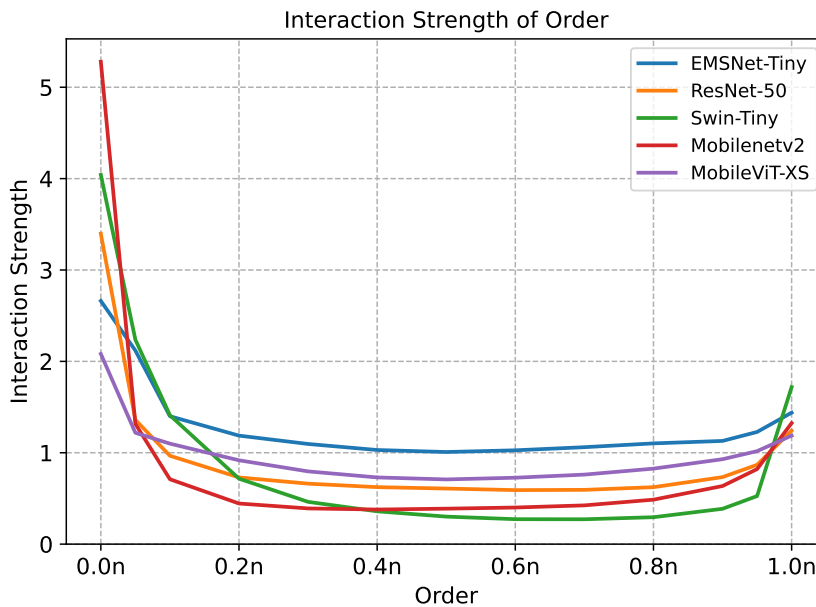


FIGURE 3.7: Interaction complexity of ResNet (He et al., 2016), Swin (Liu et al., 2021b), MobileNet (Sandler et al., 2018), MobileViT (Mehta and Rastegari, 2022), and EMSNet.



### 3.5.4 Interaction strength

The key design of the EMS block is to model spatial interactions from different levels of the input, e.g., dense tokens to sparse tokens, through *PE + Spatial Interaction + Propagation* layers. Figure 3.7 illustrates the interaction strength of order and its computation is shown in Appendices. As a result, ViT and CNN models focus on low-order interactions (sparsest tokens) and high-order interactions (densest tokens). This problem can impair the performance of the model (Deng et al., 2022). In this figure, we observe the values of the EMSNet’s interaction strength are almost higher than other methods. It verifies that the EMSNet can enhance interaction complexity by learning multi-order interactions through sampling the input feature with different rates in PE layers.

## 3.6 Conclusion

This paper proposes a novel Efficient Multi-scale Spatial interaction Network (EMSNet) for learning visual features. Typically, the EMSNet combines the strengths of convolution, global self-attention, and local self-attention (C-MHSA) through a simple channel-splitting strategy and sparse interaction mechanism. With these designs, the model can capture both low- and high-frequency components and force the model to focus on multi-order interactions, resulting in better generalization capacities. Experimental results show that the EMSNet surpasses recent methods with different types (CNN-based, ViT-based, and Hybrid models) on ImageNet-1K image classification, MS-COCO object detection, and instance segmentation. It demonstrates the EMSNet can serve as a general backbone for downstream tasks.

## 3.7 Appendices

### 3.7.1 Analysis of Coordinate DWConv, Whiten Transformation and CMHSA

#### Coordinate DWConv

The  $k_d \times k_d$  depthwise convolution can be decomposed into a couple of  $1 \times k_d$  and  $k_d \times 1$  depthwise convolution dubbed coordinate depthwise convolution. The first  $1 \times k_d$  depthwise convolution aggregates the information of the input feature along the horizontal dimension and the second  $k_d \times 1$  depthwise convolution extracts features along the vertical dimension. This decomposition can complement the vanilla convolution because coordinate convolution well extracts strip objects while vanilla convolution results in redundant semantic features for these objects.

Another reason is that coordinate depthwise convolution is more efficient than vanilla depthwise convolution with  $k_d > 1$ . The FLOPs of coordinate depthwise convolution are calculated as:

$$\underbrace{2 * h * w * c * k_d}_{\text{coordinate dw}} < \underbrace{h * w * c * k_d^2}_{\text{dw}}, \quad (3.11)$$

where  $h$ ,  $w$ , and  $c$  are the height, width and channel of the input feature.

#### Whiten Transformation

The pixel-to-pixel relation is disentangled into whiten pairwise and unary term, defined as:

$$\omega(\mathbf{x}_i, \mathbf{x}_j) = \delta\left(\underbrace{(\mathbf{q}_i - \boldsymbol{\mu}_q)^T (\mathbf{k}_j - \boldsymbol{\mu}_k)}_{\text{whiten}} + \underbrace{\boldsymbol{\mu}_q^T \mathbf{k}_j}_{\text{unary}}\right), \quad (3.12)$$

where  $\delta$  is the softmax function.  $\mathbf{q}_i$ ,  $\mathbf{k}_j$  are a query content  $i$  and a key content  $j$ .  $\boldsymbol{\mu}_q = \frac{1}{|\Omega|} \sum_{i \in \Omega} \mathbf{q}_i$ , and  $\boldsymbol{\mu}_k = \frac{1}{|\Omega|} \sum_{j \in \Omega} \mathbf{k}_j$  are the mean values computed from the set  $\Omega$  of all the pixels.

The whiten term is used to eliminate the global components of the input feature. The second term tends to learn object boundaries. In the EMSNet, we employ whiten

transformation on the static branch because DWConv operation can well capture high-frequency components. To verify this argument, we plot the Fourier spectrum of the input feature and the feature is whitened in Figure 3.8. As a result, whiten layer tends to increase the high-frequency component. With the disentangled feature DWConv can easily extract low-level information. The unary term is omitted on the static branch and learned on the dynamic branch.

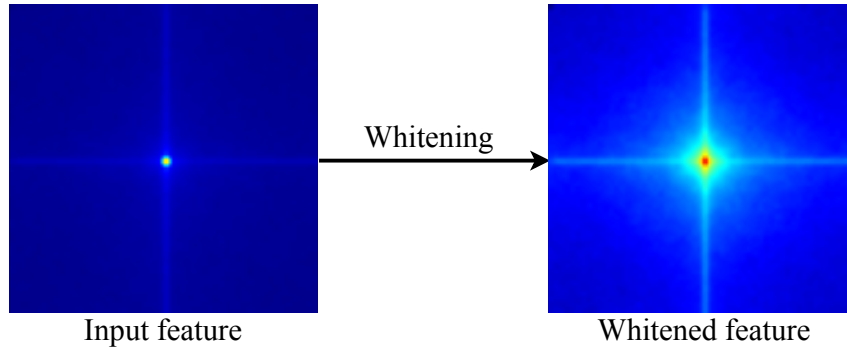


FIGURE 3.8: Whiten transformation on the input feature  $X_s$  of the static branch:  $\text{Conv}_{1 \times 1}(X_s) - \text{Mean}(\text{Conv}_{1 \times 1}(X_s))$ . The result is shown in the frequency domain.

### C-MHSA

Each output position of the DWConv is computed by aggregating the information of the neighbor positions within a local window in an input-independent way. And static weights (kernel weights) are shared across the spatial dimension. While the C-MHSA operation generates dynamic weights (attention weights) conditioned on the image content, it can extract discriminative features. Each value of the attention weight addresses the similarity of  $k * k$  neighbor positions. A visualization of the attention weights of the C-MHSA is shown in Figure 3.9.

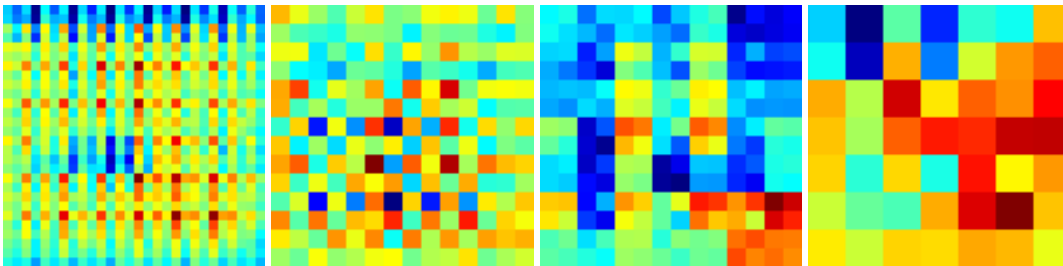


FIGURE 3.9: The attention maps of the C-MHSA operation in 5<sup>th</sup> layer, 10<sup>th</sup> layer, 15<sup>th</sup> layer, and 20<sup>th</sup> layer.

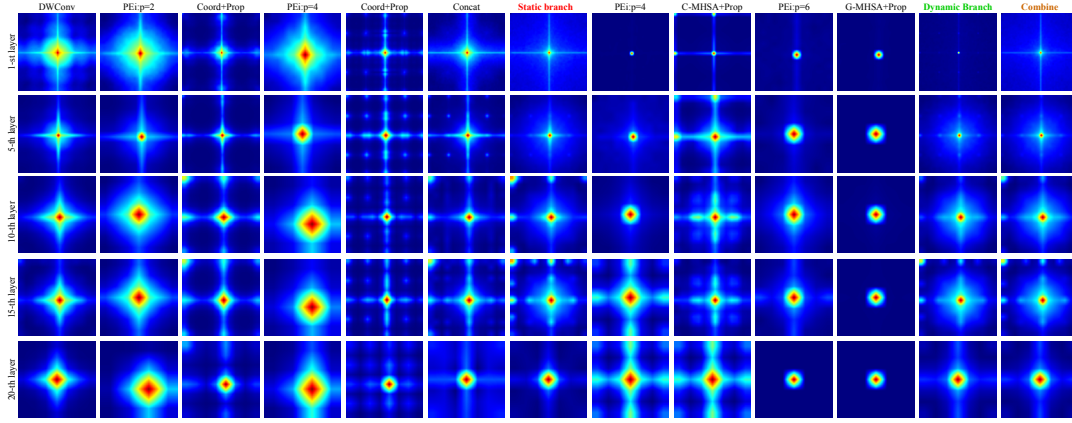


FIGURE 3.10: The amplitude spectrum of the static and dynamic branches of the EMS-Tiny model. The operators in the static branch and C-MHSA tend to increase high-frequency components. G-MHSA tends to capture low-frequency components. Prop denotes Propagation operation.

### 3.7.2 Interaction Strength

Following (Deng et al., 2022), the interaction complexity of existing methods and EMSNet is addressed by the relative interaction strength  $J^m$  of the encoded  $m^{\text{th}}$  order interaction:

$$J^{(m)} = \frac{\mathbb{E}_{x \in \Omega} [\mathbb{E}_{i,j} [ |I^{(m)}(i,j|x)| ]]}{\mathbb{E}_{m'} [\mathbb{E}_{x \in \Omega} [\mathbb{E}_{i,j} [ |I^{(m')}(i,j|x)| ] ]]}, \quad (3.13)$$

$$I^{(m)}(i,j) = \mathbb{E}_{S \subseteq N \setminus \{i,j\}, |S|=m} [\Delta v(i,j,S)], \quad (3.14)$$

where the set of all samples is represented by  $\Omega$ .  $\Delta v(i,j,S) = v(S \cup \{i,j\}) - v(S \cup \{i\}) - v(S \cup \{j\}) + v(S)$  and the output score, denoted by  $v(S)$ , is obtained by keeping the variables in subset  $S$  of  $N$  unchanged while replacing the variables in the complement of  $S$  (i.e.,  $N \setminus S$ ) with the baseline value. The average interaction utility between variables  $i$  and  $j$  is measured by the  $m^{\text{th}}$  order interaction  $I^{(m)}(i,j)$  across all possible contexts comprising  $m$  variables ( $0 \leq m \leq n - 2$ ). The variable  $m$  represents the degree of contextual complexity in the interaction, i.e., How many patches are joined during the interaction. In this work, the patch size is set to  $14 \times 14$ , and the number of patches  $n$  is equal to  $16 * 16$ . Existing networks and the EMSNet are trained on ImageNet-1K with masked rates of the images from 0 (1-order) to 1 (0-order). The comparison of the interaction complexity is shown in Figure 7 of the main paper.

### 3.7.3 Fourier analysis

Figure 3.10 illustrates the amplitude spectrum of the Fourier-transformed features of the static branch, dynamic branch, and its detailed components from 1-st layer to 20-th layer. From the visualization, we observe that:

- Static branch: DWConv operation, PEi:  $p = 2$ , CoordinateDWConv, PEi:  $p = 4$  tend to capture high frequency components. Different branches learn different frequency information.
- Dynamic branch: C-MHSA branch (PEi:  $p = 4$ , C-MHSA + Prop) has a similar trend with operators in the static branch, well capturing high and medium frequency information. Otherwise, G-MHSA branch (PEi:  $p = 6$ , G-MHSA + Prop) focuses on learning low-frequency components.
- As a final result, the EMS block can learn a wide range of frequencies from the input, e.g., low, medium, and high frequencies are extracted at each layer, and enlarge the general modeling capacity of the model.

### 3.7.4 Additional Results

In the static branch, we split the whitened feature into three branches with channel ratio  $\{1/4:1/2:1/4\}$ . Table 3.7 illustrates the contribution of each branch through controlling channel ratios. For the trade-off between accuracy and computation cost, the channel ratio  $\{1/4:1/2:1/4\}$  is selected in all experiments.

TABLE 3.7: Ablation study on channel splitting of the static branch

| Channel ratios    | #params (M) | GFLOPs | Top-1 |
|-------------------|-------------|--------|-------|
| $\{1/4:1/2:1/4\}$ | 2.56        | 0.54   | 73.1  |
| $\{1/2:1/4:1/4\}$ | 2.59        | 0.55   | 73.2  |
| $\{1/4:1/4:1/2\}$ | 2.59        | 0.55   | 73.2  |

Table 3.8 shows the trade-off between accuracy and latency. As a result, our EM-Net has similar speeds with other methods on both CPU and GPU while achieving better performance.

Table 3.9 shows the latency of the components. In our block, DWConv is decomposed into two efficient operations: horizontal and vertical DWConv (less #params

and GFLOPs than DWConv). Existing methods use `torch.split()` or  $1 \times 1$  conv() to split the input while the efficient slice operation  $\mathbf{X}[:, : C_s, :, :]$  is adopted in this work

TABLE 3.8: Latency of lightweight models

| Method                                       | #params (M) | GFLOPs | Latency (ms) |      | Top-1 (%) |
|--|-------------|--------|--------------|------|-----------|
|  |             |        | CPU          | GPU  |           |
| MobileViTv2-0.5 (Mehta and Rastegari, 2023)  | 1.3         | 0.5    | 34.4         | 0.30 | 70.2      |
| PVTv2-B0 (Wang et al., 2022b)                | 3.7         | 0.6    | 67.3         | 0.46 | 70.5      |
| MobileViTv1-XS (Mehta and Rastegari, 2022)   | 2.3         | 1.0    | 99.7         | 0.53 | 74.8      |
| MobileViTv2-0.75 (Mehta and Rastegari, 2023) | 2.9         | 1.0    | 78.9         | 0.44 | 75.6      |
| EdgeViT-XXS (Pan et al., 2022b)              | 4.1         | 0.6    | 32.0         | 0.42 | 74.4      |
| EdgeViT-XS (Pan et al., 2022b)               | 6.7         | 1.1    | 50.9         | 0.51 | 77.5      |
| <i>EMSNet-XXTiny</i>                         | 2.5         | 0.5    | 50.1         | 0.43 | 73.1      |
| <i>EMSNet-XTiny</i>                          | 3.0         | 0.7    | 70.3         | 0.57 | 77.1      |

TABLE 3.9: Latency of each operator in the EMS block

|                       | Branch         | Operations | #p (M) | G    | Latency (ms) |      | Top-1 (%) |
|-----------------------|----------------|------------|--------|------|--------------|------|-----------|
|                       |                |            |        |      | CPU          | GPU  |           |
| Baseline              | Identity()     |            | 2.09   | 0.48 | 33.2         | 0.24 | 70.2      |
| Channel Splitting     | Static branch  | CoordDW    | 2.68   | 0.59 | 43.6         | 0.41 | 71.9      |
|                       | Dynamic branch | +C-MHSA    | 3.10   | 0.55 | 45.4         | 0.39 | 72.7      |
|                       |                | +G-MHSA    | 2.56   | 0.54 | 50.1         | 0.43 | 73.1      |
| w/o fusion            | All            |            | 2.12   | 0.52 | 43.8         | 0.41 | 72.2      |
| w/o channel splitting | All            |            | 4.45   | 0.81 | 90.3         | 0.82 | 74.3      |

### 3.7.5 Grad-CAM Visualization

Figure 3.11 shows the Grad-CAM activation map visualization of our model, PoolFormerS12 (Yu et al., 2022), PVTv2-B1 (Wang et al., 2022b), and ResNet-34 (He et al., 2016). As a result, the EMSNet-Tiny produces more accurate activation maps than other methods and focuses on activating the full semantic parts.

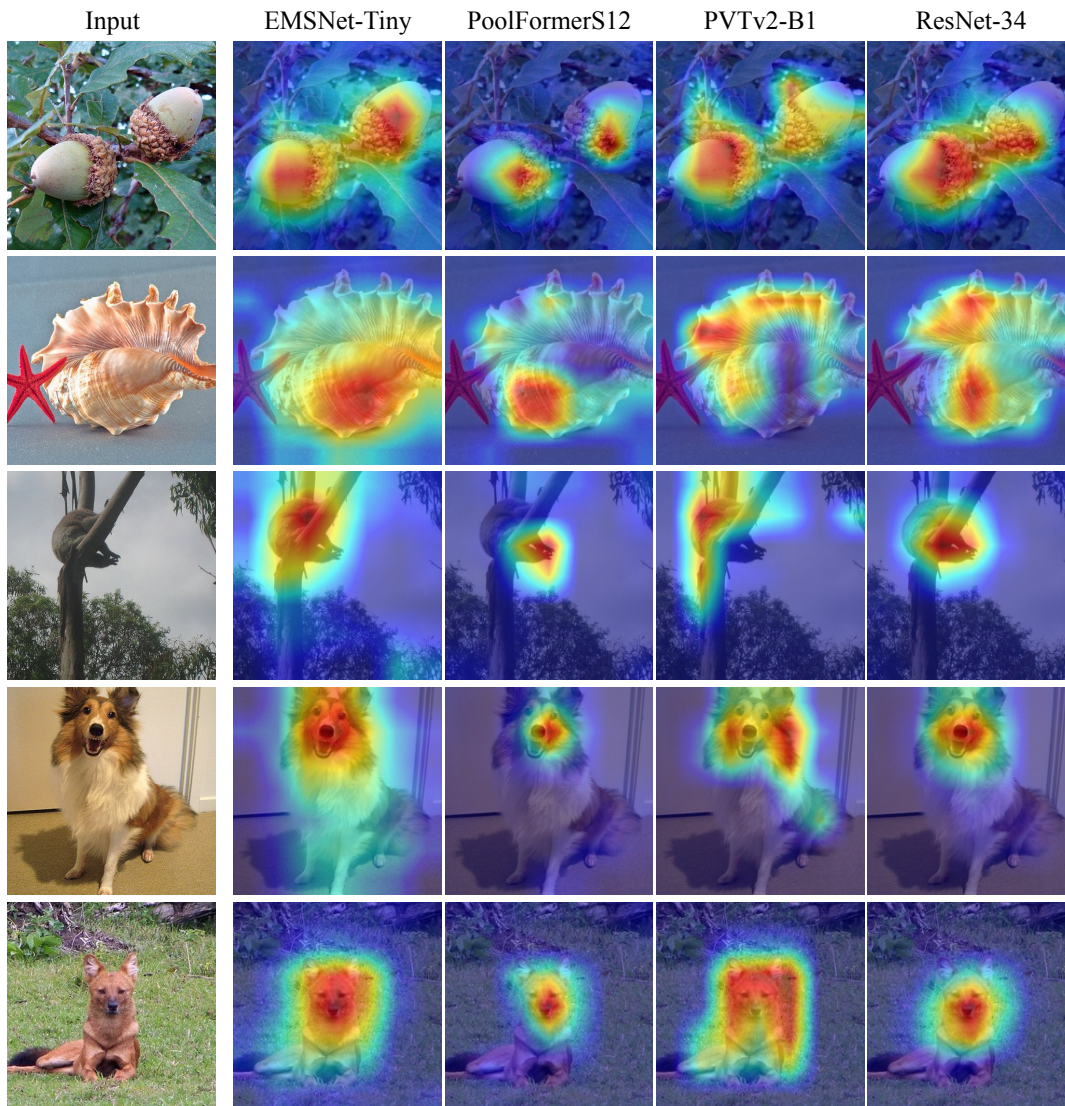


FIGURE 3.11: Grad-CAM activation map visualization of the EMSNet-Tiny, PoolFormerS12 (Yu et al., 2022), PVTv2-B1 (Wang et al., 2022b), and ResNet-34 (He et al., 2016). All the models are trained on ImageNet-1K.

## Chapter 4

# Exchanging Information across Non-overlapped Local Self-Attentions via Mixing Abstract Tokens

### 4.1 Introduction

Recently, Vision Transformers (ViT) (Dosovitskiy et al., 2021; Touvron et al., 2021) have become dominant methods in processing visual data, achieving new performances on image classification, downstream tasks, and foundation models. As a key component of the Transformer, the self-attention operation has high flexibility in modeling long-range dependencies and great generalization capability. This is due to that the interactions among all tokens are performed together via query-key matrix multiplication and input-dependent attention weights are adopted. However, global self-attention requires quadratic complexity with the token lengths and has weak inductive bias such as locality and relative positions between tokens. These issues open two main directions to finding solutions.

