



EATFormer: Improving Vision Transformer Inspired by Evolutionary Algorithm

Jiangning Zhang¹ · Xiangtai Li² · Yabiao Wang³ · Chengjie Wang³ · Yibo Yang² · Yong Liu¹ · Dacheng Tao⁴

Received: 18 June 2022 / Accepted: 12 February 2024 / Published online: 2 April 2024

© The Author(s), under exclusive licence to Springer Science+Business Media, LLC, part of Springer Nature 2024

Abstract

Motivated by biological evolution, this paper explains the rationality of Vision Transformer by analogy with the proven practical evolutionary algorithm (EA) and derives that both have consistent mathematical formulation. Then inspired by effective EA variants, we propose a novel pyramid EATFormer backbone that only contains the proposed EA-based transformer (EAT) block, which consists of three residual parts, i.e., Multi-scale region aggregation, global and local interaction, and feed-forward network modules, to model multi-scale, interactive, and individual information separately. Moreover, we design a task-related head docked with transformer backbone to complete final information fusion more flexibly and *improve* a modulated deformable MSA to dynamically model irregular locations. Massive quantitative and quantitative experiments on image classification, downstream tasks, and explanatory experiments demonstrate the effectiveness and superiority of our approach over state-of-the-art methods. E.g., our Mobile (1.8 M), Tiny (6.1 M), Small (24.3 M), and Base (49.0 M) models achieve 69.4, 78.4, 83.1, and 83.9 Top-1 only trained on ImageNet-1K with naive training recipe; EATFormer-Tiny/Small/Base armed Mask-R-CNN obtain 45.4/47.4/49.0 box AP and 41.4/42.9/44.2 mask AP on COCO detection, surpassing contemporary MPViT-T, Swin-T, and Swin-S by 0.6/1.4/0.5 box AP and 0.4/1.3/0.9 mask AP separately with less FLOPs; Our EATFormer-Small/Base achieve 47.3/49.3 mIoU on ADE20K by Upernet that exceeds Swin-T/S by 2.8/1.7. Code is available at <https://github.com/zhangzn/EATFormer>.

Keywords Computer vision · Vision transformer · Evolutionary algorithm · Image classification · Object detection · Image segmentation

Communicated by Suha Kwak.

Jiangning Zhang and Xiangtai Li have contributed equally as first authors.

✉ Yong Liu
yongliu@iipc.zju.edu.cn

Jiangning Zhang
186368@zju.edu.cn

Xiangtai Li
lxtpku@pku.edu.cn

Yabiao Wang
caseywang@tencent.com

Chengjie Wang
jasoncjwang@tencent.com

Yibo Yang
ibo@pku.edu.cn

Dacheng Tao
dacheng.tao@gmail.com

1 Introduction

Since Vaswani et al. (2017) introduce the Transformer that achieves outstanding success in the machine translation task, many improvements have been made over this structure (Choromanski et al., 2021; Kitaev et al., 2020; Devlin et al., 2019). Subsequently, Dosovitskiy et al. (2021) firstly introduce Transformer to the computer vision field and propose a novel ViT model that successfully sparks a new wave of research besides conventional CNN-based vision models.

¹ Institute of Cyber-Systems and Control, Advanced Perception on Robotics and Intelligent Learning Lab (APRIL), Zhejiang University, Hangzhou, China

² School of Artificial Intelligence, Key Laboratory of Machine Perception (MOE), Peking University, Beijing, China

³ Youtu Lab, Tencent, Shanghai, China

⁴ School of Computer Science, Faculty of Engineering, The University of Sydney, Darlingtown, NSW 2008, Australia

Recently, many excellent vision transformer models (Wang et al., 2021a; Liu et al., 2021; Ali et al., 2021; Yu et al., 2022; Hassani et al., 2023; Lee et al., 2022; Li et al., 2022) have been proposed and have achieved great success in the field of many vision tasks. Currently, many attempts have been made to explain and improve the Transformer structure from different perspectives (Goyal & Bengio, 2022; Pan et al., 2022; Cordonnier et al., 2020; Raghu et al., 2021; Min et al., 2022; d'Ascoli et al. 2021a; Bhojanapalli et al., 2021; Liu et al., 2022a; Hao et al., 2021), while continuing research is still needed. Most current models generally migrate the structural design of CNN, and they are experimentally conducted to verify the effectiveness of modules or improvements, which lacks explanations about *why improved Transformer approaches work* (Hao et al., 2021; Yu et al., 2022; Dong et al., 2021a).