To deal with inductive bias, ViT needs large-scale datasets such as ImageNet-21K (Russakovsky et al., 2015), and JFT300M (Sun et al., 2017a), and strong data augmentations (Touvron et al., 2021) to train the models. Another solution is to incorporate locality of convolution into self-attention layers (Wang et al., 2021a; Wang et al., 2022b; Chen et al., 2022b; Wu et al., 2021; Zhang and Yang, 2022), combine



| Method                                | Non-Overlapped Window (window sizes) | Cross-Window Strategy         | Implementation          |
|---------------------------------------|--------------------------------------|-------------------------------|-------------------------|
| Swin Transformer (Liu et al., 2021b)  | $7 \times 7$                         | cyclic shift                  | torch.roll()            |
| HaloNet (Vaswani et al., 2021)        | $7 \times 7$                         | sliding                       | Unfold() & Padding()    |
| NAT (Hassani et al., 2023)            | $7 \times 7$                         | sliding                       | CUDA kernels            |
| Slide-Transformer (Pan et al., 2023)  | $3 \times 3, 7 \times 7$             | sliding                       | DWConv()                |
| CSWin Transformer (Dong et al., 2022) | $[7 \times H, W \times 7]$           | expanding                     | reshape() & concat()    |
| Pale Transformer (Wu et al., 2022)    | $[7 \times H, W \times 7]$           | expanding                     | slice() & concat()      |
| CrossFormer (Wang et al., 2022c)      | $7 \times 7$                         | shuffling                     | reshape()               |
| <b>MAT (Ours)</b>                     | $7 \times 7$                         | <b>mixing abstract tokens</b> | Matrix multiplication() |

TABLE 4.1: Comparison of local self-attentions. Our MAT is quite different from recent methods in both cross-window designs and implementation.  $H, W$  indicates the height and width of the feature map.  $[k \times H, W \times k]$  denotes the model computes self-attention inside cross-shaped windows in parallel. DWConv() is depthwise convolution with special initial weights.

strengths of convolution and self-attention (Si et al., 2022; Li et al., 2022a; Chen et al., 2022a; Yang et al., 2023; Hatamizadeh et al., 2023a; Zhang et al., 2023), or insert self-attention layers into existing Convolutional Neural Networks (CNNs) (Mehta and Rastegari, 2022; Mehta and Rastegari, 2023; Zhang et al., 2022).

For the problem of model cost, many methods try to reduce ViT complexity, making the model suitable for dense prediction tasks. PVT (Wang et al., 2021a), DAT (Xia et al., 2022) computes sparse global attention by down-sampling key and value tokens. Although sparse global attention improves efficiency, the locality is difficult to achieve, and helpful information in key and value features is ignored. Another line of the research is to limit attention areas to local windows. Swin Transformer (Liu et al., 2021b) partitions the input feature into non-overlapped windows and self-attention models interactions between tokens inside windows. With this scheme, Swin Transformer has linear complexity with the token length while limiting receptive fields and weakening modeling capability. Compared to global self-attention with weak inductive biases, local self-attention has merits of both convolution with translation-equivariance and local connection, and self-attention with data-dependent weights and modeling flexibility.

To enlarge receptive fields, typical methods require extra designs, e.g., window shifting (Liu et al., 2021b), expanding (Dong et al., 2022; Wu et al., 2022), sliding (Hassani et al., 2023; Vaswani et al., 2021; Pan et al., 2023), and shuffling (Huang et al., 2021; Tu et al., 2022), to communicate the information across windows. However, the growth of the receptive field is very slow, and the models stack a large number of successive blocks (non-overlapped local self-attention block + extra local self-attention block) to result in long-range dependencies. Moreover, window

shifting and sliding are implemented by inefficient operations e.g., `torch.roll()` and `Unfold()`, that require high memory access and additional latency. These operations are not well supported and optimized in modern deep-learning frameworks such as ONNX, NVIDIA TensorRT, Tensorflow-Lite, or Torchscript (Pan et al., 2022a; Chu et al., 2021b). NAT (Hassani et al., 2023) creates overlapped windows by writing CUDA kernels and difficult to deploy the models on devices without CUDA.

In this work, we propose a new alternative way to efficiently exchange information across local windows called Mixing Abstract Tokens (MAT). Specifically, the image feature is separated into non-overlapped windows and the learnable abstract tokens are used as the bridge between non-overlapped windows. To do that, firstly, each abstract token is attached to each window, and based on the local self-attention mechanism, the abstract token spatially interacts with all tokens inside each corresponding window. In other words, each abstract token learns abstract information from all window tokens and can represent each window. Secondly, all learned abstract tokens are mixed together via a Transformer encoder to explicitly communicate information across local windows and directly result in global receptive fields. Then, the global dependencies are propagated back to each local token via a Transformer decoder and can enrich the representation of each image token. Compared to extra designs of local self-attention methods, our architecture models both local and global features in an efficient and flexible way, and only involves matrix multiplication which is easy to implement and optimize. The communication process is conducted at a low complexity because the number of abstract tokens in MAT is largely fewer than image tokens.

## 4.2 Related Works

### 4.2.1 Vision Transformers.

After the success of the original Transformer (Vaswani et al., 2017) in the language field, the significant shift of the Transformer towards incorporating the vision, audio, and foundation models has achieved noteworthy attention. DETR (Carion et al., 2020) was the first method that successfully applies the Transformer encoder, and decoder to the object detection task and opens new views in modeling image

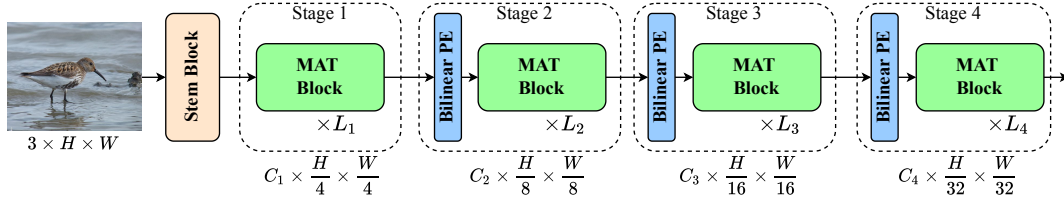


FIGURE 4.1: Overall architecture of the MAT. Our architecture consists of a Stem Block and four stages inspired by the hierarchical network. Successive convolutions in Stem Block are used to down-sample the input image by a factor of 4 and change 3 channels to  $C_1$ . In each stage, Bilinear PE (Patch Embedding) is introduced to select informative features of the patches based on input pixel locations and bilinear interpolation, and a stack of MAT Blocks is adopted.

and video data. In 2021s, ViT (Dosovitskiy et al., 2021; Touvron et al., 2021) fully adopts the Transformer encoder to image classification and achieves promising performances compared to CNNs counterparts. To process visual data with high dimensions, ViT splits images into a sequence of patches and considers one patch with size  $16 \times 16$  as one token. And using Transformer encoders learn interactions between tokens that produce global features. PVT (Wang et al., 2021a; Wang et al., 2022b) leverages the benefits of ViT into downstream tasks by designing hierarchical backbones and introducing spatial reduction attention (SRA) to reduce the complexity of self-attention. DAT (Xia et al., 2022) replaces SRA with deformable attention. To augment weak inductive biases of ViT, several methods (Chen et al., 2022b; Wu et al., 2021; Guo et al., 2022a; Wang et al., 2022b; Zhang and Yang, 2022) internally employ convolution into self-attention layers and achieve great improvements without pretraining on large-scale datasets. Based on the relative position of convolution, Twins (Chu et al., 2021b), CPVT (Chu et al., 2023), and CSWin Transformer (Dong et al., 2022) substitutes absolute positional encoding in ViT by convolution. Other research externally combines the merits of convolution and self-attentions to build hybrid models (Mehta and Rastegari, 2022; Mehta and Rastegari, 2023; Zhang et al., 2023; Li et al., 2023; Li et al., 2022a; Chen et al., 2022a; Pan et al., 2022a; Hatamizadeh et al., 2023a; Si et al., 2022).

#### 4.2.2 Local self-attentions.

Table 4.1 summarizes the key difference between our MAT and recent methods. HaloNet (Vaswani et al., 2021) separates the query feature into non-overlapped windows, and the key & value features into overlapped windows using sliding torch.roll()

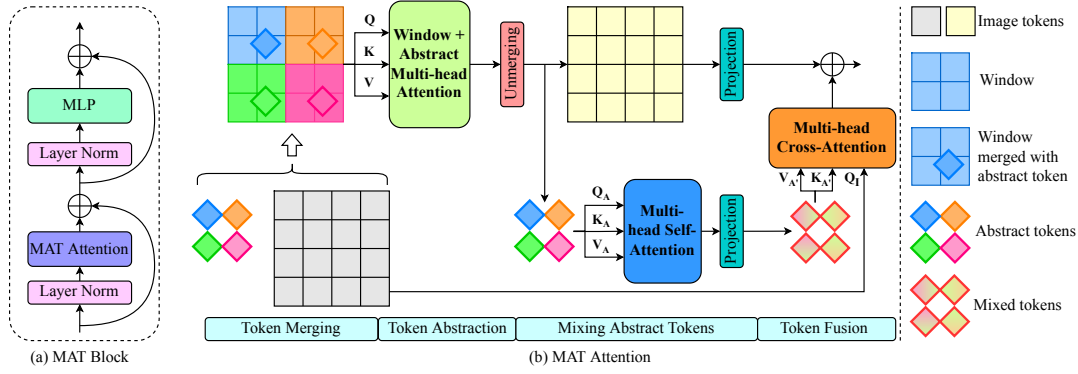


FIGURE 4.2: (a) Illustration of the MAT Block. (b) The detailed architecture of the MAT attention. The image tokens are partitioned into a set of windows and each learnable abstract token is merged with one corresponding window. Then, in each merged token, local-self attention learns spatial interactions of window tokens and the abstract token *versus* window tokens. Each abstract token learns abstract information from each window. Multi-head self-attention is used to mix learned abstract tokens, exchanging information across windows and resulting in global features. Finally, multi-head cross-attention using the image tokens as query and the mixed abstract tokens as key and value pairs propagate global information to the image features.

with zero-padding to slightly enlarge receptive fields. Slide-Transformer (Pan et al., 2023) initialize the weights of DWConv with special values to shift the features towards different locations. CSWin Transformer (Dong et al., 2022) and Pale Transformer (Wu et al., 2022) partition the input features into cross-shaped windows and apply self-attentions on these windows, resulting in larger receptive fields. CrossFormer (Wang et al., 2022c) replaces shifted window attentions in Swin Transformer with long-distance attention where the features are shuffled via `reshape()`. Instead of internally exchanging information across windows, MixFormer (Chen et al., 2022a) applies overlapped DWConv with window self-attention in the parallel scheme for externally modeling cross-window relations. MOAT (Yang et al., 2023), EMO (Zhang et al., 2023) implements non-overlapped local-self-attention and DWConv in sequential manners.

### 4.3 Methodology

The overall architecture of the proposed MAT is described in Figure 4.1. The MAT with the hierarchical pyramid structure is designed for an efficient and general backbone. Specifically, the MAT consists of one Stem Block and four stages with strides  $\{4, 8, 16, 32\}$ , and the number of channels across stages is expanded. Following

with existing methods (Wang et al., 2021a; Wang et al., 2022b; Chen et al., 2022a; Hatamizadeh et al., 2023a; Guo et al., 2022a; Si et al., 2022), Stem Block includes three successive convolution layers to down-sample the image by a factor of 4 and produce image tokens. Then, the image tokens are fed into the main MAT block. Performing local self-attention inside non-overlapped windows can reach linear complexity with the token lengths instead of full self-attention with expensive costs. However, stacking more local self-attention blocks yields limited receptive fields and hinders modeling capability. Therefore, efficiently exchanging information across windows is necessary to enlarge receptive fields and capture long-range dependencies. To meet these requirements, the MAT Block is proposed to perform cross-window attention through friendly implementation, shown in Figure 4.2(a). Additionally, a bilinear PE (Patch Embedding) between two stages is proposed to down-sample the number of image tokens in a data-dependent approach and increase the number of channels.

### 4.3.1 MAT Attention

The key design of this paper is MAT attention, which efficiently captures global receptive fields from the input tokens and has linear complexity with the token lengths. Figure 4.2(b) illustrates the structure of the MAT attention, consisting of four steps: *Token Merging*, *Token Abstraction*, *Mixing Abstract Tokens*, and *Token Fusion*. In the first step, the image token is partitioned into a set of non-overlapped windows, and each learnable abstract token is merged into one corresponding window. Then, in the second step, image tokens inside each window interact with others and with the abstract token via local self-attention, and the abstract token takes charge of abstracting information from each window. Mixing abstract tokens is performed in step three to exchange information across windows. In the last step, mixed features are delivered to the image tokens.

#### Token Merging.

First, we equally partition the input token  $\mathbf{X} \in \mathbb{R}^{C \times H \times W}$  (gray tokens in Figure 4.2(b)) into non-overlapped windows as  $\mathbf{X}^w \in \mathbb{R}^{N_w \times h \times w^2 \times C/h}$ , where  $H, W, C$  denotes height, width and channels of the image token;  $N_w = \frac{H}{w} * \frac{W}{w}$  is number of

window tokens;  $w$  indicates the window size; and  $h, C/h$  stand for number of heads and head dimension. Second, a set of learnable abstract tokens  $\mathbf{A} \in \mathbb{R}^{N_w \times h \times 1 \times C/h}$  (rhomb tokens in Figure 4.2(b)) is merged into the window token  $\mathbf{X}$  correspondingly and the merged token  $\mathbf{X}^A \in \mathbb{R}^{N_w \times h \times (w^2+1) \times C/h}$  is created.

### Token Abstraction.

After partitioning and merging, the multi-head self-attention (MHSA) operation is performed within each merged token  $\mathbf{x}_i^A \in \mathbb{R}^{h \times (w^2+1) \times C/h}$ . Obviously, each abstract token abstracts information among all tokens in each corresponding window via query-key matrix multiplication of self-attention. In other words, each abstract token can represent each window. Formally, each non-overlapped local self-attention (SA) is computed as follows:

$$\text{MHSA}(\mathbf{x}_i^A) = \text{Concat}(\text{SA}_1, \dots, \text{SA}_h), \quad (4.1)$$

$$\text{SA}_j(\mathbf{x}_{i,j}^A) = \text{softmax}\left(\frac{\mathbf{Q}\mathbf{K}^T}{\sqrt{C/h}}\right)\mathbf{V}, \quad (4.2)$$

where  $\mathbf{Q} = \mathbf{W}^Q \mathbf{x}_{i,j}^A$ ,  $\mathbf{K} = \mathbf{W}^K \mathbf{x}_{i,j}^A$ ,  $\mathbf{V} = \mathbf{W}^V \mathbf{x}_{i,j}^A \in \mathbb{R}^{(w^2+1) \times C/h}$  are query, key, and value matrices, respectively and  $\mathbf{W}^Q, \mathbf{W}^K, \mathbf{W}^V \in \mathbb{R}^{C/h \times C/h}$  are projection matrices shared across the number of merged tokens and heads.  $j \in [1, \dots, h]$  indicates the head index. Although non-overlapped local self-attention improves efficiency, its computation makes the receptive field weak and results in performance degradation. Therefore, communications across windows are supplemented to enlarge the modeling capability. An intuitive way is to mix abstract tokens because each abstract token attends to all image tokens within each window.

### Mixing Abstract Tokens.

After capturing interactions among image tokens and the abstract token *versus* image tokens inside each window, externally mixing abstract tokens can allow the model to exchange information across non-overlapped windows and result in global information. First, the interacted token  $\mathbf{X}^{A'}$  obtained by local self-attention is reversed and unmerged back to the image token  $\mathbf{X}' \in \mathbb{R}^{C \times H \times W}$  and the learned abstract token  $\mathbf{A}' \in \mathbb{R}^{h \times N_w \times C/h}$  via window reverse and slice operation. Second,

$\text{MHSA}(\mathbf{A}')$  is used to mix all abstract tokens globally. The  $\text{MHSA}(\cdot)$  is compatible with our design as number of abstract tokens  $N_w$  is changed according to the image size. Technically, the mixed abstract token  $\tilde{\mathbf{A}}'$  is computed as:

$$\text{MHSA}(\mathbf{A}') = \text{Concat}(\text{SA}_1, \dots, \text{SA}_h), \quad (4.3)$$

$$\text{SA}_j(\mathbf{A}'_j) = \text{softmax}\left(\frac{\mathbf{Q}_A \mathbf{K}_A^T}{\sqrt{C/h}}\right) \mathbf{V}_A, \quad (4.4)$$

where  $\mathbf{Q}_A = \mathbf{W}_A^Q \mathbf{A}'_j$ ,  $\mathbf{K}_A = \mathbf{W}_A^K \mathbf{A}'_j$ ,  $\mathbf{V}_A = \mathbf{W}_A^V \mathbf{A}'_j \in \mathbb{R}^{N_w \times C/h}$  are query, key, and value matrices. Through  $\text{MHSA}(\cdot)$ , exchanging information across tokens is obtained and produces the global feature  $\tilde{\mathbf{A}}'$  (mixed tokens in Figure 4.2(b)).

### Token Fusion.

After mixing abstract tokens, the global feature  $\tilde{\mathbf{A}}'$  is propagated back to the image feature  $\mathbf{X}$  through the Transformer decoder to enhance the modeling capability of local self-attention. The mixed token acts as the key and value feature and the query matrix is computed from the image token. For one head, the cross-attention (CA) is addressed as:

$$\text{CA}(\mathbf{X}, \tilde{\mathbf{A}}') = \text{softmax}\left(\frac{\mathbf{Q}_I \mathbf{K}_{A'}^T}{\sqrt{C/h}}\right) \mathbf{V}_{A'}, \quad (4.5)$$

where  $\mathbf{Q}_I = \mathbf{W}^Q \mathbf{X}$ ,  $\mathbf{K}_{A'} = \mathbf{W}^K \tilde{\mathbf{A}}'$ , and  $\mathbf{V}_{A'} = \mathbf{W}^V \tilde{\mathbf{A}}'$  are image query, abstract key, and abstract value matrices.

Finally, the final output feature is generated via element-wise addition between the output of local self-attention  $\mathbf{X}'$  and the enhanced feature in the token fusion process.

### 4.3.2 Bilinear PE

The goal of patch embeddings is to divide the image token into a sequence of patches and build pyramid networks. Swin Transformer (Liu et al., 2021b; Liu et al., 2022a) uses patch merging layers that merge patches in a uniform way and lead to an inefficient solution to keep important features. Similar to patch merging, recent methods (Wang et al., 2021a; Wang et al., 2022b; Guo et al., 2022a; Wang et al., 2022c; Pan et al.,

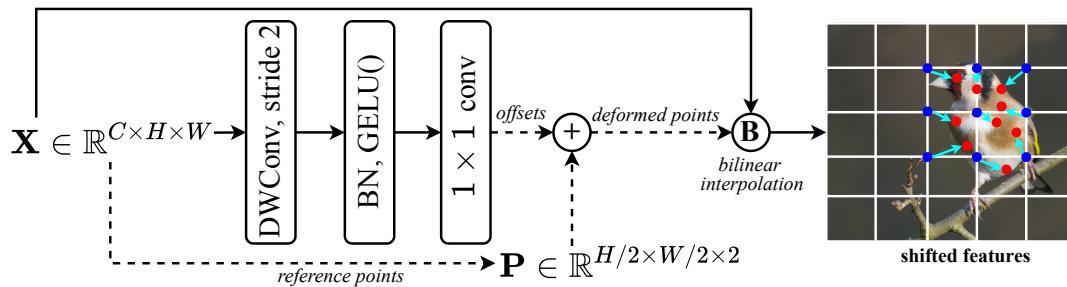


FIGURE 4.3: Bilinear PE. A grid of reference points  $\mathbf{P}$  is taken from the input feature. The offsets are learned by the DWConv layer followed by  $1 \times 1$  convolution, and conditioned on the input image. 9 reference points are shown in blue color as a simple example and sampled points are denoted by red points. Dash and solid lines describe the position and feature process, respectively.

2022a) integrate inductive biases of convolution into patch embedding. However, all patches have equal contributions to model output. Several methods (Pan et al., 2022c; Pan, Cai, and Zhuang, 2022) apply Deformable convolution (Zhu et al., 2019) for adaptively selecting informative regions based on pixel locations. Although Deformable convolution with data-dependent weights and adaptive aggregation can complement vanilla convolution, its implementation relies on CUDA kernels and is not well supported on mobile devices.

To achieve friendly deployment, this paper employs  $\text{DWConv}(\cdot)$  that learns grids of pixel locations and bilinear interpolation to sample relevant regions of the input feature based on learned offsets. Figure 4.3 illustrates the detailed architecture of the Bilinear PE.

### 4.3.3 Model Configuration

Inspired by investigations of (Si et al., 2022; Zhang and Yang, 2022; Park and Kim, 2022; Pan et al., 2022c; Pan, Cai, and Zhuang, 2022), modern CNNs and vision Transformers have manifested that earlier layers tend to learn low-level information (local features), and later layers capture high-level information (global features). Putting global self-attention layers in earlier stages only brings minor improvements and can suffer high complexity since GFLOPs of ViT-based models are sensitive to the token lengths. Moreover, our design can model long-range dependencies from the input token in one block. Hence, the MAT blocks are inserted in later stages to reduce the model cost. As shown in Table 4.2, using the MAT blocks in stages 3 and 4



gets a better trade-off between Top-1 accuracy and cost.

| Model   | Stage |   |   |   | Top-1 (%) | GFLOPs | #param (M) |
|---------|-------|---|---|---|-----------|--------|------------|
|         | 1     | 2 | 3 | 4 |           |        |            |
| Model 1 | ✓     | ✓ | ✓ | ✓ | 78.9      | 0.783  | 10.874     |
| Model 2 | ✗     | ✓ | ✓ | ✓ | 78.8      | 0.707  | 10.852     |
| Model 3 | ✗     | ✗ | ✓ | ✓ | 79.0      | 0.666  | 10.767     |
| Model 4 | ✗     | ✗ | ✗ | ✓ | 78.5      | 0.568  | 9.767      |

TABLE 4.2: Positions of the MAT blocks. ✓ indicates the MAT blocks are used in this stage and ✗ denotes there is no spatial token mixer in this stage.