Inspired by biological population evolution, we explain the rationality of Transformer by analogy with the proven effective, stable, and robust Evolutionary Algorithm (EA) in this article, which has been widely used in many practical applications. We observe that the procedure of the Transformer (*abbr.*, TR) has similar attributes to the naive EA through analogical analysis in Fig. 1a-TR and b-EA:

- (1) In terms of data format, TR processes patch embeddings while EA evolves individuals, both of them have the same data formats and necessary initialization.
- (2) In terms of optimization objective, TR aims to obtain an optimal vector representation that fuses global information through multiple layers (denoted as \mathbf{x}_N in Fig. 1), while EA focuses on getting the best individual globally through multiple iterations.
- (3) In terms of component, Multi-head Self-Attention (MSA) in TR aims to enrich patch embeddings through global information communication densely, while crossover operation in EA plays the role of interacting global individuals sparsely. Also, Feed-Forward Network (FFN) in TR enhances every single embedding for all spatial positions, which is similar to Mutation in EA that evolves each individual of the whole population.
- (4) Furthermore, we deduce the mathematical characterization of crossover and mutation operators in EA (c.f., Eqs. 5, 8) and find that they *have the same mathematical formulations* as MSA and FFN in TR (c.f., Eqs. 6, 9), respectively.

In addition to the above basic analogies between naive Transformer and EA, we explore further to improve the current vision transformer by leveraging other domain knowledge of EA variants. Without losing generality, we only study the widely used and effective EA methods that could inspire us to improve Transformer. They can be mainly divided into the following categories:

- (1) *Global and Local populations* inspired *simultaneously global and local modeling*. In contrast to naive EA that only models global interaction, local search-based EA variants focus on finding a better individual in its neighborhood (Kolen & Pesch, 1994; Land, 1998; García-Martínez & Lozano, 2008) that is more efficient without associating the global search space. Furthermore, Moscato (1989) firstly propose the Memetic Evolutionary Algorithm (MEA) that introduces a local search process for converging high-quality solutions faster than conventional evolutionary counterparts, and intuitive illustration can be viewed in Fig. 1c. For a particular individual (i.e., center sheep with red background), naive EA only contains **● Global Population** concept, while **● Local Population** idea enables the model to focus on more relevant individuals. Inspired by those EA variants, we revisit the global MSA part and *improve* a novel *Global and Local Interaction* (GLI) module, which is designed as a parallel structure that employs an extra local operation beside the global operation, i.e., introducing inductive bias and locality in MSA. The former is used to mine more relevant local information, while the latter aims to model global cue interactions. Considering that the spatial relationship among real individuals will not be as horizontal and vertical as the image, we further propose a Modulated Deformable MSA (MD-MSA) to dynamically model irregular locations, which could focus on more informative reorganizational regions.
- (2) *Multi-population* inspired *multi-scale information aggregation*. Some works (Chen & Kang, 2005; Li et al., 2021b) introduce multi-population evolutionary algorithm to solve the optimization problems, which adopts different searching regions to more efficiently enhance the diversity of individuals and can obtain a better model performance significantly. As shown in Fig. 1c, **◆ Long-Distance Population** could supplement more diverse and richer cues, while **◆ Short-Distance Population** focuses on providing general evolutionary features. Analogously, this idea inspires us to design a *Multi-Scale Region Aggregation* (MSRA) module that aggregates information from different receptive fields for vision transformer, which could integrate more expressive features from different resolutions before feeding them into the next module.
- (3) *Dynamic population* inspired *pyramid architecture design*. The works (Brest et al., 2008; 2010; Shi et al. 2014) investigate jDEdynNP-F algorithm with a dynamic population reduction scheme that significantly improves the effectiveness and accelerates the convergence of the model, which is similar to pyramid-alike improvements of some current vision transformers (Wang et al., 2022b; Chu et al., 2021; Liu et al., 2022b; Hassani et al., 2023;