Based on the above results, we manually stack a number of MAT blocks in stages 3 and 4, and variants of the MAT model are shown in Table 4.3. In all models, the expansion ratio of the pure MLP is configured to a value of 4 and kept across stages.

| Variant | #dim | #blocks    | #heads | GFLOPs | #params |
|---------|------|------------|--------|--------|---------|
| MAT-1   | 24   | 2, 2, 6, 6 | 12, 24 | 0.389  | 6.714   |
| MAT-2   | 32   | 2, 2, 6, 6 | 8, 16  | 0.666  | 10.767  |
| MAT-3   | 36   | 2, 2, 8, 8 | 8, 16  | 1.042  | 17.008  |
| MAT-4   | 48   | 3, 3, 8, 8 | 12, 24 | 1.933  | 29.057  |
| MAT-5   | 64   | 2, 2, 8, 8 | 16, 32 | 3.156  | 50.108  |

TABLE 4.3: Detailed configurations of 5 MAT models. #dim is the number of base channels and this value is duplicated in the next stage. #blocks is the number of stacked MAT blocks. #heads is the number of heads in self-attention and cross-attention layers.

## 4.4 Experiments

We conduct extensive experiments on ImageNet-1K (Russakovsky et al., 2015) image classification, MS COCO (Lin et al., 2014) object detection and instance segmentation, and ADE20k (Zhou et al., 2019) semantic segmentation. Visualizations of the learned and mixed abstract tokens are validated to show the contribution of our design to feature learning. Ablation studies are also performed to verify the effectiveness of each component in the MAT block.

| Method                                  | Size | P(M) | G   | Top-1 |
|---|------|------|-----|-------|
| MViTv1-XXS (Mehta and Rastegari, 2022)  | 256  | 1.3  | 0.4 | 69.0  |
| MViTv2-0.5 (Mehta and Rastegari, 2023)  | 256  | 1.4  | 0.5 | 70.2  |
| EMO-1M (Zhang et al., 2023)             | 224  | 1.3  | 0.3 | 71.5  |
| EfficientViT-M4 (Liu et al., 2023)      | 224  | 8.8  | 0.3 | 74.3  |
| EfficientViT-M5 (Liu et al., 2023)      | 224  | 12.4 | 0.5 | 77.1  |
| PVTv2-B0 (Wang et al., 2022b)           | 224  | 3.7  | 0.6 | 70.5  |
| Swin-0.7G (Liu et al., 2021b)           | 224  | 4.4  | 0.7 | 74.4  |
| DFvT-T (Gao et al., 2022)               | 224  | 4.0  | 0.3 | 73.0  |
| MViTv1-XS (Mehta and Rastegari, 2022)   | 256  | 2.3  | 1.0 | 74.8  |
| MViTv2-0.75 (Mehta and Rastegari, 2023) | 256  | 2.9  | 1.0 | 75.6  |
| EdgeViT-XXS (Pan et al., 2022a)         | 256  | 4.1  | 0.6 | 74.4  |
| tiny-MOAT-0 (Yang et al., 2023)         | 224  | 3.4  | 0.8 | 75.5  |
| EMO-2M (Zhang et al., 2023)             | 224  | 2.3  | 0.4 | 75.1  |
| ConvNext-Xt (Liu et al., 2022b)         | 224  | 7.4  | 0.6 | 77.5  |
| MobileFormer-294M (Chen et al., 2022b)  | 224  | 11.4 | 0.6 | 77.9  |
| VAN-B0 (Guo et al., 2022b)              | 224  | 4.1  | 0.9 | 75.4  |
| LVT (Pan et al., 2022c)                 | 224  | 5.5  | 0.9 | 74.8  |
| Swin-1G (Liu et al., 2021b)             | 224  | 7.3  | 1.0 | 77.3  |
| EMO-5M (Zhang et al., 2023)             | 224  | 5.1  | 0.9 | 78.4  |
| EMO-6M (Zhang et al., 2023)             | 224  | 6.1  | 1.0 | 79.0  |
| DFvT-S (Gao et al., 2022)               | 224  | 11.2 | 0.8 | 78.3  |
| PVT-T (Wang et al., 2021a)              | 224  | 13.2 | 1.6 | 75.1  |
| tiny-MOAT-1 (Yang et al., 2023)         | 224  | 5.1  | 1.2 | 78.3  |
| ResT-Lite (Zhang and Yang, 2021)        | 224  | 10.5 | 1.4 | 77.2  |
| ResT-Small (Zhang and Yang, 2022)       | 224  | 13.7 | 1.9 | 79.6  |
| EdgeViT-XS (Pan et al., 2022a)          | 256  | 6.7  | 1.1 | 77.5  |
| MViTv1-S (Mehta and Rastegari, 2022)    | 256  | 5.6  | 2.0 | 78.4  |
| MViTv2-1.0 (Mehta and Rastegari, 2023)  | 256  | 4.9  | 1.9 | 78.1  |
| PoolFormer-S12 (Yu et al., 2022)        | 224  | 11.9 | 1.8 | 77.2  |
| Slide-PVT-T (Pan et al., 2023)          | 224  | 12.2 | 2.0 | 78.0  |
| PVTv2-B1 (Wang et al., 2022b)           | 224  | 13.1 | 2.1 | 78.7  |
| Slide-PVTv2-B1 (Pan et al., 2023)       | 224  | 13.0 | 2.2 | 79.5  |
| Swin-2G (Liu et al., 2021b)             | 224  | 12.8 | 2.0 | 79.2  |
| PoolFormer-S24 (Yu et al., 2022)        | 224  | 21.3 | 3.4 | 80.3  |
| ParC-Net-S (Zhang, Hu, and Wang, 2022)  | 256  | 5.0  | 3.5 | 78.6  |
| PVT-S (Wang et al., 2021a)              | 224  | 24.5 | 3.8 | 79.8  |
| ResT-Base (Zhang and Yang, 2021)        | 224  | 30.3 | 4.3 | 81.6  |
| LITv1-Ti (Pan et al., 2022c)            | 224  | 19.0 | 3.6 | 81.1  |
| LITv1-S (Pan et al., 2022c)             | 224  | 27.0 | 4.1 | 81.5  |
| LITv2-S (Pan, Cai, and Zhuang, 2022)    | 224  | 28.0 | 3.7 | 82.0  |
| ConvNeXt-T (Liu et al., 2022b)          | 224  | 28.0 | 4.5 | 82.1  |
| Swin-T (Liu et al., 2021b)              | 224  | 28.3 | 4.5 | 81.3  |
| MAT-1 (Ours)                            | 224  | 6.7  | 0.4 | 76.3  |
| MAT-2 (Ours)                            | 224  | 10.8 | 0.7 | 79.0  |
| MAT-3 (Ours)                            | 224  | 17.0 | 1.0 | 80.2  |
| MAT-4 (Ours)                            | 224  | 29.1 | 1.9 | 81.0  |
| MAT-5 (Ours)                            | 224  | 50.1 | 3.2 | 81.9  |

TABLE 4.4: Comparison of the MAT variants and recent methods on ImageNet-1K validation set. P denotes the number of parameters and G indicates GFLOPs.

### 4.4.1 Image Classification

#### Settings.

We train and evaluate MAT variants on ImageNet-1K benchmark (Russakovsky et al., 2015). This dataset includes 1.2M training and 50k validation images with 1K classes. For fair comparisons, we follow the training receipt of previous methods (Wang et al., 2021a; Liu et al., 2021b; Touvron et al., 2021). In detail, 2 A100 GPUs are used to train all models for 300 epochs from scratch. The optimizer is AdamW (Loshchilov and Hutter, 2019) with an initial learning rate of 0.001 and weight decay of 0.05. The images are resized to  $224 \times 224$  and the total batch size is 4096. Common data augmentations (Touvron et al., 2021) are adopted such as RandAugment, Mixup, CutMix, and stochastic depth, and we do not use EMA (Polyak and Juditsky, 1992) to improve performance. All the models are implemented by Pytorch framework and the codebase (Wightman, 2019).

#### Results.

Table 4.4 shows the main performance of the MAT and the comparison with recent methods on the ImageNet-1K validation set. For small models, our MAT surpasses efficient methods by clear margins. For example, MAT-2 achieves 79.0% Top-1 accuracy that outperforms PVTv2-B0 (Wang et al., 2022b) by 8.5%, MobileViTv2-0.75 (Mehta and Rastegari, 2023) by 3.4% with 70% GFLOPs, and a concurrent work EMO-5M (Zhang et al., 2023) by 0.6% with 77.8% GFLOPs. For medium size, MAT-3 gets 80.2% Top-1 accuracy with only 1.0 GFLOPs. It is better than the current method Slide-PVTv2-B1 (Pan et al., 2023) by 0.7% while saving 54.5% GFLOPs. For base models, MAT-5 achieves similar results to previous methods under smaller GFLOPs. Compared with the baseline Swin Transformer (Liu et al., 2021b), our MAT achieves better performance with lower GFLOPs. In detail, MAT-3 surpasses Swin-2G by 1.0% with only 50% GFLOPs.

*CPU Latency.* We also report accuracy-latency comparisons among representative methods in Figure 4.4. Swin0.7G-T indicates four variants of Swin Transformer (Liu et al., 2021b): Swin-0.7G (GFLOPs), Swin-1G, Swin-2G, and Swin-T (Tiny). As a result, MAT gets superior trade-offs between accuracy and CPU latency on ImageNet-1K

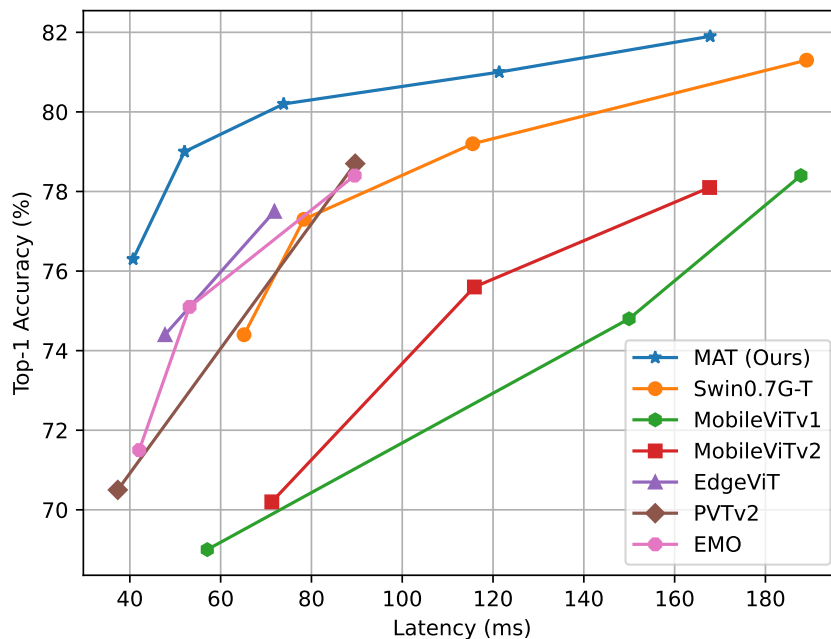


FIGURE 4.4: Accuracy-latency comparison between our MAT and representative networks. MAT achieves better trade-offs between Top-1 accuracy and efficiency. Latency (ms) is measured on the CPU Intel(R) Xeon(R) Gold 5220R @2.20GHz.

image classification. Interestingly, MAT runs faster than efficient models such as MobileViTv1, MobileViTv2, EdgeViT, and EMO. Compared to Swin Transformer, our MAT achieves significant improvements in both accuracy and efficiency. These results verify the advantages of our MAT in terms of cross-window design.

*Visualizations of attention maps.* Figure 4.5 illustrates the role of abstract tokens in token abstraction and token fusion steps across blocks. We can observe that abstract tokens tend to learn object boundaries in earlier blocks and mixing learned abstract tokens results in the larger focused regions. In later blocks, abstract tokens capture the key parts of objects, and through mixing and fusing abstract tokens globally, the image features are enhanced and focused on target regions.

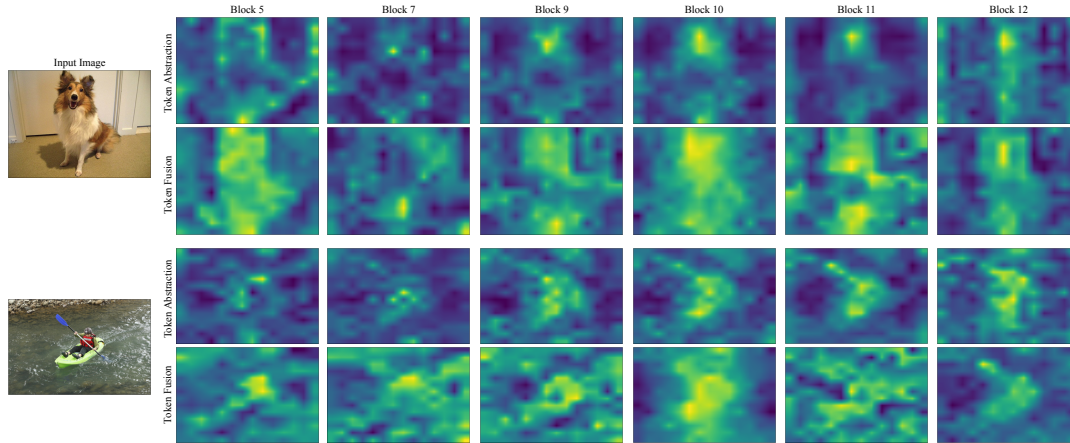


FIGURE 4.5: Visualization of attention maps of the MAT model across blocks. In each input image, the first row shows where each abstraction token attends, and the second row illustrates the enhanced information of the image token via mixing abstract tokens globally and its aggregation.

#### 4.4.2 Downstream tasks

##### Object detection & instance segmentation.

We verify the effectiveness of MAT on object detection with detector SSD (Liu et al., 2016), RetinaNet (Lin et al., 2017b), and Mask R-CNN (He et al., 2017b). All the models are trained and evaluated on MS-COCO dataset (Lin et al., 2014) with 118K training and 5K validation images. Following common settings (Wang et al., 2021a; Liu et al., 2021b), the models are fine-tuned by AdamW optimizer with an initial learning rate of  $1 \times 10^{-4}$  and a weight decay of 0.05. The input images are resized to  $320 \times 320$  for SSD detector, and  $1333 \times 800$  for RetinaNet, Mask R-CNN. The learning schedule of the RetinaNet and Mask R-CNN is configured to  $1 \times \text{schedule}$  (12 epochs). Table 4.5, 4.6, and 4.7 shows comparisons of various backbones on detector SSD (a), RetinaNet (b), and Mask R-CNN (c). Under similar GFLOPs, the MAT models surpass other backbones. Specifically, for RetinaNet, the MAT-4 achieves 41.9  $AP^{bb}$  higher than Swin-T by 0.4% with only 76% GFLOPs, and original RetinaNet by 5.6% with also 76% GFLOPs. For the Mask R-CNN, our models still maintain efficiency while achieving better  $AP^{bb}$  and  $AP^m$ .

| Backbone                                | P(M) | G   | $AP^{bb}$ |
|---|------|-----|-----------|
| MViTv1-XXS (Mehta and Rastegari, 2022)  | 1.7  | 0.9 | 19.9      |
| MViTv2-0.5 (Mehta and Rastegari, 2023)  | 2.0  | 0.9 | 21.2      |
| MNetv2 (Sandler et al., 2018)           | 4.3  | 0.8 | 22.1      |
| EMO-1M (Zhang et al., 2023)             | 2.3  | 0.6 | 22.0      |
| EMO-2M (Zhang et al., 2023)             | 3.3  | 0.9 | 25.2      |
| MViTv2-0.75 (Mehta and Rastegari, 2023) | 3.6  | 1.8 | 24.6      |
| MViTv1-S (Mehta and Rastegari, 2022)    | 5.7  | 3.4 | 27.7      |
| MViTv2-1.25 (Mehta and Rastegari, 2023) | 8.2  | 4.7 | 27.8      |
| EMO-5M (Zhang et al., 2023)             | 6.0  | 1.8 | 27.9      |
| MViTv2-1.75 (Mehta and Rastegari, 2023) | 14.9 | 9.0 | 29.5      |
| ResNet-50 (He et al., 2016)             | 26.6 | 8.8 | 25.2      |
| MAT-1 (Ours)                            | 6.4  | 0.8 | 23.3      |
| MAT-2 (Ours)                            | 10.5 | 1.5 | 26.3      |
| MAT-3 (Ours)                            | 16.7 | 2.3 | 28.2      |

TABLE 4.5: Object detection performances with detector SSD (Liu et al., 2016)

| Backbone                         | P(M) | G   | $AP^{bb}$ |
|----------------------------------|------|-----|-----------|
| ResNet-18 (He et al., 2016)      | 21   | 189 | 31.8      |
| ResNet-50 (He et al., 2016)      | 38   | 250 | 36.3      |
| PVT-T (Wang et al., 2021a)       | 23   | 183 | 36.7      |
| PVTv2-B0 (Wang et al., 2022b)    | 13   | 160 | 37.1      |
| PoolFormer-S12 (Yu et al., 2022) | 22   | 207 | 36.2      |
| EMO-2M (Zhang et al., 2023)      | 12   | 167 | 36.2      |
| EMO-5M (Zhang et al., 2023)      | 15   | 178 | 38.9      |
| PVT-S (Wang et al., 2021a)       | 34   | 273 | 40.4      |
| LIT-S (Pan et al., 2022c)        | 39   | 305 | 41.6      |
| Swin-T (Liu et al., 2021b)       | 38   | 251 | 41.5      |
| MAT-2 (Ours)                     | 18   | 164 | 38.1      |
| MAT-3 (Ours)                     | 25   | 172 | 39.6      |
| MAT-4 (Ours)                     | 37   | 191 | 41.9      |
| MAT-5 (Ours)                     | 58   | 217 | 42.8      |

TABLE 4.6: Object detection performances with detector RetinaNet (Lin et al., 2017b) (1×schedule)

### Semantic Segmentation.

The experiments are conducted and evaluated on the benchmark ADE20K with 20K training and 2K validation images. For fair comparisons, we employ the receipts of (Wang et al., 2021a; Liu et al., 2021b) to train and evaluate the models. Table 4.8 reports the mIoU results on ADE20K dataset. Interestingly, our MAT block is quite suitable for semantic segmentation since improvement is better than the gain in detection. For example, MAT-4 gets 43.3% mIoU which outperforms Swin-T by 1.8% with only 70% GFLOPs, PVT-L by 1.2% with only half of GFLOPs, and original

| Backbone                      | P(M) | G   | $AP^{bb}$ | $AP^m$ |
|-------------------------------|------|-----|-----------|--------|
| ResNet-18 (He et al., 2016)   | 31   | 207 | 34.0      | 31.2   |
| ResNet-50 (He et al., 2016)   | 44   | 260 | 38.0      | 34.4   |
| ResNet-101 (He et al., 2016)  | 63   | 336 | 40.4      | 36.4   |
| PVTv2-B0 (Wang et al., 2022b) | 23   | 196 | 38.2      | 36.2   |
| PVT-T (Wang et al., 2021a)    | 33   | 208 | 36.7      | 35.1   |
| PVT-S (Wang et al., 2022b)    | 44   | 245 | 40.4      | 37.8   |
| PVT-M (Wang et al., 2021a)    | 64   | 302 | 42.0      | 39.0   |
| PVT-L (Wang et al., 2021a)    | 81   | 364 | 42.9      | 39.5   |
| LIT-S (Pan et al., 2022c)     | 48   | 324 | 42.0      | 39.1   |
| Swin-T (Liu et al., 2021b)    | 48   | 264 | 42.2      | 39.1   |
| MAT-2 (Ours)                  | 29   | 182 | 39.4      | 36.7   |
| MAT-3 (Ours)                  | 35   | 190 | 41.2      | 38.1   |
| MAT-4 (Ours)                  | 47   | 209 | 43.2      | 39.6   |
| MAT-5 (Ours)                  | 68   | 235 | 43.8      | 40.0   |

TABLE 4.7: Object detection and instance segmentation performances with detector Mask R-CNN (He et al., 2017b) ( $1\times$ schedule) on MS-COCO (Lin et al., 2014) validation set. All the backbones are pre-trained on ImageNet-1K (Russakovsky et al., 2015).  $AP^{bb}$  and  $AP^m$  indicate bounding box  $AP$ , and mask  $AP$ .

Semantic FPN (ResNet-50) by 6.6% with only 69% GFLOPs. These results verify the proposed MAT has a high potential for improving dense prediction tasks with high-resolution inputs.

| Backbone                     | #params (M) | GFLOPs | mIoU (%) |
|------------------------------|-------------|--------|----------|
| ResNet-50 (He et al., 2016)  | 29          | 183    | 36.7     |
| ResNet-101 (He et al., 2016) | 48          | 260    | 38.8     |
| PVT-S (Wang et al., 2021a)   | 28          | 161    | 39.8     |
| PVT-M (Wang et al., 2021a)   | 48          | 219    | 41.6     |
| PVT-L (Wang et al., 2021a)   | 65          | 283    | 42.1     |
| Swin-T (Liu et al., 2021b)   | 32          | 182    | 41.5     |
| MAT-2 (Ours)                 | 13          | 98     | 40.0     |
| MAT-3 (Ours)                 | 19          | 107    | 41.9     |
| MAT-4 (Ours)                 | 31          | 127    | 43.3     |
| MAT-5 (Ours)                 | 52          | 154    | 44.1     |

TABLE 4.8: Semantic Segmentation performances with Semantic FPN (Kirillov et al., 2019) on ADE20K (Zhou et al., 2019) val set. All the models are trained for 80K iterations.