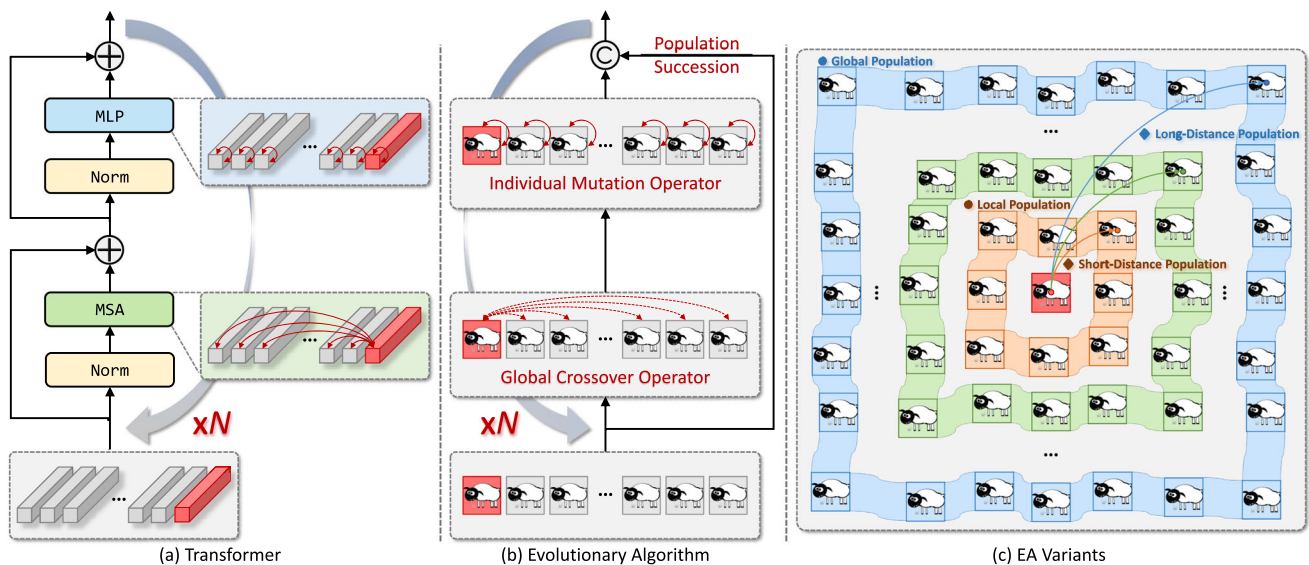


Fig. 1 Comparisons of structural analogousness between **a** Transformer module and **b** evolutionary algorithm, where they have analogous concepts of (1) individual definition (Token Embedding vs. Individual), (2) global information interaction (MSA vs. Crossover), (3)

individual feature enhancement (FFN vs. Mutation), (4) feature inheritance (Skip Connection vs. Population Succession), etc. **c** Intuitive illustration of some EA variants

Yu et al., 2022). Analogously, we extend our previous columnar-like work (Zhang et al., 2021) to a pyramid structure like PVT (Wang et al., 2021a), which significantly boosts the performance for many vision tasks.

- (4) *Self-adapted parameters inspired weighted operation mixing.* Brest et al. (2006) propose an adaptation mechanism to control different optimization processes for better results, and some memetic EAs (Moscato, 1989; Hart et al., 2005; Kumar et al., 2014) own a similar concept of search intensity to balance the global and local calculation. This encourages us to learn appropriate weights for different operations, which can increase the performance and be more interpretable.
- (5) *Multi-objective EA inspired task-related feature merging.* Current TR-based vision models would initialize different tokens for different tasks (Touvron et al., 2021a) (e.g., classification and distillation) or use the pooling operation to obtain global representation (Liu et al., 2021). However, both manners suffer from potentially incompatibility: the former treats the task token and image patches coequally that is unreasonable, and the calculation of each layer will slightly increase the amount of calculation ($O(n^2)$ to $O((n+1)^2)$), while the latter uses only one pooling result for multiple tasks that could potentially damage the model accuracy. Inspired by multi-objective EAs (Coello & Lamont, 2004; Khare et al., 2003; Toffolo et al., 2003) that find a set of solutions for different targets, we design a *Task-Related Head* (TRH) docked with transformer backbone to com-

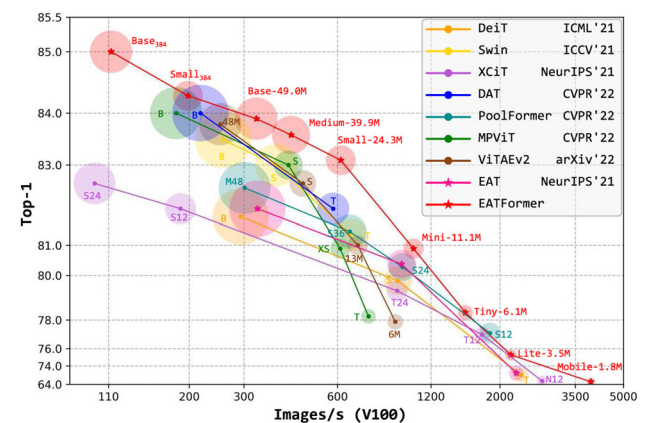


Fig. 2 Paradigm of the proposed basic EAT Block, which contains three $y = f(x) + x$ residuals to model: **a** multi-scale information aggregation, **b** feature interactions among tokens, and **c** individual enhancement

plete final information fusion, which is elegant and flexible for different tasks learning (2).

Based on the above analyses, we improve our columnar EAT model (Zhang et al., 2021) to a pyramid EA-inspired Transformer (EATFormer) that achieves a new SOTA result. Figure 2 illustrates intuitive comparisons with SOTAs under GPU throughput, Top-1, and the number of parameters evaluation indexes, where our smallest EATFormer-Mobile obtains 69.4 Top-1 with 3,926 throughput under one V100 GPU, and the EATFormer-Base achieves 83.9 Top-1 with only 49.0M parameters. Specifically, we make the following

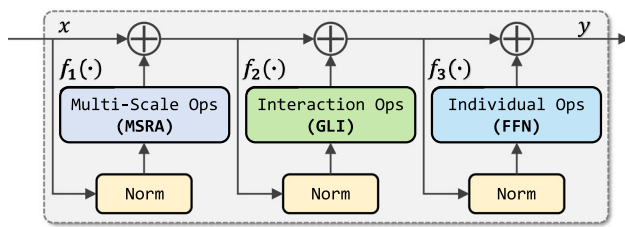


Fig. 3 Comparison with SOTAs in terms of Top-1 vs. GPU throughput. All models are trained only with ImageNet-1K (Deng et al., 2009) dataset in 224×224 , and the radius represents the relative number of parameters

four contributions compared with the previous conference work:

- In theory, we enrich evolutionary explanations for *the rationality of Vision Transformer* and derive a consistent mathematical formulation with evolutionary algorithm.
- On framework, we propose a novel basic *EA-based Transformer* (EAT) block (shown in Fig. 3) that consists of three residual parts to model multi-scale, interactive, and individual information, respectively, which is stacked to form our proposed pyramid EATFormer.
- For method, inspired by effective EA variants, we analogously design: (1) Global and Local Interaction module, (2) Multi-Scale Region Aggregation module, (3) Task-Related Head module, and (4) Modulated Deformable MSA module to improve effectiveness and usability of our EATFormer.
- Massive experiments on classification, object detection, and semantic segmentation tasks demonstrate the superiority and efficiency of our approach, while ablation and explanatory experiments further prove the efficacy of EATFormer and its components.

2 Related Work

2.1 Evolution Algorithms

Evolution algorithm (EA) is a subset of evolutionary computation in computational intelligence that belongs to modern heuristics, and it serves as an essential umbrella term to describe population-based optimization and search techniques in the last 50 years (Sloss & Gustafson, 2020; Vikhar, 2016; Bartz-Beielstein et al., 2014). Inspired by biological evolution, general EAs mainly contain reproduction, crossover, mutation, and selection steps, which have been proven effective and stable in many application scenarios (Moscato, 1989; Hart et al., 2005), and a series of improved EA approaches have been advanced in succession. Differential Evolution (DE) developed in 1995 (Storn & Price,

1997) is arguably one of the most competitive improved EA that significantly advances the global modeling capability (Das & Suganthan, 2010; Padhye et al., 2013). The core idea of DE is introducing a complete differential concept to the conventional EA, which differentiates and scales two individuals in the same population and interacts with the third individual to generate a new individual. In contrast to the category mentioned above, local search-based EAs aim to find a solution that is as good as or better than all other solutions in its neighborhood (Kolen & Pesch, 1994; Land, 1998; García-Martínez & Lozano, 2008). This thought is more efficient than global search in that a solution can quickly be verified as a local optimum without associating the global search space. However, the locality-aware operation will restrict the ability of global modeling that could lead to suboptimal results in some scenarios, so some researchers attempt to fuse both above modeling manners. Moscato (1989) firstly propose the Memetic Evolutionary Algorithm (MEA) in 1989 that applies a local search process to refine solutions for hard problems, which could converge to high-quality solutions more efficiently than conventional evolutionary counterparts. In detail, this variant is a particular global–local search hybrid: the global character is given by the traditional EA, while the local aspect is mainly performed through constructive methods and intelligent local search heuristics (Hart et al., 2005). Analogously, some later works (Chen & Kang, 2005; Li et al., 2021b) introduce a multi-population evolutionary algorithm to solve the constrained function optimization problems relatively efficiently, which adopts different searching regions to enhance the diversity of individuals that improves the model ability dramatically. This strategy inspires us to design a basic feature extraction module for vision transformer: whether a similar multi-scale manner can be adopted to enhance model expressiveness. Furthermore, Brest et al. (2006) propose an adaptation mechanism on the control parameters CR and F for crossover and mutation operations associated with DE, where adapted parameters are applied to different optimization processes for obtaining better results. Remarkably, those MEAs (Moscato, 1989; Hart et al., 2005; Kumar et al., 2014) mentioned above own a similar concept of search intensity to balance the global and local calculation. Subsequent work (Brest et al., 2008) investigates jDEdynNP-F algorithm with a dynamic population reduction scheme, where the population size of the next generation is equal to half the previous population size. This strategy significantly improves the effectiveness and accelerates the convergence of the model that is consistently illustrated by works (Brest et al., 2010; Shi et al., 2014). Furthermore, some literatures (Motter, 1993; Bhowmik et al., 2021; Felleman & Van Essen, 1991) suggest that there are hierarchical structures of V1, V2, V4, and inferotemporal cortex in the evolutionary brain, which have ordered interconnection among them in the form of both feed-forward

and feedback connections. Moreover, researchers (Coello & Lamont, 2004; Khare et al., 2003; Toffolo et al., 2003) study multi-objective EAs to find optimal trade-offs to get a set of solutions for different targets.