### 4.4.3 Ablation Study

Our aim is to augment the modeling capability of non-overlapped local self-attention (Window Attn) in Swin Transformer with global MAT attention for enabling better

| Modules      |               | Top-1 | #params | GFLOPs | Throughput |
|--------------|---------------|-------|---------|--------|------------|
| Pure MLP     |               | 58.4  | 5.699   | 0.461  | 10303      |
| Token Mixers | +Window Attn  | 76.9  | 7.802   | 0.659  | 4353       |
|              | +MAT Attn     | 79.0  | 10.767  | 0.666  | 4333       |
| Patch Embed  | 3×3 conv      | 78.7  | 11.107  | 0.703  | 4432       |
|              | Patch Merging | 78.5  | 10.892  | 0.679  | 4944       |
|              | Bilinear PE   | 79.0  | 10.767  | 0.666  | 4333       |

TABLE 4.9: Ablation study of the MAT attention on ImageNet-1K. Throughput is measured on one GPU Tesla V100.

feature learning. Table 4.9 shows the effect of each component. As a result, the proposed MAT Attn achieves significant improvements (79.0% vs. 76.9%) over Window Attn, while still keeping the efficiency of Window Attn (similar throughput). It verifies the effectiveness of our attention design. We also replace the proposed Bilinear PE with other common designs such as 3×3 conv, and Patch Merging (Liu et al., 2021b). Using Bilinear PE for patch embedding layers achieves better accuracy and similar computational costs with existing methods.

## 4.5 Conclusion

This paper proposes MAT Transformer, an efficient and flexible backbone for enhancing receptive fields and modeling capability of non-overlapped local self-attentions. Thanks to the interaction of abstract tokens with window tokens via self-attention, aggregated information from window tokens is obtained. Mixing learned abstract tokens can allow the model to exchange information across windows efficiently and result in global feature representation. Finally, the mixed features are delivered back to the image token. Extensive experiments show that MAT Transformer surpasses previous methods on ImageNet image classification, MS-COCO object detection and instance segmentation, and ADE20K semantic segmentation.

## 4.6 Appendices

### 4.6.1 Limitations & Future Works

Due to computational resource constraints, we have not scaled the MAT design to larger models (more than 4 GFLOPs) and trained MAT on larger-scale datasets



(ImageNet-21K and JFT300M). Hence, comparing with efficient networks is the main intention of this paper. Moreover, MAT Transformer is a general and efficient backbone because our design performs well under various settings (from 0.4 GFLOPs to 3.2 GFLOPs) and is fine-tuned well on downstream tasks.

One obvious limitation of the proposed MAT Transformers is that our model has high parameters originating from query, key, and value projection matrices. However, in this paper, we verified that the models with higher parameters still run faster than the methods with smaller parameters. For instance, MAT-5 has 50.1M parameters (1.8 times higher than Swin-T with 28.3M parameters) while MAT-5 outperforms Swin-T (Liu et al., 2021b) in both accuracy and efficiency. It is similar to the original ViT (Dosovitskiy et al., 2021) in that the models with higher parameters still achieve better speeds. More efforts are needed to optimize the parameters of Transformer models. We leave this for future work.

## 4.6.2 Training Receipts

### Image Classification

All the models are trained for 300 epochs and on ImageNet-1K (Russakovsky et al., 2015) without pre-training on larger-scale datasets. The input images are resized to  $224 \times 224$ . Training receipts in (Touvron et al., 2021; Wang et al., 2021a; Liu et al., 2021b) have been widely used in modern networks. Therefore, for comparisons, we employ these settings to train all the models. Detailed hyper-parameters are shown in Table 4.10. Note that we do not use the EMA method to improve performances.

### Object detection & Instance segmentation

We transfer the MAT Transformers to MS-COCO (Lin et al., 2014) object detection and instance segmentation using three classical detectors: SSD (Liu et al., 2016), RetinaNet (Lin et al., 2017b), and Mask R-CNN (He et al., 2017a). The codebase is mmdetection (Chen et al., 2019b) and the models are trained on four NVIDIA 2080Ti GPUs. For training SSD model, we follow the hyperparameters in (Sandler et al., 2018; Mehta and Rastegari, 2022; Mehta and Rastegari, 2023). The input images are resized to  $320 \times 320$  and the total batch size is 192. For training RetinaNet (Lin et

| Configuration          | Value            |
|------------------------|------------------|
| Image size             | 224 <sup>2</sup> |
| Batch size             | 4096             |
| Epochs                 | 300              |
| Optimizer              | AdamW            |
| AdamW momentum         | (0.9, 0.999)     |
| Learning rate          | 0.001            |
| Learning rate schedule | cosine decay     |
| Weight decay           | 0.05             |
| Minimum learning rate  | 1e <sup>-6</sup> |
| Warmup epochs          | 5                |
| Warmup learning rate   | 1e <sup>-7</sup> |
| Mixup                  | 0.1              |
| Cutmix                 | 1.0              |
| Random erasing         | 0.25             |
| Drop path              | 0.05             |
| Color jitter           | 0.4              |
| Rand Augment           | (9, 0.5)         |
| Label smoothing        | 0.1              |
| EMA decay              | not used         |

TABLE 4.10: Detailed training settings for image classification MAT models.

| Configuration          | Value         |
|------------------------|---------------|
| Optimizer              | AdamW         |
| AdamW momentum         | (0.9, 0.999)  |
| Learning rate          | 0.0001        |
| Learning rate schedule | steps:[8, 11] |
| Weight decay           | 0.05          |
| Warmup iterations      | 500 (0.001)   |
| Drop path              | 0.05          |

TABLE 4.11: Detailed training settings for object detection and instance segmentation.

| Stage   | Out             | Layer Name  | MAT-1   | MAT-2  | MAT-3  | MAT-4  | MAT-5  |
|---------|-----------------|-------------|---|--|--|--|--|
| Stem    | 56 <sup>2</sup> | Patch Embed | 3×3 conv, stride 2, 12<br>3×3 DWconv, stride 1, 12<br>3×3 conv, stride 2, 24  | 3×3 conv, stride 2, 16<br>3×3 DWconv, stride 1, 16<br>3×3 conv, stride 2, 32   | 3×3 conv, stride 2, 18<br>3×3 DWconv, stride 1, 18<br>3×3 conv, stride 2, 36   | 3×3 conv, stride 2, 24<br>3×3 DWconv, stride 1, 24<br>3×3 conv, stride 2, 48   | 3×3 conv, stride 2, 32<br>3×3 DWconv, stride 1, 32<br>3×3 conv, stride 2, 64   |
| Stage 1 | 56 <sup>2</sup> | Pure MLP    | [MLP, exp=4] × 2  | [MLP, exp=4] × 2   | [MLP, exp=4] × 2   | [MLP, exp=4] × 3   | [MLP, exp=4] × 2   |
| Stage 2 | 28 <sup>2</sup> | Bilinear PE | 3×3 DWconv, stride 2, 24<br>1×1 conv, stride 1, 48<br>bilinear interpolation  | 3×3 DWconv, stride 2, 32<br>1×1 conv, stride 1, 64<br>bilinear interpolation   | 3×3 DWconv, stride 2, 36<br>1×1 conv, stride 1, 72<br>bilinear interpolation   | 3×3 DWconv, stride 2, 48<br>1×1 conv, stride 1, 96<br>bilinear interpolation   | 3×3 DWconv, stride 2, 64<br>1×1 conv, stride 1, 128<br>bilinear interpolation  |
|         |                 | Pure MLP    | [MLP, exp=4] × 2  | [MLP, exp=4] × 2   | [MLP, exp=4] × 2   | [MLP, exp=4] × 3   | [MLP, exp=4] × 2   |
| Stage 3 | 14 <sup>2</sup> | Bilinear PE | 3×3 DWconv, stride 2, 48<br>1×1 conv, stride 1, 96<br>bilinear interpolation  | 3×3 DWconv, stride 2, 64<br>1×1 conv, stride 1, 128<br>bilinear interpolation  | 3×3 DWconv, stride 2, 72<br>1×1 conv, stride 1, 144<br>bilinear interpolation  | 3×3 DWconv, stride 2, 96<br>1×1 conv, stride 1, 192<br>bilinear interpolation  | 3×3 DWconv, stride 2, 128<br>1×1 conv, stride 1, 256<br>bilinear interpolation |
|         |                 | Transformer | MATAttn $h=12$<br>MLP $exp=4$ × 6   | MATAttn $h=8$<br>MLP $exp=4$ × 6   | MATAttn $h=8$<br>MLP $exp=4$ × 8   | MATAttn $h=12$<br>MLP $exp=4$ × 8  | MATAttn $h=16$<br>MLP $exp=4$ × 8  |
| Stage 4 | 7 <sup>2</sup>  | Bilinear PE | 3×3 DWconv, stride 2, 96<br>1×1 conv, stride 1, 192<br>bilinear interpolation | 3×3 DWconv, stride 2, 128<br>1×1 conv, stride 1, 256<br>bilinear interpolation | 3×3 DWconv, stride 2, 144<br>1×1 conv, stride 1, 288<br>bilinear interpolation | 3×3 DWconv, stride 2, 192<br>1×1 conv, stride 1, 384<br>bilinear interpolation | 3×3 DWconv, stride 2, 256<br>1×1 conv, stride 1, 512<br>bilinear interpolation |
|         |                 | Transformer | MATAttn $h=24$<br>MLP $exp=4$ × 6   | MATAttn $h=16$<br>MLP $exp=4$ × 6  | MATAttn $h=16$<br>MLP $exp=4$ × 8  | MATAttn $h=24$<br>MLP $exp=4$ × 8  | MATAttn $h=32$<br>MLP $exp=4$ × 8  |

 TABLE 4.12: Detailed model configurations of 5 MAT variants. *exp* indicates the expansion ratio in MLP. *h* denotes the number of heads in MAT Attention (MATAttn). DWconv is depth-wise convolution.

al., 2017b) and (He et al., 2017a), we use the training receipts in (Wang et al., 2021a; Liu et al., 2021b). For instance, the input images are resized to  $1333 \times 800$ , and the learning schedule is set to  $1 \times (12 \text{ epochs})$ . The stochastic depth is kept the same as in image classification models. Table 4.11 shows the training settings used in SSD, RetinaNet, and Mask R-CNN models.

### Semantic Segmentation

Following (Wang et al., 2021a; Liu et al., 2021b), we replace ResNet-50 (He et al., 2016) in Semantic FPN (Kirillov et al., 2019) with our MAT backbone. All the models are trained on ADE20K (Zhou et al., 2019) for 80K iterations. The optimizer is AdamW with an initial learning rate of 0.0002, weight decay of 0.0001, and poly learning rate scheduler. The input images are resized to  $512 \times 512$  and the total batch size is 16.

#### 4.6.3 Model Configuration

Because our MAT block can capture long-range dependencies from the image tokens, inserting MAT blocks into stages 3-4 results in better trade-offs between accuracy and efficiency. Therefore, we put more MAT blocks in stages 3-4 and employ pure MLP blocks in stages 1-2 for capturing local features. Table 4.12 shows the detailed configurations of MAT-1, MAT-2, MAT-3, MAT-4, and MAT-5. We also provide more visualizations of attention maps learned by token abstraction and after mixing abstract tokens in Figure 4.6.

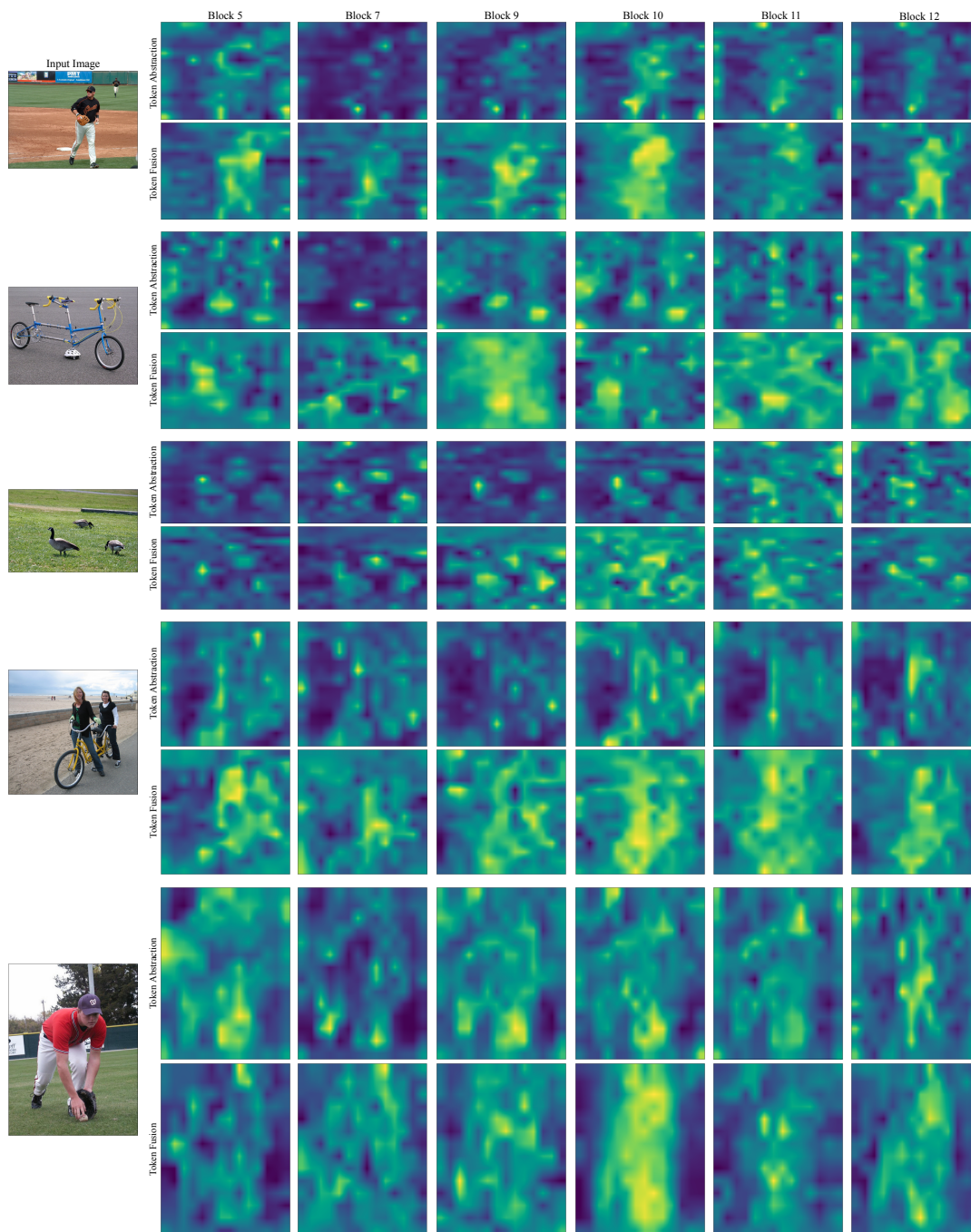


FIGURE 4.6: Visualizations of attention maps across MAT blocks. As can be seen, different blocks capture different information of inputs. In earlier blocks, abstract tokens learn low-level features, and mixing learned abstract tokens results in larger focused regions. In later blocks, mixing abstract tokens produces high-level features, and the image tokens are enhanced.

## Chapter 5

# Efficient Vision Transformers with Partial Attention

### 5.1 Introduction

Transformer (Vaswani et al., 2017) primitively designed for machine translation and achieved remarkable improvements in modeling global token-to-token interactions in an input-dependent manner. With high general modeling capability and scalability to model and data size, shifting Transformer to vision and multimodal foundation models pays a lot of attention. In vision tasks, DETR (Carion et al., 2020) was the first successful method that leverages the Transformer encoder and decoder into object detection models. ViT (Dosovitskiy et al., 2021) views patches as tokens and uses Transformer encoders to model spatial interactions. Although ViT defines a new paradigm in feature extraction, self-attention has quadratic complexity with token lengths. When transferring ViT to downstream tasks, the model suffers a huge computational cost. Hence, a lot of work attempts to alleviate this issue by proposing sparse attention such as spatial reduction attention (Wang et al., 2021a; Wang et al., 2022b; Pan et al., 2022b; Chu et al., 2021b) and window attention (Liu et al., 2021b; Chu et al., 2021b).

To be compatible with dense prediction tasks, PVT (Wang et al., 2021a; Wang et al., 2022b) proposes spatial reduction attention that down-samples the key and value tokens to reduce the model complexity. Swin (Liu et al., 2021b) limits attention areas to local windows and needs cycle shifts to exchange information across windows. Although PVT, Swin, and its variants improve the efficiency of the Transformer, their

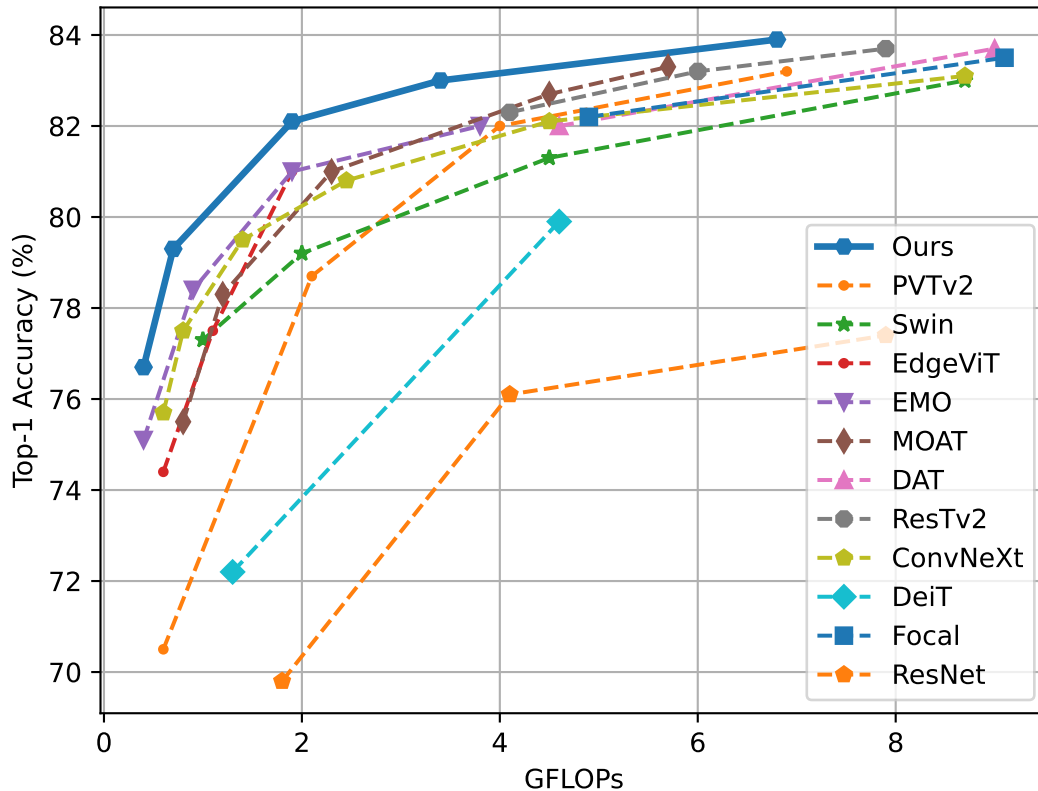
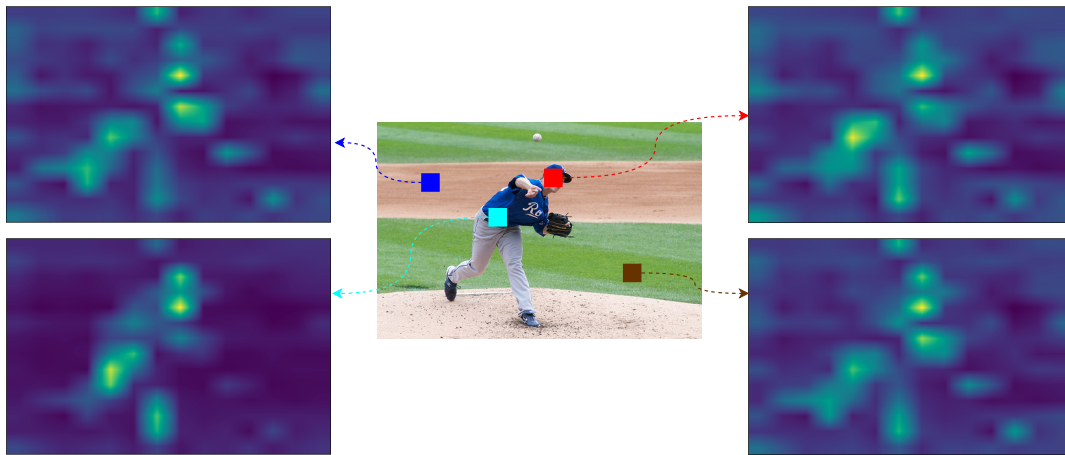


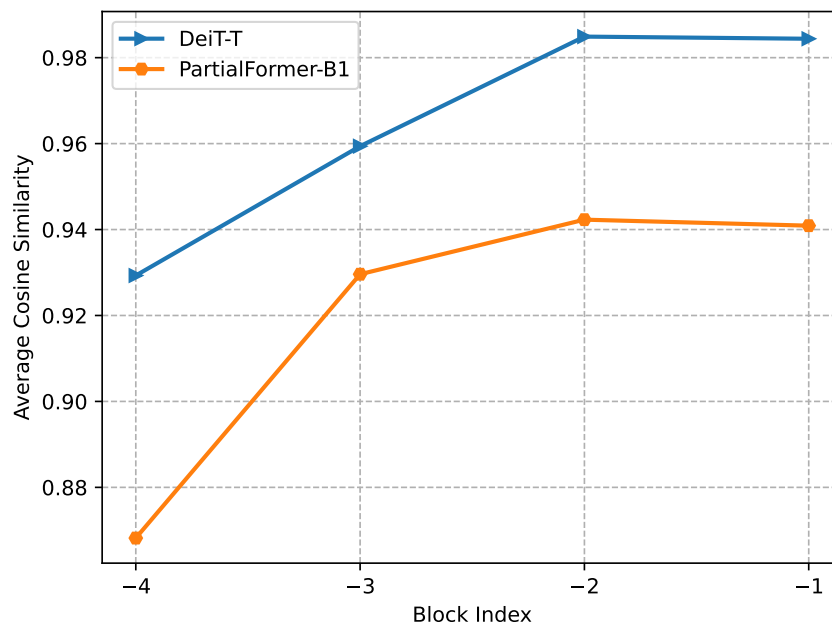
FIGURE 5.1: Performances of recent methods on ImageNet-1K.

attention patterns are data-agnostic and have high similarities. As shown in Figure 5.2a, we observe that attention maps for foreground/background queries are almost the same. It indicates that interacting each query with the full set of key and value tokens may be suboptimal and result in computation redundancy. This is further verified by average cosine similarity computation in Figure 5.2b. Moreover, spatial reduction attention still remains less important tokens while informative tokens are ignored. Receptive fields of window attention are not sufficient and lead to weak modeling capability.