Inspired by the aforementioned EA variants that introduce various and valid concepts for optimization, we explain and improve the naive transformer structure by conceptual analogy in the paper, where a novel and potent EATFormer with pyramid architecture, multi-scale region aggregation, and global–local modeling is hand-designed. Furthermore, a plug-and-play task-related head module is developed to solve different targets separately and improve model performance.

2.2 Vision Transformers

Since Transformer structure achieves significant progress for machine translation task (Vaswani et al., 2017), many improved language models (Peters et al., 2018; Devlin et al., 2019; Radford et al., 2018, 2019; Brown et al., 2020) are proposed and obtain great achievements, and some later works (He et al., 2020; Wang et al., 2020b; Choromanski et al., 2021; Kitaev et al., 2020; Bello, 2021; Wang et al., 2021b) advance the basic transformer module for better efficiency. Inspired by the success of Transformer in NLP and the rapid improvement of computing power, Dosovitskiy et al. (2021) propose a novel ViT that firstly introduces the transformer to vision classification and sparks a new wave of research besides conventional CNN-based vision models. Subsequently, many excellent vision transformer models are proposed, and they can mainly be divided into two categories, i.e., pure and hybrid vision transformers. *The former* only contains transformer module without CNN-based layers, and early works (Touvron et al., 2021a; Zhou et al., 2021; Han et al., 2021; Touvron et al., 2021b; Yuan et al., 2021b; Chu et al., 2023; Ali et al., 2021) follow columnar structure of original ViT. Typically, DeiT (Touvron et al., 2021a) propose an efficient training recipe to moderate the dependence on large datasets, DeepViT (Zhou et al., 2021) and CaiT (Touvron et al., 2021b) focus on fast performance saturation when scaling ViT to be deeper, and TNT (Han et al., 2021) divide local patches into smaller patches for fine-grained modeling. Furthermore, researchers (Wang et al., 2021a; Liu et al., 2021; Huang et al., 2021; Vaswani et al., 2021; Chu et al., 2021; Liu et al., 2022b; Hassani et al., 2023; Chen et al., 2021g; Yu et al., 2022; Ren et al., 2022) advance ViT to pyramid structure that is more powerful and suitable for dense prediction. PVT (Wang et al., 2021a) leverages a non-overlapping patch partition to reduce feature size, while Swin (Liu et al., 2021) utilizes a shifted window scheme to alternately model in-window and cross-window connection. *The latter* incorporates the idea of convolution that owns natural inductive bias of locality and translation invariance, and this kind of combination dramatically improves the model effect. Specif-

ically, Srinivas et al. (2021) advance CNN-based models by replacing the convolution of the bottleneck block with the MSA structure. Later researches (Li et al., 2021c; Yuan et al., 2021a; d’Ascoli et al., 2021a; Xu et al., 2021c) introduce convolution designs into columnar visual transformers, while works (Yang et al., 2022; Yuan et al., 2022; Wang et al., 2022c; Lu et al., 2021; Li et al., 2022; Xu et al., 2021b; Wu et al., 2021; Wang et al., 2022b; Zhang et al., 2023b) fuse convolution structures into pyramid structure or use CNN-based backbone on early stages, which has obvious advantages over pure transformer models. Moreover, some researchers design hybrid models from the parallel perspective of convolution and transformer (Xu et al., 2021b; Chen et al., 2022c, 2022a), while Dosovitskiy et al. (2021) introduce a deformable idea (Dai et al., 2017) to MSA module and obtain a boost on Swin (Liu et al., 2021). Recently, MPViT (Lee et al., 2022) explores multi-scale patch embedding and multi-path structure that enable both fine and coarse feature representations simultaneously. Benefiting from advances in basic vision transformer models, many task-specific models are proposed and achieve significant progress in down-stream vision tasks, e.g., object detection (Carion et al., 2020; Zhu et al., 2021; Fang et al., 2021; Chen et al., 2022b), semantic segmentation (Zheng et al., 2021; Chen et al., 2021c; Valanarasu et al., 2021; Xie et al., 2021; Yuan et al., 2021c; Cheng et al., 2021, 2022), generative adversarial network (Jiang et al., 2021a; Wan et al., 2022; Hudson and Zitnick, 2021), low-level vision (Chen et al., 2021b; Liang et al., 2021; Zamir et al., 2022), video understanding (Neimark et al., 2021; Bertasius et al., 2021; Thatipelli et al., 2022; Wang et al., 2022a; Xu et al., 2021a), self-supervised learning (Atito et al., 2021; Chen et al., 2021f; Nakashima et al., 2022; He et al., 2022; Bao et al., 2022; Caron et al., 2021; Dong et al., 2023; Chen et al., 2023; Xie et al., 2022; Wei et al., 2022; Baevski et al., 2022; Gao et al., 2022), neural architecture search (Wang et al., 2020a; Li et al., 2021a; Chen et al., 2021a,d,e), etc. Inspired by practical improvements in EA variants, this work migrates them to Transformer improvement and designs a powerful visual model with higher precision and efficiency than contemporary works. Also, thanks to the elaborate analogical design, the proposed EATFormer in this paper is highly explanatory.

2.3 Explanatory of Transformers

Transformer-based models have achieved remarkable results on CV tasks, leading us to question why Transformer works so well, even better than Convolutional Neural Networks (CNN). Many efforts have done to answer this question. Goyal and Bengio (2022) point out that studying the kind of inductive biases that humans and animals exploit could help inspire AI research and neuroscience theories. Pan et al. (2022) show a strong underlying relation between convolution and self-attention operations. Cordonnier et al. (2020)

prove that a multi-head self-attention layer with a sufficient number of heads is at least as expressive as any convolutional layer, while Raghu et al. (2021) find striking differences between ViT and CNN on image classification. Li et al. (2022) seamlessly integrate the merits of convolution and self-attention in a concise transformer format, while Con-Vit (d'Ascoli et al., 2021a) combines the strengths of both CNN/Transformer architectures to introduce gated positional self-attention. Introducing local CNN into Transformer is followed by many subsequent works (Chu et al., 2023; Li et al., 2022; Guo et al., 2022; Zhang et al., 2023b, Mehta & Rastegari, 2022; Maaz et al., 2023). Furthermore, Min et al. (2022) take a biologically inspired approach and explore modeling peripheral vision by incorporating peripheral position encoding to the multi-head self-attention layers in Transformer. Besides, some works explore the relation between Transformer and other models, e.g., Katharopoulos et al. (2020) reveals their relationship of Transformer to recurrent neural networks, and Kim et al. (2022) prove that Transformer is theoretically at least as expressive as an invariant graph network composed of equivariant linear layers. Moreover, Bhojanapalli et al. (2021) find that ViT models are at least as robust as the ResNet counterparts on a broad range of perturbations when pre-trained with a sufficient amount of data. Hao et al. (2021) propose a self-attention attribution method to interpret the information interactions inside Transformer, while Liu et al. (2022a) propose an actionable diagnostic methodology to measure the consistency between explanation weights and the impact polarity for attention-based models. Dong et al. (2021a) find that MLP stops the output from degeneration, and removing MSA in Transformer would also significantly damage the performance. Recently, Qiang et al. (2022) propose a novel Transformer explanation technique via attentive class activation tokens by leveraging encoded features, gradients, and attention weights to generate a faithful and confident explanation. Xu et al. (2023) propose a new way to visualize the model by firstly computing attention scores based on attribution and then propagating these attention scores through the layers. Works Yu et al. (2022), Zhang et al. (2023a) demonstrate that the general architecture of the Transformers is more essential to performance rather than the specific token mixer module. The above work explores the interpretation of Transformer from a variety of perspectives. At the same time, we will provide another explanation from the perspective of evolutionary algorithms and design a robust model to perform multiple CV tasks.

3 Preliminary Transformer

The vision transformer generally refers to the encoder part of the original transformer structure, which consists of Multi-head Self-Attention layer (MSA), Feed-Forward Network

(FFN), Layer Normalization (LN), and Residual Connection (RC). Given the input feature maps $X_{img} \in \mathbb{R}^{C \times H \times W}$, Img2Seq operation firstly flattens it to a 1D sequence $X_{seq} \in \mathbb{R}^{C \times N}$ that complies with standard NLP format, denoted as: $X_{seq} = \text{Img2Seq}(X_{img})$.