An intuitive way is to force a query to attend to a small set of key and value tokens such as deformable key/value tokens (Xia et al., 2022), top-k selection of local windows (Zhu et al., 2023), top-k selection of regions at each token pyramid (Tang et al., 2022), and evaluation of attention weights (Gupta et al., 2021; Wang et al., 2022a). However, these methods still require interactions between all queries and selected keys/values (Xia et al., 2022; Zhu et al., 2023; Tang et al., 2022), or have the quadratic complexity (Gupta et al., 2021; Wang et al., 2022a). It means that all input tokens participate in computing attention weights. Different queries produce



(A) Attention maps of various query positions



(B) Average cosine similarity between query points.

FIGURE 5.2: (a) Four attention maps in a self-attention layer are almost the same. (b) Average cosine similarity between query points in the last four blocks of DeiT (Touvron et al., 2021) and our PartialFormer.

almost the same attention patterns and only a few queries dominate the contribution to final attention maps. This motivates us to develop efficient vision Transformers by mitigating computation redundancy of query-key interactions while still keeping the high performances of ViT.

To address this issue, this paper proposes intuitive solutions, called partial attention, that significantly reduce computation redundancy in MSA and enhance the diversity of attention heads. The key idea is to directly split the set of image tokens into two sets, a foreground set with few tokens and the remaining tokens as a background set, based on the evaluation of context scores. Next, foreground tokens are fed into MSA to learn relevant features, and attention heads for the foreground set are mixed via simple head MLP to augment the diversity of attention heads. In this way, foreground queries only focus on important regions and thus, the model captures informative features at a low computational cost as the number of foreground tokens is much smaller than the number of all image tokens. This modification of MSA is called Mixed Multi-Head Self-Attention (MMSA). As background tokens contain less important information, we introduce efficient single-query attention (SQA) to force one unique query to attend to the background set. Single-query attention only results in linear complexity with background token length. Therefore, a lot of computational costs are cut down while still keeping the global context modeling of the Transformer.

However, MMSA performs interactions on foreground tokens (a partial part of image tokens) while SQA squeezes the information of the background set. Although this mechanism improves efficiency, two sets are not communicated to each other. Hence, this paper introduces an efficient information exchange by adding a learnable abstract token to the foreground set. After abstracting information from all foreground tokens, this token serves as a single query, and background tokens act as key-value pairs. Attendance of a single query to background tokens can allow the information exchange among tokens with high efficiency. In summary, the main contributions of this paper are as follows:

- We propose novel partial attention, where tokens are treated unevenly according to the importance of tokens. Relevant tokens are grouped into the foreground set and the model learns informative features from this set via simple



Mixed Multi-Head Self-Attention (MMSA). Less important tokens are gathered into the background set and efficient Single-Query Attention (SQA) performs interactions between single query and background tokens. Furthermore, the efficient information exchange between foreground and background sets is proposed to enhance the modeling ability.

- We present PartialFormer to clarify the efficiency and effectiveness of the partial Transformer block. As shown in Figure 5.1, PartialFormer with various scaling models achieves better trade-offs between Top-1 accuracy and computational costs. Typically, PartialFormer-B3 surpasses DAT-T (Xia et al., 2022) by 1.0% Top-1 accuracy while saving 25% GFLOPs. We also apply PartialFormer for downstream tasks and achieve comparative performances with recent methods across various tasks.

## 5.2 Related Works

**Vision Transformers.** Though ViT (Dosovitskiy et al., 2021) exchanges global information between image tokens via a data-driven way, there are two main drawbacks: weak inductive biases (translation-equivariance and locality) and quadratic complexity with input lengths. For the first drawback, ViT requires large-scale datasets (Deng et al., 2009; Sun et al., 2017b) or strong data augmentation & distillation (Touvron et al., 2021) to train the models. Another solution is to inject locality and translation-invariant of vanilla convolution inside self-attention layers (Guo et al., 2022a; Dai et al., 2021b; Liu et al., 2021b; Dong et al., 2022; Liu et al., 2023) or outside self-attention layers (Wang et al., 2022b; Pan et al., 2022b; Liu et al., 2023; Grainger et al., 2023; Pan et al., 2023; Lin et al., 2023; Zhang et al., 2023). For the model cost, many works try to reduce the complexity of self-attention to be smaller (Wang et al., 2021a; Wang et al., 2022b; Pan et al., 2022b; Zhang and Yang, 2022), linear (Liu et al., 2021b; Dong et al., 2022; Mehta and Rastegari, 2023; Chen et al., 2022b), and apply vanilla self-attention at latter stages (Pan et al., 2022c; Pan, Cai, and Zhuang, 2022).

**Efficient Vision Transformers.** Many methods are proposed to address the computational bottleneck of global self-attention by using sparse attention. There are two popular ways: spatial reduction attention (Wang et al., 2021a; Wang et al., 2022b; Pan et al., 2022b; Zhang and Yang, 2022; Xia et al., 2022; Gupta et al., 2021; Wang et

al., 2022a; Tang et al., 2022) and local self-attention (Liu et al., 2021b; Dong et al., 2022; Tu et al., 2022; Zhu et al., 2023). The common goal of the spatial reduction attention is that each query interacts with down-sampled key/value tokens via depthwise convolution (Wang et al., 2021a; Pan et al., 2022b; Zhang and Yang, 2022), average pooling (Wang et al., 2022b), deformable operation (Xia et al., 2022),  $k$  candidate tokens (Gupta et al., 2021; Tang et al., 2022), and masked attention weights (Wang et al., 2022a). In local self-attention research, Swin Transformer (Liu et al., 2021b) is the state-of-the-art method that achieves linear complexity while adopting both merits of self-attention and convolution. Self-attention in Swin is performed within non-overlapped windows and information across windows is exchanged via cycle shifts. However, Swin has limited receptive fields and weak modeling capabilities. A lot of works are introduced to solve this problem such as window expanding (Dong et al., 2022), window shuffling (Tu et al., 2022), window sliding (Pan et al., 2023), and  $k$  candidate windows (Zhu et al., 2023). Although sparse attention methods improve the efficiency of vanilla self-attention, they still incur computation redundancy. Therefore, in this paper, we investigate a way that further reduces the computational cost of sparse attention to build an efficient vision Transformer.

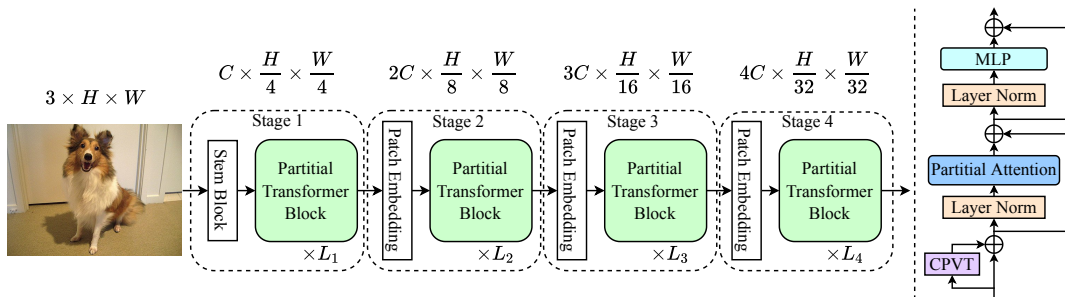


FIGURE 5.3: Left: the overall architecture of PartialFormer. Right: Partial Transformer block.  $L_1$  to  $L_4$  are the number of stacked partial Transformer blocks. CPVT (Chu et al., 2023) is adopted as positional encoding. MLP denotes a multi-layer perceptron, consisting of two fully connected layers and GELU inserted between them.

## 5.3 Partial Vision Transformer

### 5.3.1 Overview of multi-head self-attention

As the core of Transformer, multi-head self-attention (MSA) has high flexibility in modeling long-range dependencies from spatial locations. Its attention mechanism

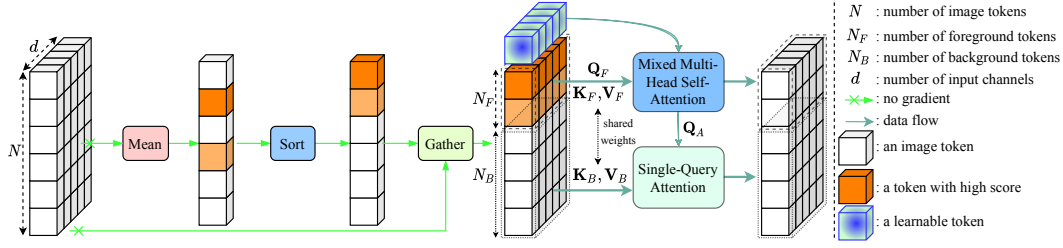


FIGURE 5.4: The structure of partial attention. Image tokens are separated into foreground and background tokens based on the evaluation of context scores. Mixed Multi-Head Self-Attention is proposed to capture informative features from the foreground set and Single-Query Attention squeezes information from background tokens to partially focus on background regions. Learnable query  $\mathbf{Q}_A$  serves as a bridge between foreground and background tokens. Note that linear projections in two attentions are shared to further reduce the cost.

relies on query-key interactions where each query attends to all spatial locations. Technically, given the input token  $\mathbf{X} \in \mathbb{R}^{N \times d}$ , MSA is defined as,

$$\text{MSA}(\mathbf{X}) = \text{concat} [\text{SA}_h(\mathbf{X})] \mathbf{W}_O, \quad (5.1)$$

$$\text{SA}_h(\mathbf{X}) = \text{Softmax} \left( \frac{\mathbf{Q}\mathbf{K}^\top}{\sqrt{d_h}} \right) \mathbf{V}, \quad (5.2)$$

where  $\mathbf{Q} = \mathbf{X}\mathbf{W}_Q$ ,  $\mathbf{K} = \mathbf{X}\mathbf{W}_K$ ,  $\mathbf{V} = \mathbf{X}\mathbf{W}_V$  are query, key, and value projected from the same input token  $\mathbf{X}$  via linear transforms  $\mathbf{W}_Q \in \mathbb{R}^{d \times d_q}$ ,  $\mathbf{W}_K \in \mathbb{R}^{d \times d_k}$ ,  $\mathbf{W}_V \in \mathbb{R}^{d \times d_v}$ .  $\mathbf{W}_O \in \mathbb{R}^{d \times d}$  is output linear projection after all heads are concatenated.  $N = H \times W$  is the token length (height  $\times$  width),  $d$  is a number of channels,  $d_{q,k,v} = d/h$  is a head dimension, and  $h$  is a number of heads.

The MSA splits  $\mathbf{X}$  into  $h$  heads along channel dimension where each head has a size of  $N \times d_h$ . For each head, self-attention SA learns spatial interactions between tokens via query-key matrix multiplication  $\mathbf{Q}\mathbf{K}^\top \in \mathbb{R}^{N \times N}$ . Each row of matrix  $\mathbf{Q}\mathbf{K}^\top$  denotes a weighted sum of one query over all keys. The softmax() is used on each row of  $\mathbf{Q}\mathbf{K}^\top$  to compute attention weight  $\mathbf{A} \in \mathbb{R}^{N \times N}$ . Obviously, matrix multiplication  $\mathbf{Q}\mathbf{K}^\top$  results in quadratic complexity  $\mathcal{O}(N^2)$ . This bottleneck limits applications of the Transformer to downstream tasks.

Spatial reduction attentions (SRA) (Wang et al., 2021a; Wang et al., 2022b; Chu et al., 2021b) alleviate the bottleneck of MSA by shortening the key and value lengths

to  $N/r^2$ , addressed as follows:

$$\mathbf{K}', \mathbf{V}' = \text{SR}(K, V), \quad (5.3)$$

where  $\text{SR}()$  is implemented by convolution (Wang et al., 2021a; Chu et al., 2021b), average pooling (Wang et al., 2022b; Pan et al., 2022b; Pan, Cai, and Zhuang, 2022), and deformed tokens (Xia et al., 2022).

Window attentions (Liu et al., 2021b; Dong et al., 2022; Zhu et al., 2023; Zhang et al., 2023) achieve linear complexity with token lengths by limiting self-attention within window regions.  $\mathbf{Q}, \mathbf{K}, \mathbf{V}$  are partitioned into windows with dimension  $N_w \times h \times w^2 \times C_d$  and attention weights are performed within  $w^2$ . However, these models have limited receptive fields and weak modeling capabilities. To enlarge receptive fields, the model requires stacked blocks and extra designs to exchange information across windows.

Although sparse attention such as SRA and window attention reduces the cost of the Transformer, such attention treats tokens equally and heads independently. It leads to two main issues. First, computation redundancy: there are few relevant regions (foreground tokens) and many unimportant tokens (background tokens). Attendance of each query to all spatial locations/irrelevant regions produces a lot of calculations and lost information. Moreover, attention patterns are data-agnostic and have high similarities. To mitigate this issue, this paper treats foreground and background tokens differently. The model fully captures informative features from foreground sets and weakens the contribution of the background set to feature learning. Second, diversity of features: as shown in Equation 5.1, there is no communication across attention heads. Such independent treatment can impair performance and limit the diversity ability of Transformers. The solution for this issue is to utilize simple MLP Mixer across heads that increases the number of heads and also mixes attention maps in higher dimensions.

### 5.3.2 Partial Transformer

Partial attention learns spatial interaction more efficiently by focusing on foreground regions in the feature maps and squeezing the information of the most background

tokens. The structure of partial attention is shown in Figure 5.4. Detailed analysis is described in the following section.

### Token Separation

Given the image token  $\mathbf{X} \in \mathbb{R}^{N \times d}$ , we separate  $\mathbf{X}$  into two sets: foreground set  $\mathbf{X}_F \in \mathbb{R}^{N_F \times d}$  and background set  $\mathbf{X}_B \in \mathbb{R}^{N_B \times d}$ . To achieve this goal, the evaluation of tokens is performed on the context score summarized from the input token. This is determined by utilizing  $\text{Mean}()$  operation along channel dimension to create spatial-wise vector  $\mathbf{c} \in \mathbb{R}^N$ . Then, we sort the value of context vector  $\mathbf{c}$  in descending order and store the sorted indices in the vector  $\mathbf{s} \in \{s_1, s_2, \dots, s_N\}$ . Intuitively, the tokens  $\mathbf{X}[s_1, :]$  and  $\mathbf{X}[s_N, :]$  are the most relevant and less important tokens, respectively. Therefore, based on sorted vector  $\mathbf{s}$ , we gather the input token  $\mathbf{X}$  along the token dimension. Briefly, these processes are addressed as follows:

$$\mathbf{c} = \text{Mean}(\mathbf{X}), \mathbf{s} = \text{Sort}(\mathbf{c}), \mathbf{X}_G = \text{Gather}(\mathbf{X}, \mathbf{s}) \quad (5.4)$$

where  $\mathbf{X}_G \in \mathbb{R}^{N \times C}$  is the gathered image token. The index vector  $\mathbf{s} \in \mathbb{R}^N$  is repeated along channel dimension to keep gathered information consistent.

With the determination of the gathered image token, we directly separate  $\mathbf{X}_G$  into the foreground token  $\mathbf{X}_F = \mathbf{X}[:, N_F, :] \in \mathbb{R}^{N_F \times d}$  and  $\mathbf{X}_B = \mathbf{X}[N_F :, :] \in \mathbb{R}^{N_B \times d}$ , where  $N_F, N_B$  is the number of foreground, background tokens and  $N = N_F + N_B$ . The foreground set composes relevant tokens and the background set composes less important tokens. As most tokens contain background information,  $N_F \ll N_B$  is set. Therefore, feeding foreground tokens into global MSA only produces an inconsiderable cost while directly capturing relevant information from the foreground set. For the background set, interacting one latent token with background tokens can encode the necessary information of this set and result in linear complexity. This is achieved by our single-query attention (SQA).

### Mixed Multi-Head Self-Attention

After acquiring the foreground set, MSA is applied to capture global information between relevant tokens. To promote diversity of attention heads in MSA, in this

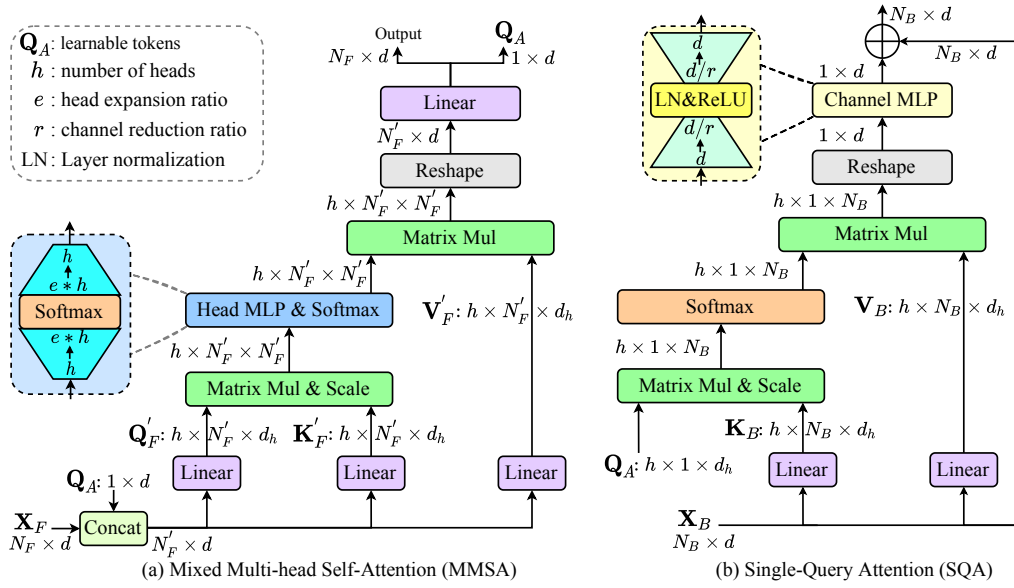


FIGURE 5.5: (a) Detailed structure of Mixed Multi-head Self-Attention and (b) Detailed structure of Single-Query Attention (SQA).

paper, we introduce Mixed Multi-Head Self-Attention (MMSA) that mixes head information in the higher dimension, and modification is minor compared to MSA:

$$\mathbf{A}_F = \text{HeadMLP} \left( \frac{\mathbf{Q}_F \mathbf{K}_F^\top}{\sqrt{d_h}} \right), \quad (5.5)$$

where  $\mathbf{A}_F$  is foreground attention weights.  $\mathbf{Q}_F = \mathbf{X}_F \mathbf{W}_Q$ ,  $\mathbf{K}_F = \mathbf{X}_F \mathbf{W}_K$  are foreground query and key tokens. HeadMLP is MLP Mixer applied across heads, including two fully-connected (FC) layers with expansion ratio  $e$  and  $\text{Softmax}()$  inserted between 2 FC layers. Thanks to HeadMLP, dependencies across attention heads are obtained. Finally, mixed attention weights and value  $\mathbf{V}_F$  are aggregated via Matrix Mul, and all heads are concatenated and linearly projected to generate the final output  $\mathbf{O}_F$ . The detailed architecture is shown in Figure 5.5(a).

### Single-Query Attention

As the background set has many tokens and contains less important information, Single-Query Attention (SQA) is proposed to capture token-to-token attention efficiently. The detailed structure is illustrated in Figure 5.5(b). Formally, given the

background token  $\mathbf{X}_B \in \mathbb{R}^{N_B \times d}$ , SQA is defined as follows:

$$\mathbf{Y}_B = \text{Softmax} \left( \frac{\mathbf{Q}_A \mathbf{K}_B^\top}{\sqrt{d_h}} \right) \mathbf{V}_B, \quad (5.6)$$

$$\mathbf{Y}'_B = \text{ChannelMLP}(\mathbf{Y}_B), \mathbf{O}_B = \mathbf{X}_B + \mathbf{Y}'_B, \quad (5.7)$$

where  $\mathbf{Q}_A \in \mathbb{R}^{1 \times d_h}$  is the single query discussed in the next section.  $\mathbf{K}_B = \mathbf{X}_B \mathbf{W}_K$ ,  $\mathbf{V}_B = \mathbf{X}_B \mathbf{W}_V$  are background key and value tokens. Note that weights in  $\mathbf{W}_{Q,K,V}$  are shared with the MMSA branch to reduce the cost.

$\mathbf{A}_B = \mathbf{Q}_A \mathbf{K}_B^\top \in \mathbb{R}^{1 \times N_B}$  is the attention weights produced by the attendance of one unique query to the background set. Obviously, SQA still encodes all the information from background tokens while enjoying linear complexity with  $N_B$ . This information is propagated to value  $\mathbf{V}_B$  to generate the feature  $\mathbf{Y}_B \in \mathbb{R}^{1 \times d}$ .

Instead of using a linear projection, this paper employs lightweight ChannelMLP that consists of two FC layers with channel reduction  $r$  and Layer Normalization+ReLU inserted between two FC layers to stabilize training. The fusion output  $\mathbf{O}_B$  is implemented by broadcast element-wise addition between  $\mathbf{Y}'_B$  and the input  $\mathbf{X}_B$ .

### The role of single query

This section discusses the way we generate single query  $\mathbf{Q}_A$ . In the partial attention, MMSA is applied for the foreground set, and SQA learns interactions between  $\mathbf{Q}_A$  and background tokens. Foreground tokens are not explicitly connected to background tokens. Therefore, this paper proposes an efficient information exchange and query  $\mathbf{Q}_A$  takes charge of a bridge between two sets.

To achieve this target, we add learnable abstract token  $\mathbf{Q}_A$  to the foreground set  $\mathbf{X}_F$  to generate  $\mathbf{X}'_F \in \mathbb{R}^{N_F+1 \times d}$ . Thanks to the query-key mechanism in MMSA,  $\mathbf{Q}_A$  abstracts information from all foreground tokens, i.e., a weighted sum of all foreground tokens. Hence, interacting abstract token  $\mathbf{Q}_A$  with the background tokens can help to exchange information among tokens at a low cost. As shown in Figure, in SQA, abstracted token  $\mathbf{Q}_A$  acts as the unique query, and background tokens are set as key and value pairs. The SQA layer updates background features with abstracted information from foreground features.

### 5.3.3 Computational Complexity of Partial Attention

The proposed method decomposes spatial interactions of vanilla attention into foreground token-to-token interactions and background token-to-token interactions, which have smaller computational costs than PVT, Swin, and DAT attention. The inconsiderable cost comes from  $\text{HeadMLP}()$  used to augment the diversity of features. The computation of a partial attention module consists of three parts: (1) Shared  $\mathbf{Q}, \mathbf{K}, \mathbf{V}$  projections; (2) mixed multi-head self-attention; and (3) single-query attention, addressed as:

$$\mathcal{O}(\text{proj}) = 3Nd^2, \quad (5.8)$$

$$\mathcal{O}(\text{MMSA}) = 2N_F^2d + (2eh^2d + d^2)N_F, \quad (5.9)$$

$$\mathcal{O}(\text{SSA}) = 3N_Bd + 2\frac{d^2}{r}, \quad (5.10)$$

where  $2eh^2dN_F$  is the minor cost of  $\text{HeadMLP}()$  as  $h$  is much smaller than spatial or channel sizes.  $\text{MMSA}()$  results in quadratic complexity with the length of foreground tokens  $N_F$ . As  $N_F \ll N$ , the cost of  $\text{MMSA}()$  is smaller than spatial reduction attention and window attention. Moreover,  $\text{SSA}()$  is efficient as it only produces linear computational costs with the length of the background set while other methods have linear complexity with the number of all tokens  $N$ . In total, the computational complexity of our attention is smaller than vanilla self-attention and sparse attention. For example, in stage 1 of the classification models ( $H = W = 56, d = 64, h = 8$ ), one partial attention module results in 40.1 MFLOPs while the cost of attention in PVT and Swin is 58.6 MFLOPs and 71.1 MFLOPs, respectively. Therefore, partial attention learns spatial interaction more efficiently.

### 5.3.4 Model Configuration

Similar to the meta-design in the original ViT (Dosovitskiy et al., 2021), the partial Transformer block is introduced as shown in Figure 5.3. CPVT (Chu et al., 2023) is used as positional encoding (Wang et al., 2022b; Chu et al., 2021b; Dong et al., 2022; Guo et al., 2022a; Pan et al., 2022b) to learn local features and also keep geometric information of the 2D images. Based on the acquired block, we propose PartialFormer models that are efficient and versatile vision Transformers. As illustrated in



Figure 5.3, the model consists of four stages, and spatial dimensions across stages are downsampled with rates  $\{4, 8, 16, 32\}$  by convolution-based patch embeddings. Along with spatial reduction, the number of channels is doubled as  $\{C, 2C, 3C, 4C\}$  across four stages. Finally, global average pooling and linear layers are adopted to predict class logits.

**Model Variants.** We build five models (B0-B4) by changing the number of stacked blocks  $L_i$  ( $i \in \{1, 2, 3, 4\}$ ) and base channels  $C$ . The costs of the models range from 0.4 GFLOPs to 6.8 GFLOPs. The detailed configurations are listed in Table 5.1.

| Models           | $C$ | $L$          | #heads          | #Params | FLOPs |
|------------------|-----|--------------|-----------------|---------|-------|
| PartialFormer-B0 | 24  | [2, 2, 6, 6] | [2, 4, 8, 16]   | 5.3M    | 0.4G  |
| PartialFormer-B1 | 32  | [2, 2, 6, 6] | [2, 4, 8, 16]   | 8.2M    | 0.7G  |
| PartialFormer-B2 | 48  | [2, 2, 8, 8] | [3, 6, 12, 24]  | 21.1M   | 1.9G  |
| PartialFormer-B3 | 64  | [2, 2, 8, 8] | [4, 8, 16, 32]  | 36.1M   | 3.4G  |
| PartialFormer-B4 | 96  | [2, 2, 8, 6] | [6, 12, 24, 48] | 64.5M   | 6.8G  |

TABLE 5.1: Detailed configurations of five PartialFormers.

## 5.4 Experiments and Results

To validate the PartialFormer, we conduct experiments on ImageNet-1K (Russakovsky et al., 2015) classification, MS-COCO (Lin et al., 2014) object detection, and ADE20K semantic segmentation (Zhou et al., 2019). Firstly, we train the models on ImageNet-1K from scratch and present Top-1 accuracy on image classification. Secondly, we fine-tune the pre-trained backbones for downstream tasks. Finally, ablation studies are conducted to verify the effectiveness of partial attention and ablate other design choices.

### 5.4.1 ImageNet-1K Classification

**Settings.** PartialFormer models are trained and evaluated on ImageNet-1K (Russakovsky et al., 2015). We follow the training receipts in (Touvron et al., 2021; Wang et al., 2021a; Liu et al., 2021b) for fair comparisons. Typically, the models are trained for 300 epochs with an image size of  $224 \times 224$ . We adopt the AdamW optimizer with an initial learning rate of  $1 \times 10^{-3}$  and a weight decay of 0.005. Data augmentation and regularization are used the same as DeiT and Swin, including RandAugment

(Cubuk et al., 2020), MixUp (Zhang et al., 2017), CutMix (Yun et al., 2019), Random Erasing, and stochastic depth (Huang et al., 2016).

**Results.** We report classification results on ImageNet-1K in Table 5.2 and 5.3. Compared to recent efficient CNNs and vision Transformers, our PartialFormer from B0 to B4 attains significant improvements on classification accuracy with smaller computational costs. For lightweight models, in Table 5.2, PartialFormer-B0/B1/B2 outperforms EdgeViT-XXS/XS/S (Pan et al., 2022b) by 3.0%, 2.2%, 1.0% under smaller costs. PartialFormer-B1 achieves 79.3% Top-1 accuracy that surpasses concurrent work EMO-5M (Zhang et al., 2023) by 0.9% while saving 22% GFLOPs, SwiftFormer-S by 0.8% with only 70% GFLOPs, and state-of-the-art MixFormer-B1 (Chen et al., 2022a) by 0.4% with similar costs.

For larger models, in Table 5.3, PartialFormer-B3 achieves 83.0% Top-1 accuracy with only 3.4 GFLOPs, outperforming Swin-T (Liu et al., 2021b) by 1.7% while saving 25% GFLOPs, the most competitive DAT-T (Xia et al., 2022) by 1.0% with only 75% costs, Focal-T by 0.8% while saving 31% GFLOPs, efficient ConNeXtV2-T by 0.5% with only 75% GFLOPs, and stronger model CSWin-T by 0.3%. When scaling PartialFormer from B3 to B4, our method still keeps consistent gains while having smaller GFLOPs. It demonstrates our models achieve better trade-offs between Top-1 and computational complexity.

#### 5.4.2 MS-COCO Object Detection and Segmentation

**Settings.** Next, we transfer the pre-trained PartialFormer backbones on ImageNet-1K for MS-COCO (Lin et al., 2014) object detection and instance segmentation by using the fine-tuning technique. For lightweight backbones PartialFormer B0-B1, SSDLite (Liu et al., 2016) is used as the detection baseline and we adopt experimental settings (Mehta and Rastegari, 2022; Mehta and Rastegari, 2023) to train the models. For larger models, we integrate PartialFormer B1-B4 into object detectors: RetinaNet (Lin et al., 2017b), ATSS (Zhang et al., 2020), and Sparse-RCNN (Sun et al., 2021a); instance segmentation Mask R-CNN (He et al., 2017a). Specifically, we follow common experimental receipts (Wang et al., 2021a; Wang et al., 2022b; Liu et al., 2021b) provided by the codebase (Chen et al., 2019b) to validate the effectiveness of PartialFormers. All the models are trained for 12 epochs ( $1 \times \text{schedule}$ ) with a batch

| Method                                    | FLOPs<br>(G) | Params<br>(M) | Top-1 Acc<br>(%) |
|---|--------------|---------------|------------------|
| MViTv1-XXS (Mehta and Rastegari, 2022)    | 0.4          | 1.3           | 69.0             |
| MViTv2-0.5 (Mehta and Rastegari, 2023)    | 0.5          | 1.4           | 70.2             |
| PVTv2-B0 (Wang et al., 2022b)             | 0.6          | 3.7           | 70.5             |
| DFvT-T (Gao et al., 2022)                 | 0.3          | 4.0           | 73.0             |
| EfficientViT-M4 (Liu et al., 2023)        | 0.4          | 8.8           | 74.3             |
| EdgeViT-XXS (Pan et al., 2022b)           | 0.6          | 4.1           | 74.4             |
| SwiftFormer-XS (Shaker et al., 2023)      | 0.6          | 3.5           | 75.7             |
| <b>PartialFormer-B0</b>                   | <b>0.4</b>   | <b>5.3</b>    | <b>76.7</b>      |
| DeiT-T (Touvron et al., 2021)             | 1.3          | 6.0           | 72.2             |
| MViTv1-XS (Mehta and Rastegari, 2022)     | 1.0          | 2.3           | 74.8             |
| LVT (Yang et al., 2022a)                  | 0.9          | 3.4           | 74.8             |
| MViTv2-0.75 (Mehta and Rastegari, 2023)   | 1.0          | 2.9           | 75.6             |
| ConvNeXtV1-A (Liu et al., 2022c)          | 0.6          | 3.7           | 75.7             |
| ConvNeXtV2-A (Woo et al., 2023)           | 0.6          | 3.7           | 76.2             |
| ResT-Lite (Zhang and Yang, 2021)          | 1.4          | 10.5          | 77.2             |
| Swin-1G* (Liu et al., 2021b)              | 1.0          | 7.3           | 77.3             |
| EdgeViT-XS (Pan et al., 2022b)            | 1.1          | 6.7           | 77.5             |
| DFvT-S (Gao et al., 2022)                 | 0.8          | 11.2          | 78.3             |
| tiny-MOAT-1 (Yang et al., 2023)           | 1.2          | 5.1           | 78.3             |
| EMO-5M (Zhang et al., 2023)               | 0.9          | 5.1           | 78.4             |
| SwiftFormer-S (Shaker et al., 2023)       | 1.0          | 6.1           | 78.5             |
| MixFormer-B1 (Chen et al., 2022a)         | 0.7          | 8.0           | 78.9             |
| CMT-Ti <sup>†</sup> (Guo et al., 2022a)   | 0.6          | 9.5           | 79.1             |
| <b>PartialFormer-B1</b>                   | <b>0.7</b>   | <b>8.2</b>    | <b>79.3</b>      |
| PVT-T (Wang et al., 2021a)                | 1.8          | 13.0          | 75.1             |
| Slide-PVT-T (Pan et al., 2023)            | 2.0          | 12.2          | 78.0             |
| PVTv2-B1 (Wang et al., 2022b)             | 2.1          | 13.1          | 78.7             |
| Swin-2G* (Liu et al., 2021b)              | 2.0          | 12.8          | 79.2             |
| ResT-Small (Zhang and Yang, 2021)         | 1.9          | 13.7          | 79.6             |
| Shunted-T (Ren et al., 2022)              | 2.1          | 11.5          | 79.8             |
| GC ViT-XXT (Hatamizadeh et al., 2023b)    | 2.1          | 12.0          | 79.9             |
| QuadTree-B-B1 (Tang et al., 2022)         | 2.3          | 13.6          | 80.0             |
| ConvNeXtV1-N (Liu et al., 2022c)          | 2.5          | 15.6          | 80.8             |
| SwiftFormer-T (Shaker et al., 2023)       | 1.6          | 12.1          | 80.9             |
| tiny-MOAT-2 (Yang et al., 2023)           | 2.3          | 9.8           | 81.0             |
| EdgeViT-S (Pan et al., 2022b)             | 1.9          | 11.1          | 81.0             |
| ConvNeXtV2-N (Woo et al., 2023)           | 2.5          | 15.6          | 81.2             |
| BiFormer-T (Zhu et al., 2023)             | 2.2          | 13.1          | 81.4             |
| EffNet-B3 <sup>‡</sup> (Tan and Le, 2019) | 1.8          | 12.0          | 81.6             |
| Twins-SVT-S (Chu et al., 2021b)           | 2.9          | 24.0          | 81.7             |
| <b>PartialFormer-B2</b>                   | <b>1.9</b>   | <b>21.1</b>   | <b>82.0</b>      |

TABLE 5.2: Classification performances of our PartialFormer (B0-B2) and recent methods on ImageNet-1K. All the models are trained for 300 epochs, except for CMT-Ti<sup>†</sup> and EffNet-B3<sup>‡</sup> trained for 800 epochs and 350 epochs, respectively. \* denotes the downscaled versions of Swin Transformer (Liu et al., 2021b).

| Method                               | FLOPs<br>(G) | Params<br>(M) | Top-1 Acc<br>(%) |
|--------------------------------------|--------------|---------------|------------------|
| ResNet50 (He et al., 2016)           | 4.1          | 26            | 76.1             |
| PVT-S (Wang et al., 2021a)           | 3.8          | 25            | 79.8             |
| DeiT-S (Touvron et al., 2021)        | 4.6          | 22            | 79.9             |
| PaCa-Tiny (Grainger et al., 2023)    | 3.2          | 12            | 80.9             |
| Swin-T (Liu et al., 2021b)           | 4.5          | 29            | 81.3             |
| LIT-S (Pan et al., 2022c)            | 4.1          | 27            | 81.5             |
| ResT-Base (Zhang and Yang, 2021)     | 4.3          | 30            | 81.6             |
| Slide-PVT-S (Pan et al., 2023)       | 4.0          | 23            | 81.7             |
| PVTv2-B2 (Wang et al., 2022b)        | 4.0          | 25            | 82.0             |
| DAT-T (Xia et al., 2022)             | 4.5          | 29            | 82.0             |
| EMO-20M (Zhang et al., 2023)         | 3.8          | 20            | 82.0             |
| LITv2-S (Pan, Cai, and Zhuang, 2022) | 3.7          | 28            | 82.0             |
| ConvNeXt-T (Liu et al., 2022c)       | 4.5          | 29            | 82.1             |
| Focal-T (Yang et al., 2021)          | 4.9          | 29            | 82.2             |
| ResTv2-T (Zhang and Yang, 2022)      | 4.1          | 30            | 82.3             |
| ConvNeXtV2-T (Woo et al., 2023)      | 4.5          | 29            | 82.5             |
| tiny-MOAT-3 (Yang et al., 2023)      | 4.5          | 20            | 82.7             |
| CSWin-T (Dong et al., 2022)          | 4.5          | 23            | 82.7             |
| <b>PartialFormer-B3</b>              | <b>3.4</b>   | <b>36</b>     | <b>83.0</b>      |
| PVT-M (Wang et al., 2021a)           | 6.7          | 44            | 81.2             |
| PVT-S (Wang et al., 2021a)           | 9.8          | 61            | 81.7             |
| Swin-S (Liu et al., 2021b)           | 8.7          | 50            | 83.0             |
| Twins-SVT-B (Chu et al., 2021b)      | 8.6          | 56            | 83.2             |
| PVTv2-B3 (Wang et al., 2022b)        | 6.9          | 45            | 83.2             |
| LITv2-M (Pan et al., 2022c)          | 7.5          | 49            | 83.3             |
| Focal-S (Yang et al., 2021)          | 9.1          | 51            | 83.6             |
| CSWin-S (Dong et al., 2022)          | 6.9          | 35            | 83.6             |
| DAT-S (Xia et al., 2022)             | 9.0          | 50            | 83.6             |
| PVTv2-B4 (Wang et al., 2022b)        | 10.1         | 63            | 83.6             |
| ResTv2-B (Zhang and Yang, 2022)      | 7.9          | 56            | 83.7             |
| <b>PartialFormer-B4</b>              | <b>6.8</b>   | <b>64</b>     | <b>83.9</b>      |

TABLE 5.3: Comparison of PartialFormer-B3, B4 with recent methods on ImageNet-1K.

size of 16 and input images are resized to  $1333 \times 800$ . The optimizer AdamW with a learning rate of  $1 \times e^{-4}$  and a weight decay of 0.05 is utilized to fine-tune the models.

**Results.** Table 5.4 shows the comparisons of various backbones on detectors RetinaNet (Lin et al., 2017b) and Mask R-CNN (He et al., 2017a). With less computational costs, PartialFormer achieves consistent improvements over other methods (He et al., 2016; Wang et al., 2021a; Pan et al., 2022c; Liu et al., 2021b; Chu et al., 2021b; Xia et al., 2022). For example, in the RetinaNet detector, PartialFormer-B3

| Backbone                     | RetinaNet 1× |            |                  | Mask R-CNN 1× |            |                  |                    |
|------------------------------|--------------|------------|------------------|---------------|------------|------------------|--------------------|
|                              | Param(M)     | GFLOPs     | AP <sup>bb</sup> | Param(M)      | GFLOPs     | AP <sup>bb</sup> | AP <sup>mask</sup> |
| ResNet-18 (He et al., 2016)  | 21           | 189        | 31.8             | 31            | 207        | 34.0             | 31.2               |
| ResNet-50 (He et al., 2016)  | 38           | 250        | 36.3             | 44            | 260        | 38.0             | 34.4               |
| ResNet-101 (He et al., 2016) | 57           | 315        | 38.5             | 63            | 336        | 40.4             | 36.4               |
| PVT-T (Wang et al., 2021a)   | 23           | 183        | 36.7             | 33            | 208        | 36.7             | 35.1               |
| PVT-S (Wang et al., 2021a)   | 34           | 273        | 40.4             | 44            | 245        | 40.4             | 37.8               |
| PVT-M (Wang et al., 2021a)   | 54           | 384        | 41.9             | 64            | 367        | 42.0             | 39.0               |
| LIT-S (Pan et al., 2022c)    | 39           | 305        | 41.6             | 48            | 324        | 42.9             | 39.6               |
| Swin-T (Liu et al., 2021b)   | 38           | 251        | 41.5             | 48            | 270        | 42.2             | 39.1               |
| Twin-S (Chu et al., 2021b)   | 34           | 225        | 43.0             | 44            | 244        | 43.4             | 40.3               |
| DAT-T (Xia et al., 2022)     | 38           | 253        | 42.8             | 48            | 272        | 44.4             | 40.4               |
| <b>PartialFormer-B1</b>      | <b>16</b>    | <b>167</b> | <b>40.2</b>      | <b>26</b>     | <b>185</b> | <b>41.2</b>      | <b>38.2</b>        |
| <b>PartialFormer-B2</b>      | <b>29</b>    | <b>196</b> | <b>43.5</b>      | <b>39</b>     | <b>214</b> | <b>44.1</b>      | <b>40.4</b>        |
| <b>PartialFormer-B3</b>      | <b>44</b>    | <b>230</b> | <b>44.1</b>      | <b>54</b>     | <b>248</b> | <b>45.0</b>      | <b>40.9</b>        |

TABLE 5.4: Object detection and instance segmentation performances on MS-COCO validation set using RetinaNet (Lin et al., 2017b) and Mask R-CNN (He et al., 2017a). GFLOPs are calculated with the input size  $1280 \times 800$ .

| Backbone                                | Image size       | GFLOPs     | Param(M)   | mAP         |
|---|------------------|------------|------------|-------------|
| MViTv1-XXS (Mehta and Rastegari, 2022)  | $320 \times 320$ | 0.9        | 1.7        | 19.9        |
| MViTv2-0.5 (Mehta and Rastegari, 2023)  | $320 \times 320$ | 0.9        | 2.0        | 21.2        |
| MNetv3 (Howard et al., 2019)            | $320 \times 320$ | 0.6        | 5.0        | 22.0        |
| MNetv2 (Sandler et al., 2018)           | $320 \times 320$ | 0.8        | 4.3        | 22.1        |
| MNetv1 (Howard et al., 2017)            | $320 \times 320$ | 1.3        | 5.1        | 22.2        |
| MViTv2-0.75 (Mehta and Rastegari, 2023) | $320 \times 320$ | 1.8        | 3.6        | 24.6        |
| ResNet-50 (He et al., 2016)             | $320 \times 320$ | 20.2       | 22.9       | 25.2        |
| <b>PartialFormer-B0</b>                 | $320 \times 320$ | <b>0.9</b> | <b>5.0</b> | <b>24.3</b> |
| <b>PartialFormer-B1</b>                 | $320 \times 320$ | <b>1.5</b> | <b>8.0</b> | <b>27.1</b> |

TABLE 5.5: Object detection performances using SSDLite (Liu et al., 2016).

gets 43.1% AP<sup>bb</sup> higher than PVT-M by 1.2% while saving 60% GFLOPs, Swin-T by 1.5%. For instance segmentation Mask R-CNN, our model achieves better performances than previous methods. The results indicate that our finetuned models keep the efficiency of partial attention on higher resolution while enabling higher performances with reduced computational costs.