- **MSA** fuses several SA operations to process QKV that jointly attend to information in different representation subspaces. Specifically, LN solved X_{seq} goes through linear layers to obtain projected queries (Q), keys (K) and values (V) presentations, formulated as:

$$\begin{aligned} \text{MSA}(X_{seq}) &= \left(\bigoplus_{h=1}^H X_h \right) W^O, \\ \text{where } X_h &= \text{Attention} \left(X_{seq} W_h^Q, X_{seq} W_h^K, X_{seq} W_h^V \right) \\ &= \text{Softmax} \left(\left[\frac{(X_{seq} W_h^Q)(X_{seq} W_h^K)^T}{\sqrt{d_k}} \right] \right) X_{seq} W_h^V \\ &= \text{Softmax} \left(\left[\frac{Q_h K_h^T}{\sqrt{d_k}} \right] \right) V_h \\ &= A_h V_h, \end{aligned} \quad (1)$$

where d_m is the input dimension, while d_q , d_k , and d_v are hidden dimensions of the corresponding projection subspace, and generally d_q equals d_k ; h is the head number; $W_h^Q \in \mathbb{R}^{d_m \times d_q}$, $W_h^K \in \mathbb{R}^{d_m \times d_k}$, and $W_h^V \in \mathbb{R}^{d_m \times d_v}$ are parameter matrices for QKV , respectively; $W^O \in \mathbb{R}^{hd_v \times d_m}$ maps each head feature X_h to the output; \oplus means concatenation operation; $A_h \in \mathbb{R}^{l \times l}$ is the attention matrix of h -th head.

- **FFN** consists of two cascaded linear transformations with a ReLU activation in between:

$$\text{FFN}(X_{seq}) = \max(0, X_{seq} W_1 + b_1) W_2 + b_2, \quad (2)$$

where W_1 and W_2 are weights of two linear layers, while b_1 and b_2 are corresponding biases.

- **LN** is applied before each layer of MSA and FFN, and the transformed \hat{X}_{seq} is calculated by:

$$\hat{X}_{seq} = X_{seq} + [\text{MSA} \mid \text{FFN}](\text{LN}(X_{seq})). \quad (3)$$

Finally, reversed Seq2Img operation *reshapes* the enhanced \hat{X}_{seq} back to 2D feature maps, denoted as: $\hat{X}_{img} = \text{Seq2Img}(\hat{X}_{seq})$.

4 EA-Inspired Vision Transformer

In this section, we expand the relationship among operators in naive EA and modules in naive Transformer, and con-

sistent mathematical formulations for each conceptual pair can be derived, revealing evolutionary explanations for *the rationality of Vision Transformer structure*. Inspired by the core ideas of some effective EA variants, we deduce them into transformer architecture design and improve a *mighty* pyramid EATFormer over the previous columnar model.

4.1 Evolutionary Explanation of Transformer

As aforementioned in Fig. 1, the Transformer block has conceptually similar sub-modules analogously to evolutionary algorithm. Basically, Transformer inputs a sequence of patch tokens while EA evolves a population that consists of many individuals. Both of which have the consistent vector format and necessary initialization. In order to facilitate the subsequent analogy and formula derivation, we symbolize the patch token (individual) as $\mathbf{x}_i = [x_{i,1}, x_{i,2}, \dots, x_{i,D}]$, where i and D indicate data order and dimension, respectively. Define L as the sequence length, the sequence (population) can be denoted as $\mathbf{X} = [\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_L]^T$. The specific relationship analyses of different components are as follows:

• Crossover Operator vs MSA Module

For the crossover operator of EA, it aims at creating new individuals by combining parts of other individuals. For an individual \mathbf{x}_i specifically, the operator will randomly pick another individual $\mathbf{x}_j = [x_{j,1}, x_{j,2}, \dots, x_{j,D}]$ ($1 \leq j \leq L$) in the global population and randomly replaces features of \mathbf{x}_i with \mathbf{x}_j to form the new individual $\hat{\mathbf{x}}_i$:

$$\hat{\mathbf{x}}_{i,d} = \begin{cases} \mathbf{x}_{j,d}, & \text{if randb}(d) \leq CR \\ \mathbf{x}_{i,d}, & \text{otherwise} \end{cases} \quad (4)$$

s.t. $i \neq j, d \in \{1, 2, \dots, D\}$,

where randb(d) is the d -th evaluation of a uniform random number generator with outcome in $[0, 1]$, and CR is the crossover constant in $[0, 1]$ that is determined by the user. We re-formulate this process as:

$$\begin{aligned} \hat{\mathbf{x}}_i &= \mathbf{x}_{i,1}\mathbf{w}_{i,1}, \dots, \mathbf{x}_{i,D}\mathbf{w}_{i,D} \\ &\quad + [\mathbf{x}_{j,1}\mathbf{w}_{j,1}, \dots, \mathbf{x}_{j,D}\mathbf{w}_{j,D}] \\ &= \mathbf{x}_1 \odot \mathbf{0} + \dots \mathbf{x}_i \odot \mathbf{w}_i + \dots \mathbf{x}_j \odot \mathbf{w}_j + \dots \mathbf{x}_L \odot \mathbf{0} \\ &= \mathbf{x}_1 \mathbf{0} + \dots \mathbf{x}_i \mathbf{W}_i^{cr} + \dots \mathbf{x}_j \mathbf{W}_j^{cr} + \dots \mathbf{x}_L \mathbf{0} \\ &= \sum_{l=1}^L (\mathbf{x}_l \mathbf{W}_l^{cr}), \quad (5) \\ \text{s.t. } &\mathbf{w}_i + \mathbf{w}_j = \mathbf{1}, \\ &\mathbf{w}_{i,d} \in [0, 1], \mathbf{w}_{j,d} \in [0, 1], d \in \{1, 2, \dots, D\}, \end{aligned}$$