Table 5.5 reports the performance of our smaller model and lightweight backbones (Howard et al., 2017; Sandler et al., 2018; Howard et al., 2019; Mehta and Rastegari, 2022; Mehta and Rastegari, 2023; He et al., 2016) using SSDLite (Liu et al., 2016). Compared to other methods, PartialFormer-B0/B1 still achieves consistent improvements similar to RetinaNet and Mask R-CNN.

**MS-COCO keypoint detection.** We also evaluate PartialFormer on 2D keypoint detection using SimpleBaseline (Xiao, Wu, and Wei, 2018) for general-purpose backbones. We follow receipts in (Xiao, Wu, and Wei, 2018; Wang et al., 2021a; Liu et al.,

| Backbone                     | Crop size      | Params (M)  | GFLOPs     | AP <sup>kp</sup> | AP <sub>50</sub> <sup>kp</sup> | AP <sub>75</sub> <sup>kp</sup> |
|------------------------------|----------------|-------------|------------|------------------|--------------------------------|--------------------------------|
| RSN-18 (Cai et al., 2020)    | 256×192        | 9.1         | 2.3        | 70.4             | 88.7                           | 77.9                           |
| ResNet-50 (He et al., 2016)  | 256×192        | 34.0        | 5.5        | 71.8             | 89.8                           | 79.6                           |
| ResNet-101 (He et al., 2016) | 256×192        | 53.0        | 9.1        | 72.8             | 90.4                           | 80.9                           |
| PVT-S (Wang et al., 2021a)   | 256×192        | 28.2        | 4.1        | 71.4             | 89.6                           | 79.4                           |
| Swin-T (Liu et al., 2021b)   | 256×192        | 32.8        | 6.1        | 72.4             | 90.1                           | 80.6                           |
| <b>PartialFormer-B1</b>      | <b>256×192</b> | <b>9.8</b>  | <b>1.7</b> | <b>70.6</b>      | <b>89.5</b>                    | <b>78.8</b>                    |
| <b>PartialFormer-B2</b>      | <b>256×192</b> | <b>23.0</b> | <b>2.9</b> | <b>72.7</b>      | <b>90.2</b>                    | <b>80.8</b>                    |
| <b>PartialFormer-B3</b>      | <b>256×192</b> | <b>38.4</b> | <b>4.4</b> | <b>73.2</b>      | <b>90.3</b>                    | <b>81.6</b>                    |

TABLE 5.6: 2D keypoint detection using SimpleBaseline (Xiao, Wu, and Wei, 2018).

| Backbone                          | Semantic FPN 80k |              |             | UperNet 160k |              |             |
|-----------------------------------|------------------|--------------|-------------|--------------|--------------|-------------|
|                                   | Param(M)         | GFLOPs       | mIoU        | Param(M)     | GFLOPs       | mIoU        |
| ResNet-50 (He et al., 2016)       | 28.5             | 183          | 36.7        | 66.5         | 951          | 42.0        |
| ResNet-101 (He et al., 2016)      | 47.5             | 260          | 38.8        | 86.0         | 1029         | 43.8        |
| PVT-S (Wang et al., 2021a)        | 28.2             | 161          | 39.8        | -            | -            | -           |
| PVT-M (Wang et al., 2021a)        | 48.0             | 219          | 41.6        | -            | -            | -           |
| Swin-T (Liu et al., 2021b)        | 31.9             | 182          | 41.5        | 59.9         | 945          | 44.5        |
| Focal-T (Yang et al., 2021)       | -                | -            | -           | 62.0         | 998          | 45.8        |
| MixFormer-B3 (Chen et al., 2022a) | -                | -            | -           | 44.0         | 880          | 44.5        |
| DAT-T (Xia et al., 2022)          | 32.0             | 198          | 42.6        | 60.0         | 957          | 45.5        |
| <b>PartialFormer-B1</b>           | <b>10.8</b>      | <b>101.8</b> | <b>40.2</b> | <b>34.8</b>  | <b>856.3</b> | <b>43.3</b> |
| <b>PartialFormer-B2</b>           | <b>23.5</b>      | <b>131.3</b> | <b>42.3</b> | <b>48.6</b>  | <b>886.9</b> | <b>45.9</b> |
| <b>PartialFormer-B3</b>           | <b>38.4</b>      | <b>166.6</b> | <b>43.5</b> | <b>64.5</b>  | <b>923.2</b> | <b>47.0</b> |

TABLE 5.7: ADE20K semantic segmentation performances with Semantic FPN (Kirillov et al., 2019) and UperNet (Xiao et al., 2018). GFLOPs are measured with the input size 2048×512.

2021b) and present results in Table 5.6. As a result, PartialFormer-B3 surpasses baseline methods such as ResNet-50 (He et al., 2016) by 1.4%, PVT-T by 1.8%, and Swin-T (Liu et al., 2021b) by 0.8%, with less computational costs.

### 5.4.3 ADE20K Semantic Segmentation

**Settings.** Our PartialFormer is evaluated on the popular dataset ADE20K (Zhou et al., 2019) for semantic segmentation using two basic methods, Semantic FPN (Kirillov et al., 2019) and UperNet (Xiao et al., 2018). We adopt the experimental settings in Swin (Liu et al., 2021b) and PVT (Wang et al., 2021a) to train Semantic FPN for 80k iterations and UperNet for 160k iterations. Specifically, the input images are resized to 512×512. The optimizer AdamW is used with an initial learning rate of  $2 \times e^{-4}$  and a weight decay of 0.01.

**Results.** Table 5.7 presents performances of our PartialFormer B1/B2/B3 and recent methods. As a result, PartialFormer surpasses previous methods in both accuracy

and efficiency. For instance, in Semantic FPN, PartialFormer-B3 achieves 43.5 mIoU greater than Swin-T (Liu et al., 2021b) by 2.0%, and DAT-T (Xia et al., 2022) by 0.9%. In UperNet, PartialFormer-B3 outperforms Swin-T by 2.5%, DAT-T (Xia et al., 2022) by 1.5%, and Focal-T (Yang et al., 2021) by 1.2%, with smaller computational costs.

## 5.5 Conclusion

In this paper, we propose an efficient and general Partialformer for image classification and downstream tasks, addressing the computation redundancy of sparse attention such as spatial reduction attention and window attention. Thanks to novel partial attention, our model captures informative features from important regions and partially learns information from background regions, separately. The efficient information exchange between foreground and background sets is proposed to enhance modeling ability via learning abstract tokens. Extensive experiments verify the efficiency and effectiveness of PartialFormer across visual tasks.

## Chapter 6

# Conclusion

The works of this research aim to develop efficient vision Transformers for image classification and dense prediction tasks. Addressing issues in local and global self-attentions, this research seeks to mitigate computational bottlenecks in the Transformer encoders of Vision Transformer, computation redundancy in global self-attention, and limited receptive fields in window attention. Our methods have improved the efficiency of Vision Transformer and also enlarged the modeling ability of window self-attention. Each proposed method was discussed, analyzed, and evaluated comprehensively.

Firstly, this work discovered single-scale spatial interactions in hierarchical pyramid networks and integration of global self-attention into hybrid networks. To tackle these issues, this work proposed an Efficient Multi-scale Spatial interaction Network (EMSNNet) that takes advantage of convolution and self-attention operations. At each stage, convolution and local self-attention are performed on the features with small patch sizes to capture high-frequency components. Global self-attention is used for the feature with a large patch size to avoid high computational costs and also extract low-frequency components. With this strategy, EMSNNet has learned a wide range of frequencies from the input and enhanced generalization capacities. Extensive experiments verified the effectiveness of the proposed method on ImageNet-1K classification, MS-COCO object detection, and instance segmentation.

Secondly, we presented MAT Transformers that enlarge receptive fields and modeling capability of Window-based vision Transformers. In the literature, common methods improve information exchange across non-overlapped windows by using window shifting and sliding as additional operations along with window attention. Although these operations boost performance, they result in high memory access,



and is difficult to optimize and deploy the models in practical devices. Therefore, the MAT block was proposed to efficiently exchange information across windows and is easy to implement. Abstract tokens are added to each window and then, aggregate information from windows. Mixing learned abstract tokens can capture global context modeling. Extensive experiments showed that the MAT Transformer outperforms previous methods on object recognition tasks. Typically, MAT-2 achieved 79.0% Top-1 accuracy that surpasses PVTv2-B0 by 8.5% under similar latency on the CPU Intel(R) Xeon(R) Gold 5220R@2.20GHz. The MAT-4 surpassed Swin-T 1.8% mIoU while saving 30% GFLOPs.

Finally, this study proposed partial attention that learns spatial interactions more efficiently, by reducing computation redundancy in sparse attention such as spatial reduction attention and window attention. Token-to-token interactions are performed on a small part of image tokens (only important regions) and background information is squeezed to weaken the contribution of less important tokens to feature learning. Relevant tokens are grouped into the foreground set and the model learns informative features from this set via simple Mixed Multi-Head Self-Attention. Less important tokens are gathered into the background set and efficient Single-Query Attention performs interactions between single query and background tokens. Additionally, the efficient information exchange between foreground and background sets is proposed to enhance the modeling ability. Based on the design of partial attention, PartialFormer is proposed to achieve an efficient and general vision Transformer. In experimental results, PartialFormer attains better trade-offs between Top-1 accuracy and computational costs. For example, PartialFormer-B3 gets 83% Top-1 accuracy with 3.4 GFLOPs that surpasses state-of-the-art Swin-T by 1.7% accuracy while reducing 25% GFLOPs, DAT-T by 1.0% with only 75% GFLOPs, and Focal-T by 0.8% while saving 30 % GFLOPs. It verifies the proposed PartialFormer is efficient and effective.

## 6.1 Future Works

This research on object recognition mainly focuses on visual data. Integrating additional information from other modalities into feature learning can further improve

the performance and also evaluate the generalization ability of the proposed methods. For example, a pair of the image and text in CLIP (Radford et al., 2021) to the input of the network can fully describe objects and help to increase the usability of Vision Transformer in real-world applications. Therefore, using multimodal models is left for future work.

The proposed methods are trained and evaluated on GPU and CPU devices. Deploying EMSNet, MAT Transformer, and PartialFormer on mobile or practical devices is a promising direction for the future.

## Appendix A

# Publications

### A.1 Journal

1. Xuan-Thuy Vo and Kang-Hyun Jo, Accurate Bounding Box Prediction for Single-Shot Object Detection, *IEEE Transactions on Industrial Informatics*, Volume 18, Issue 9, pages: 5961-5971, 24 December 2021.
2. Xuan-Thuy Vo, and Kang-Hyun Jo, A Review on Anchor Assignment and Sampling Heuristics in Deep Learning-based Object Detection, *Neurocomputing*, Volume 506, Pages: 96-116, 16 July 2022.
3. Van-Dung Hoang, Xuan-Thuy Vo, and Kang-Hyun Jo, Categorical Weighting Domination for Imbalanced Classification with Skin Cancer in Intelligent Healthcare Systems, *IEEE Access*, 2023.

### A.2 Conference

1. Xuan-Thuy Vo, Duy-Linh Nguyen, Adri Priadana, and Kang-Hyun Jo, Hierarchical Vision Transformers with Shuffled Local Self-Attentions, *IWIS*, Korea, 2023.
2. Duy-Linh Nguyen, Xuan-Thuy Vo, Adri Priadana, and Kang-Hyun Jo, Simultaneous Person, Face, and Hand Detector Based on Improved YOLOv5, *IWIS*, Korea, 2023.
3. Adri Priadana, Muhamad Dwisnanto Putro, Jinsu An, Duy-Linh Nguyen, Xuan-Thuy Vo, and Kang-Hyun Jo, Gender Recognizer based on Human Face using CNN and Bottleneck Transformer Encoder, *IWIS*, Korea, 2023.

4. Duy-Linh Nguyen, Xuan-Thuy Vo, Adri Priadana, and Kang-Hyun Jo, Vehicle Detector Based on YOLOv5 Architecture for Traffic Management and Control Systems, IECON, Singapore, 2023.
5. Adri Priadana, Muhamad Dwisnanto Putro, Jinsu An, Duy-Linh Nguyen, Xuan-Thuy Vo, and Kang-Hyun Jo, Facial Attribute Recognition using Lightweight Multi-Label CNN-Transformer Architecture for Intelligent Advertising, IECON, Singapore, 2023.
6. Duy-Linh Nguyen, Xuan-Thuy Vo, Adri Priadana, and Kang-Hyun Jo, Car Detector Based on YOLOv5 for Parking Management, CITA, Vietnam, 2023.
7. Xuan-Thuy Vo, Jehwan Choi, Duy-Linh Nguyen, Adri Priadana, and Kang-Hyun Jo, Unifying Local and Global Fourier Features for Image Classification, ISIE, Finland, 2023.
8. Adri Priadana, Muhamad Dwisnanto Putro, Duy-Linh Nguyen, Xuan-Thuy Vo, and Kang-Hyun Jo, Age Group Recognizer based on Human Face Supporting Smart Digital Advertising Platforms, ISIE, Finland, 2023.
9. Adri Priadana, Muhamad Dwisnanto Putro, Duy-Linh Nguyen, Xuan-Thuy Vo, and Kang-Hyun Jo, Human Face Detector with Gender Identification by Split-Based Inception Block and Regulated Attention Module, IW-FCV, 2023.
10. Duy-Linh Nguyen, Xuan-Thuy Vo, Adri Priadana, and Kang-Hyun Jo, YOLO5PKLot: A Parking Lot Detection Network Based on Improved YOLOv5 for Smart Parking Management System, IW-FCV, 2023.
11. Xuan-Thuy Vo, Duy-Linh Nguyen, Adri Priadana, and Kang-Hyun Jo, Dynamic Circular Convolution for Image Classification, IW-FCV, Korea, 2023.
12. Tien-Dat Tran, Xuan-Thuy Vo, Duy-Linh Nguyen, and Kang-Hyun Jo, Combination of Deep Learner Network and Transformer for 3D Human Pose Estimation, ICCAS, Korea, 2022.
13. Adri Priadana, Muhamad Dwisnanto Putro, Xuan-Thuy Vo, and Kang-Hyun Jo, A Facial Gender Detector on CPU using Multi-dilated Convolution with Attention Modules, ICCAS, Korea, 2022.

14. Xuan-Thuy Vo, Tien-Dat Tran, Duy-Linh Nguyen, Adri Priadana, and Kang-Hyun Jo, Balancing Multiple Object Tracking Objectives based on Learned Weighting Factors, MAPR, Vietnam, 2022.
15. Adri Priadana, Muhamad Dwisnanto Putro, Xuan-Thuy Vo, and Kang-Hyun Jo, An Efficient Face-based Age Group Detector on a CPU using Two Perspective Convolution with Attention Modules, MAPR, Vietnam, 2022.
16. Duy-Linh Nguyen, Muhamad Dwisnanto Putro, Xuan-Thuy Vo, Tien-Dat Tran, and Kang-Hyun Jo, Fire Warning Based on Convolutional Neural Network and Inception Mechanism, MAPR, Vietnam, 2022.
17. Xuan-Thuy Vo, Tien-Dat Tran, Duy-Linh Nguyen, and Kang-Hyun Jo, A Study on Efficient Multi-task Networks for Multiple Object Tracking, IWIS, Korea, 2022.
18. Duy-Linh Nguyen, Muhamad Dwisnanto Putro, Xuan-Thuy Vo, Tien-Dat Tran, and Kang-Hyun Jo, Robust Hand Detection Based on Convolutional Neural Network and Attention Module, IWIS, Korea, 2022.
19. Tien-Dat Tran, Xuan-Thuy Vo, Duy-Linh Nguyen, and Kang-Hyun Jo, Efficient High-Resolution Network for Human Pose Estimation, IWIS, Korea, 2022.
20. Van-Dung Hoang, Xuan-Thuy Vo, Khac-Anh Phu, and Kang-Hyun Jo, Fusion of Segmentation and Classification for Improving Skin Disease Diagnosis, GTSD, Vietnam, 2022.
21. Xuan-Thuy Vo, Tien-Dat Tran, Duy-Linh Nguyen, Kang-Hyun Jo, Multi-level Feature Reweighting and Fusion for Instance Segmentation, INDIN, Australia, 2022
22. Tien-Dat Tran, Xuan-Thuy Vo, Duy-Linh Nguyen, and Kang-Hyun Jo, High-Resolution Network with Attention Module for Human Pose Estimation, ASCC, Korea, 2022.
23. Xuan-Thuy Vo, Van-Dung Hoang, Duy-Linh Nguyen, and Kang-Hyun Jo, Pedestrian Head Detection and Tracking via Global Vision Transformer, IW-FCV, Japan, 2022.

24. Duy-Linh Nguyen, Muhamad Dwisnanto Putro, Xuan-Thuy Vo, and Kang-Hyun Jo, Convolutional Neural Network Design for Eye Detection Under Low-Illumination, IW-FCV, Japan, 2022.
25. Duy-Linh Nguyen, Muhamad Dwisnanto Putro, Xuan-Thuy Vo, and Kang-Hyun Jo, Light-weight Convolutional Neural Network for Distracted Driver Classification, IECON, Canada, 2021.
26. Xuan-Thuy Vo, Tien-Dat Tran, Duy-Linh Nguyen, and Kang-Hyun Jo, Dynamic Multi-loss Weighting for Multiple People Tracking in Video Surveillance Systems, INDIN, Spain, 2021.
27. Duy-Linh Nguyen, Muhamad Dwisnanto Putro, Xuan-Thuy Vo, and Kang-Hyun Jo, Triple Detector based on Feature Pyramid Network for License Plate Detection and Recognition System in Unusual Conditions, ISIE, Japan, 2021.
28. Xuan-Thuy Vo, Tien-Dat Tran, Duy-Linh Nguyen, and Kang-Hyun Jo, Regression-Aware Classification Feature for Pedestrian Detection and Tracking in Video Surveillance Systems, ICIC, China, 2021.
29. Xuan-Thuy Vo, Tien-Dat Tran, Duy-Linh Nguyen, and Kang-Hyun Jo, Stair-step Feature Pyramid Networks for Object Detection, IW-FCV, Korea, 2021.
30. Tien-Dat Tran, Xuan-Thuy Vo, Duy-Linh Nguyen, and Kang-Hyun Jo, Efficient Spatial-Attention Module for Human Pose Estimation, IW-FCV, Korea, 2021.
31. Tien-Dat Tran, Xuan-Thuy Vo, Moahammad-Ashraf Russo, and Kang-Hyun Jo, Simple Fine-tuning Attention Modules for Human Pose Estimation, ICCCI, Vietnam, 2020.
32. Lihua Wen, Xuan-Thuy Vo, and Kang-Hyun Jo, 3D SaccadeNet: A Single-Shot 3D Object Detector for LiDAR Point Clouds, ICCAS, Korea, 2020.
33. Xuan-Thuy Vo, and Kang-Hyun Jo, Enhanced Feature Pyramid Networks by Feature Aggregation Module and Refinement Module, HSI, Japan, 2020.
34. Xuan-Thuy Vo, Lihua Wen, Tien-Dat Tran, and Kang-Hyun Jo, Bidirectional Non-local Networks for Object Detection, ICCCI, Vietnam, 2020.

# Bibliography

- Bochkovskiy, Alexey, Chien-Yao Wang, and Hong-Yuan Mark Liao (2020). “Yolov4: Optimal speed and accuracy of object detection”. In: *arXiv preprint arXiv:2004.10934*.
- Bolya, Daniel et al. (2019). “Yolact: Real-time instance segmentation”. In: *Proceedings of the IEEE/CVF international conference on computer vision*, pp. 9157–9166.
- Cai, Yuanhao et al. (2020). “Learning delicate local representations for multi-person pose estimation”. In: *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part III 16*. Springer, pp. 455–472.
- Cao, Yue et al. (2019). “Gcnet: Non-local networks meet squeeze-excitation networks and beyond”. In: *Proceedings of the IEEE/CVF international conference on computer vision workshops*, pp. 0–0.
- Carion, Nicolas et al. (2020). “End-to-end object detection with transformers”. In: *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part I 16*. Springer, pp. 213–229.
- Chen, Kai et al. (2019a). “Hybrid task cascade for instance segmentation”. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 4974–4983.
- Chen, Kai et al. (2019b). “MMDetection: Open mmlab detection toolbox and benchmark”. In: *arXiv preprint arXiv:1906.07155*.
- Chen, Qiang et al. (2022a). “Mixformer: Mixing features across windows and dimensions”. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 5249–5259.
- Chen, Yinpeng et al. (2022b). “Mobile-former: Bridging mobilenet and transformer”. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 5270–5279.
- Chu, Xiangxiang et al. (2021a). “Conditional positional encodings for vision transformers”. In: *arXiv preprint arXiv:2102.10882*.

- Chu, Xiangxiang et al. (2021b). "Twins: Revisiting the design of spatial attention in vision transformers". In: *Advances in Neural Information Processing Systems* 34, pp. 9355–9366.
- Chu, Xiangxiang et al. (2023). "Conditional Positional Encodings for Vision Transformers". In: *The Eleventh International Conference on Learning Representations*. URL: <https://openreview.net/forum?id=3KWnuT-R1bh>.
- Cubuk, Ekin D et al. (2020). "Randaugment: Practical automated data augmentation with a reduced search space". In: *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition workshops*, pp. 702–703.
- Dai, Xiyang et al. (2021a). "Dynamic DETR: End-to-End Object Detection With Dynamic Attention". In: *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 2988–2997.
- Dai, Zihang et al. (2021b). "Coatnet: Marrying convolution and attention for all data sizes". In: *Advances in Neural Information Processing Systems* 34, pp. 3965–3977.
- Deng, Huiqi et al. (2022). "Discovering and Explaining the Representation Bottleneck of DNNs". In: *International Conference on Learning Representations*.
- Deng, Jia et al. (2009). "Imagenet: A large-scale hierarchical image database". In: *2009 IEEE conference on computer vision and pattern recognition*. Ieee, pp. 248–255.
- Dong, Xiaoyi et al. (2022). "Cswin transformer: A general vision transformer backbone with cross-shaped windows". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 12124–12134.
- Dosovitskiy, Alexey et al. (2021). "An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale". In: *International Conference on Learning Representations*.
- Fang, Yuxin et al. (2021). "You Only Look at One Sequence: Rethinking Transformer in Vision through Object Detection". In: *arXiv preprint arXiv:2106.00666*.
- Fu, Cheng-Yang et al. (2017). "Dssd: Deconvolutional single shot detector". In: *arXiv preprint arXiv:1701.06659*.
- Gao, Li et al. (2022). "Doubly-Fused ViT: Fuse Information from Vision Transformer Doubly with Local Representation". In: *Computer Vision—ECCV 2022: 17th European Conference, Tel Aviv, Israel, October 23–27, 2022, Proceedings, Part XXIII*. Springer, pp. 744–761.



- Gao, Peng et al. (2021). “Fast Convergence of DETR With Spatially Modulated Co-Attention”. In: *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 3621–3630.
- Girshick, Ross (2015). “Fast r-cnn”. In: *Proceedings of the IEEE international conference on computer vision*, pp. 1440–1448.
- Girshick, Ross et al. (2014). “Rich feature hierarchies for accurate object detection and semantic segmentation”. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 580–587.
- Graham, Benjamin et al. (2021). “Levit: a vision transformer in convnet’s clothing for faster inference”. In: *Proceedings of the IEEE/CVF international conference on computer vision*, pp. 12259–12269.
- Grainger, Ryan et al. (2023). “PaCa-ViT: Learning Patch-to-Cluster Attention in Vision Transformers”. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 18568–18578.
- Guo, Jianyuan et al. (2022a). “Cmt: Convolutional neural networks meet vision transformers”. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 12175–12185.
- Guo, Meng-Hao et al. (2022b). “Visual attention network”. In: *arXiv preprint arXiv:2202.09741*.
- Gupta, Ankit et al. (2021). “Memory-efficient Transformers via Top-k Attention”. In: *Proceedings of SustaiNLP: Workshop on Simple and Efficient Natural Language Processing*. Association for Computational Linguistics.
- Han, Kai et al. (2021). “Transformer in Transformer”. In: *Advances in Neural Information Processing Systems*. Ed. by A. Beygelzimer et al.
- Hassani, Ali et al. (2023). “Neighborhood attention transformer”. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 6185–6194.
- Hatamizadeh, Ali et al. (2023a). “Global context vision transformers”. In: *International conference on machine learning*. PMLR.
- Hatamizadeh, Ali et al. (2023b). “Global context vision transformers”. In: *International Conference on Machine Learning*. PMLR, pp. 12633–12646.
- He, Kaiming et al. (2016). “Deep residual learning for image recognition”. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 770–778.

- He, Kaiming et al. (2017a). "Mask r-cnn". In: *Proceedings of the IEEE international conference on computer vision*, pp. 2961–2969.
- (2017b). "Mask r-cnn". In: *Proceedings of the IEEE international conference on computer vision*, pp. 2961–2969.
- Heo, Byeongho et al. (2021). "Rethinking spatial dimensions of vision transformers". In: *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 11936–11945.
- Howard, Andrew et al. (2019). "Searching for mobilenetv3". In: *Proceedings of the IEEE/CVF international conference on computer vision*, pp. 1314–1324.
- Howard, Andrew G et al. (2017). "Mobilenets: Efficient convolutional neural networks for mobile vision applications". In: *arXiv preprint arXiv:1704.04861*.
- Hu, Jie, Li Shen, and Gang Sun (2018). "Squeeze-and-excitation networks". In: *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 7132–7141.
- Huang, Gao et al. (2016). "Deep networks with stochastic depth". In: *Computer Vision—ECCV 2016: 14th European Conference, Amsterdam, The Netherlands, October 11–14, 2016, Proceedings, Part IV 14*. Springer, pp. 646–661.
- Huang, Gao et al. (2017). "Densely connected convolutional networks". In: *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 4700–4708.
- Huang, Zilong et al. (2021). "Shuffle transformer: Rethinking spatial shuffle for vision transformer". In: *arXiv preprint arXiv:2106.03650*.
- Jin, Ruibing et al. (2023). "An adaptive and dynamical neural network for machine remaining useful life prediction". In: *IEEE Transactions on Industrial Informatics*.
- Jing, Tao, Qing-Hao Meng, and Hui-Rang Hou (2023). "SmokeSegger: A Transformer-CNN coupled model for urban scene smoke segmentation". In: *IEEE Transactions on Industrial Informatics*.
- Kirillov, Alexander et al. (2019). "Panoptic feature pyramid networks". In: *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 6399–6408.
- Kong, Tao et al. (2020). "Foveabox: Beyond anchor-based object detection". In: *IEEE Transactions on Image Processing* 29, pp. 7389–7398.
- Krizhevsky, Alex, Ilya Sutskever, and Geoffrey E Hinton (2017). "Imagenet classification with deep convolutional neural networks". In: *Communications of the ACM* 60.6, pp. 84–90.

- Li, Jiashi et al. (2022a). "Next-vit: Next generation vision transformer for efficient deployment in realistic industrial scenarios". In: *arXiv preprint arXiv:2207.05501*.
- Li, Yanyu et al. (2022b). "EfficientFormer: Vision Transformers at MobileNet Speed". In: *Advances in Neural Information Processing Systems*. Ed. by Alice H. Oh et al. URL: <https://openreview.net/forum?id=NXHXoYMLIG>.
- Li, Yanyu et al. (2023). "Rethinking Vision Transformers for MobileNet Size and Speed". In: *Proceedings of the IEEE international conference on computer vision*.
- Lin, Tsung-Yi et al. (2014). "Microsoft coco: Common objects in context". In: *Computer Vision—ECCV 2014: 13th European Conference, Zurich, Switzerland, September 6–12, 2014, Proceedings, Part V 13*. Springer, pp. 740–755.
- Lin, Tsung-Yi et al. (2017a). "Feature pyramid networks for object detection". In: *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 2117–2125.
- Lin, Tsung-Yi et al. (2017b). "Focal loss for dense object detection". In: *Proceedings of the IEEE international conference on computer vision*, pp. 2980–2988.
- Lin, Weifeng et al. (2023). "Scale-Aware Modulation Meet Transformer". In: *Proceedings of the IEEE/CVF international conference on computer vision*.
- Liu, Haotian et al. (2021a). "Yolactedge: Real-time instance segmentation on the edge". In: *2021 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, pp. 9579–9585.
- Liu, Wei et al. (2016). "Ssd: Single shot multibox detector". In: *Computer Vision—ECCV 2016: 14th European Conference, Amsterdam, The Netherlands, October 11–14, 2016, Proceedings, Part I 14*. Springer, pp. 21–37.
- Liu, Xinyu et al. (2023). "EfficientViT: Memory Efficient Vision Transformer with Cascaded Group Attention". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 14420–14430.
- Liu, Ze et al. (2021b). "Swin transformer: Hierarchical vision transformer using shifted windows". In: *Proceedings of the IEEE/CVF international conference on computer vision*, pp. 10012–10022.
- Liu, Ze et al. (2022a). "Swin transformer v2: Scaling up capacity and resolution". In: *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 12009–12019.
- Liu, Zhuang et al. (2022b). "A convnet for the 2020s". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 11976–11986.

- Liu, Zhuang et al. (2022c). “A convnet for the 2020s”. In: *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 11976–11986.
- Loshchilov, Ilya and Frank Hutter (2019). “Decoupled Weight Decay Regularization”. In: *International Conference on Learning Representations*. URL: <https://openreview.net/forum?id=Bkg6RiCqY7>.
- Luo, Jie et al. (2023). “A Dual-Branch Spatio-Temporal-Spectral Transformer Feature Fusion Network for EEG-Based Visual Recognition”. In: *IEEE Transactions on Industrial Informatics*.
- Mehta, Sachin and Mohammad Rastegari (2022). “MobileViT: Light-weight, General-purpose, and Mobile-friendly Vision Transformer”. In: *International Conference on Learning Representations*.
- (2023). “Separable Self-attention for Mobile Vision Transformers”. In: *Transactions on Machine Learning Research*. ISSN: 2835-8856.
- Meng, Depu et al. (2021). “Conditional DETR for Fast Training Convergence”. In: *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 3651–3660.
- Min, Juhong et al. (2022). “Peripheral Vision Transformer”. In: *Advances in Neural Information Processing Systems*. Ed. by Alice H. Oh et al. URL: <https://openreview.net/forum?id=nE8IJLT7nW->.
- Pan, Junting et al. (2022a). “Edgevits: Competing light-weight cnns on mobile devices with vision transformers”. In: *Computer Vision—ECCV 2022: 17th European Conference, Tel Aviv, Israel, October 23–27, 2022, Proceedings, Part XI*. Springer, pp. 294–311.
- (2022b). “Edgevits: Competing light-weight cnns on mobile devices with vision transformers”. In: *Computer Vision—ECCV 2022: 17th European Conference, Tel Aviv, Israel, October 23–27, 2022, Proceedings, Part XI*. Springer, pp. 294–311.
- Pan, Xuran et al. (2023). “Slide-Transformer: Hierarchical Vision Transformer with Local Self-Attention”. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 2082–2091.
- Pan, Zizheng, Jianfei Cai, and Bohan Zhuang (2022). “Fast Vision Transformers with HiLo Attention”. In: *Advances in Neural Information Processing Systems*. Ed. by Alice H. Oh et al. URL: <https://openreview.net/forum?id=Pyd6Rh9r10T>.

- Pan, Zizheng et al. (2022c). "Less is more: Pay less attention in vision transformers". In: *Proceedings of the AAAI Conference on Artificial Intelligence*. Vol. 36. 2, pp. 2035–2043.
- Park, Namuk and Songkuk Kim (2022). "How Do Vision Transformers Work?" In: *International Conference on Learning Representations*. URL: <https://openreview.net/forum?id=D78Go4hVcx0>.
- Polyak, Boris T and Anatoli B Juditsky (1992). "Acceleration of stochastic approximation by averaging". In: *SIAM journal on control and optimization* 30.4, pp. 838–855.
- Radford, Alec et al. (2021). "Learning transferable visual models from natural language supervision". In: *International conference on machine learning*. PMLR, pp. 8748–8763.
- Redmon, Joseph and Ali Farhadi (2018). "Yolov3: An incremental improvement". In: *arXiv preprint arXiv:1804.02767*.
- Redmon, Joseph et al. (2016). "You only look once: Unified, real-time object detection". In: *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 779–788.
- Ren, Shaoqing et al. (2015). "Faster r-cnn: Towards real-time object detection with region proposal networks". In: *Advances in neural information processing systems* 28, pp. 91–99.
- Ren, Sucheng et al. (2022). "Shunted self-attention via multi-scale token aggregation". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 10853–10862.
- Russakovsky, Olga et al. (2015). "Imagenet large scale visual recognition challenge". In: *International journal of computer vision* 115, pp. 211–252.
- Sandler, Mark et al. (2018). "Mobilenetv2: Inverted residuals and linear bottlenecks". In: *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 4510–4520.
- Shaker, Abdelrahman et al. (2023). "SwiftFormer: Efficient Additive Attention for Transformer-based Real-time Mobile Vision Applications". In: *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 17425–17436.
- Si, Chenyang et al. (2022). "Inception Transformer". In: *Advances in Neural Information Processing Systems*. Ed. by Alice H. Oh et al. URL: <https://openreview.net/forum?id=qf12cWVSksq>.

- Simonyan, Karen and Andrew Zisserman (2014). “Very deep convolutional networks for large-scale image recognition”. In: *arXiv preprint arXiv:1409.1556*.
- Sun, Chen et al. (2017a). “Revisiting unreasonable effectiveness of data in deep learning era”. In: *Proceedings of the IEEE international conference on computer vision*, pp. 843–852.
- (2017b). “Revisiting unreasonable effectiveness of data in deep learning era”. In: *Proceedings of the IEEE international conference on computer vision*, pp. 843–852.
- Sun, Mingyi et al. (2023). “Attention-Rectified and Texture-Enhanced Cross-Attention Transformer Feature Fusion Network for Facial Expression Recognition”. In: *IEEE Transactions on Industrial Informatics*.
- Sun, Peize et al. (2021a). “Sparse r-cnn: End-to-end object detection with learnable proposals”. In: *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 14454–14463.
- Sun, Zhiqing et al. (2021b). “Rethinking transformer-based set prediction for object detection”. In: *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 3611–3620.
- Szegedy, Christian et al. (2015). “Going deeper with convolutions”. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 1–9.
- Tan, Mingxing and Quoc Le (2019). “Efficientnet: Rethinking model scaling for convolutional neural networks”. In: *International conference on machine learning*. PMLR, pp. 6105–6114.
- Tang, Shitao et al. (2022). “Quadtree Attention for Vision Transformers”. In: *International Conference on Learning Representations*. URL: [https://openreview.net/forum?id=fR-EnKWL\\_Zb](https://openreview.net/forum?id=fR-EnKWL_Zb).
- Tian, Zhi et al. (2019). “Fcos: Fully convolutional one-stage object detection”. In: *Proceedings of the IEEE/CVF international conference on computer vision*, pp. 9627–9636.
- Touvron, Hugo et al. (2021). “Training data-efficient image transformers & distillation through attention”. In: *International conference on machine learning*. PMLR, pp. 10347–10357.
- Tu, Zhengzhong et al. (2022). “Maxvit: Multi-axis vision transformer”. In: *Computer Vision—ECCV 2022: 17th European Conference, Tel Aviv, Israel, October 23–27, 2022, Proceedings, Part XXIV*. Springer, pp. 459–479.
- Vaswani, Ashish et al. (2017). “Attention is all you need”. In: *Advances in neural information processing systems* 30.

- Vaswani, Ashish et al. (2021). “Scaling local self-attention for parameter efficient visual backbones”. In: *CVPR*, pp. 12894–12904.
- Wadekar, Shakti N and Abhishek Chaurasia (2022). “Mobilevitv3: Mobile-friendly vision transformer with simple and effective fusion of local, global and input features”. In: *arXiv preprint arXiv:2209.15159*.
- Wang, Chien-Yao, Alexey Bochkovskiy, and Hong-Yuan Mark Liao (2021). “Scaled-yolov4: Scaling cross stage partial network”. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 13029–13038.
- Wang, Jiaqi et al. (2019). “Region proposal by guided anchoring”. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 2965–2974.
- Wang, Pichao et al. (2022a). “Kvt: k-nn attention for boosting vision transformers”. In: *European conference on computer vision*. Springer, pp. 285–302.
- Wang, Wenhai et al. (2021a). “Pyramid vision transformer: A versatile backbone for dense prediction without convolutions”. In: *Proceedings of the IEEE/CVF international conference on computer vision*, pp. 568–578.
- (2022b). “Pvt v2: Improved baselines with pyramid vision transformer”. In: *Computational Visual Media* 8.3, pp. 415–424.
- Wang, Wenhai et al. (2023). “Internimage: Exploring large-scale vision foundation models with deformable convolutions”. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 14408–14419.
- Wang, Wenxiao et al. (2022c). “CrossFormer: A Versatile Vision Transformer Hinging on Cross-scale Attention”. In: *International Conference on Learning Representations, ICLR*. URL: [https://openreview.net/forum?id=\\_PHymLIxuI](https://openreview.net/forum?id=_PHymLIxuI).
- Wang, Xinlong et al. (2020a). “Solo: Segmenting objects by locations”. In: *European Conference on Computer Vision*. Springer, pp. 649–665.
- Wang, Xinlong et al. (2020b). “Solov2: Dynamic and fast instance segmentation”. In: *Advances in Neural information processing systems* 33, pp. 17721–17732.
- Wang, Yingming et al. (2021b). “Anchor DETR: Query Design for Transformer-Based Detector”. In: *arXiv preprint arXiv:2109.07107*.
- Wightman, Ross (2019). *PyTorch Image Models*. <https://github.com/rwightman/pytorch-image-models>. DOI: 10.5281/zenodo.4414861.

- Woo, Sanghyun et al. (2023). "Convnext v2: Co-designing and scaling convnets with masked autoencoders". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 16133–16142.
- Wu, Haiping et al. (2021). "Cvt: Introducing convolutions to vision transformers". In: *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 22–31.
- Wu, Sitong et al. (2022). "Pale transformer: A general vision transformer backbone with pale-shaped attention". In: *Proceedings of the AAAI Conference on Artificial Intelligence*. Vol. 36. 3, pp. 2731–2739.
- Xia, Zhuofan et al. (2022). "Vision transformer with deformable attention". In: *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 4794–4803.
- Xiao, Bin, Haiping Wu, and Yichen Wei (2018). "Simple baselines for human pose estimation and tracking". In: *Proceedings of the European conference on computer vision (ECCV)*, pp. 466–481.
- Xiao, Tete et al. (2018). "Unified perceptual parsing for scene understanding". In: *Proceedings of the European conference on computer vision (ECCV)*, pp. 418–434.
- Xie, Enze et al. (2020). "Polarmask: Single shot instance segmentation with polar representation". In: *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 12193–12202.
- Yang, Chenglin et al. (2022a). "Lite vision transformer with enhanced self-attention". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 11998–12008.
- Yang, Chenglin et al. (2023). "MOAT: Alternating Mobile Convolution and Attention Brings Strong Vision Models". In: *The Eleventh International Conference on Learning Representations*. URL: <https://openreview.net/forum?id=HOHG1jkkQFN>.
- Yang, Jianwei et al. (2021). "Focal Attention for Long-Range Interactions in Vision Transformers". In: *Advances in Neural Information Processing Systems*. Ed. by A. Beygelzimer et al. URL: <https://openreview.net/forum?id=2zCRcTafea>.
- Yang, Jianwei et al. (2022b). "Focal modulation networks". In: *Advances in Neural Information Processing Systems* 35, pp. 4203–4217.
- Yao, Zhuyu et al. (2021). "Efficient DETR: Improving End-to-End Object Detector with Dense Prior". In: *arXiv preprint arXiv:2104.01318*.



- Yu, Weihao et al. (2022). “Metaformer is actually what you need for vision”. In: *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 10819–10829.
- Yun, Sangdoon et al. (2019). “Cutmix: Regularization strategy to train strong classifiers with localizable features”. In: *Proceedings of the IEEE/CVF international conference on computer vision*, pp. 6023–6032.
- Zhang, Haokui, Wenzhe Hu, and Xiaoyu Wang (2022). “Parc-net: Position aware circular convolution with merits from convnets and transformer”. In: *Computer Vision—ECCV 2022: 17th European Conference, Tel Aviv, Israel, October 23–27, 2022, Proceedings, Part XXVI*. Springer, pp. 613–630.
- Zhang, Hongyi et al. (2017). “mixup: Beyond empirical risk minimization”. In: *arXiv preprint arXiv:1710.09412*.
- Zhang, Jiangning et al. (2023). *Rethinking Mobile Block for Efficient Attention-based Models*. arXiv: 2301.01146 [cs.CV].
- Zhang, Qinglong and Yu-Bin Yang (2021). “Rest: An efficient transformer for visual recognition”. In: *Advances in Neural Information Processing Systems 34*, pp. 15475–15485.
- (2022). “Rest v2: simpler, faster and stronger”. In: *Advances in Neural Information Processing Systems 35*, pp. 36440–36452.
- Zhang, Shifeng et al. (2020). “Bridging the gap between anchor-based and anchor-free detection via adaptive training sample selection”. In: *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 9759–9768.
- Zhang, Wenqiang et al. (2022). “TopFormer: Token pyramid transformer for mobile semantic segmentation”. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 12083–12093.
- Zhou, Bolei et al. (2019). “Semantic understanding of scenes through the ade20k dataset”. In: *International Journal of Computer Vision 127*, pp. 302–321.
- Zhu, Chenchen, Yihui He, and Marios Savvides (2019). “Feature selective anchor-free module for single-shot object detection”. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 840–849.
- Zhu, Lei et al. (2023). “BiFormer: Vision Transformer with Bi-Level Routing Attention”. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 10323–10333.

- Zhu, Xizhou et al. (2019). “Deformable convnets v2: More deformable, better results”. In: *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 9308–9316.
- Zhu, Xizhou et al. (2021). “Deformable detr: Deformable transformers for end-to-end object detection”. In: *International Conference on Learning Representations*.