where \mathbf{w}_i and \mathbf{w}_j are vectors filled with zeros or ones, indicating the feature selections of \mathbf{x}_i and \mathbf{x}_j , while \mathbf{W}_i^{cr}

and \mathbf{W}_j^{cr} are corresponding diagonal matrix representations. \odot means the point-wise multiplication operation for each position. $\mathbf{0}$ represents that corresponding individual has no contribution, i.e., $\mathbf{W}_l^{cr} (l \neq i, j)$ fulls of zeros. As can be seen above, crossover operator is actually a sparse global feature interaction process.

For the MSA module of Transformer, each patch embedding interacts with all embeddings in dense communications. Without loss of generality, \mathbf{x}_i interacts with the whole population \mathbf{X} as follows:

$$\begin{aligned} \hat{\mathbf{x}}_i &= \bigoplus_{h=1}^H \hat{\mathbf{x}}_{i,h} \\ &= \bigoplus_{h=1}^H \sum_{l=1}^L \mathbf{A}_{l,h} \mathbf{V}_{l,h} \\ &= \bigoplus_{h=1}^H \sum_{l=1}^L \mathbf{A}_{l,h} \mathbf{x}_l \mathbf{W}_h^V \\ &= \bigoplus_{h=1}^H \sum_{l=1}^L (\mathbf{A}_{l,h} \mathbf{x}_l \mathbf{W}_h^V) \\ &= \sum_{l=1}^L \left(\mathbf{x}_l \bigoplus_{h=1}^H (\mathbf{A}_{l,h} \mathbf{W}_h^V) \right) \\ &= \sum_{l=1}^L (\mathbf{x}_l (\mathbf{A}_l \mathbf{W}^V)), \quad (6) \end{aligned}$$

where $\mathbf{A}_{l,h}$ ($l \in \{1, 2, \dots, L\}$) is the attention weight of h -th head from embedding token \mathbf{x}_l to \mathbf{x}_i , which is calculated between the query value of $\mathbf{x}_{i,h}$ and the key value of $\mathbf{x}_{l,h}$ of h -th head followed with a Softmax(\cdot) post-processing; $\mathbf{V}_{l,h}$ ($l \in \{1, 2, \dots, L\}$) is the projected Value feature for \mathbf{x}_l with corresponding weights \mathbf{W}_h^V ; $\hat{\mathbf{x}}_{i,h}$ is the sum of all weighted $\mathbf{V}_{l,h}$ ($l \in \{1, 2, \dots, L\}$) by $\mathbf{A}_{l,h}$ ($l \in \{1, 2, \dots, L\}$), i.e., $\hat{\mathbf{x}}_{i,h} = \sum_{l=1}^L \mathbf{A}_{l,h} \mathbf{V}_{l,h}$, (c.f., Eq. 1 in Sect. 3) for more details. \mathbf{W}^V is the parameter matrix for the value projection and \bigoplus means the concatenation operation. By comparing Eq. (5) with Eq. (6), we find that both above components have the same formula representation, and the crossover operation is a sparse global interaction while densely-modeling MSA has more complex computing and modeling capabilities.

• Mutation Operator vs FFN Module

For the mutation operator in EA, it brings random evolutions into the population by stochastically changing specific features of individuals. Specifically, an individual \mathbf{x}_i in the population goes through Mutation operation

to form the new individual \hat{x}_i , formulated as follows:

$$\hat{x}_{i,d} = \begin{cases} \text{rand}(v_d^L, v_d^H) \cdot x_{i,d}, & \text{if } \text{randb}(d) \leq MU \\ 1 \cdot x_{i,d}, & \text{otherwise} \end{cases} \quad (7)$$

s.t. $d \in \{1, 2, \dots, D\}$,

where $\text{randb}(d)$ is the d -th evaluation of a uniform random number generator with outcome in $[0, 1]$, and MU is the mutation constant in $[0, 1]$ that user determines. v_j^L and v_j^H are lower and upper scale bounds of the j -th feature relative to $x_{i,d}$. Similarly, we re-formulate this process as:

$$\begin{aligned} \hat{x}_i &= x_i \odot w_i \\ &= x_i W_i^{mu}, \end{aligned} \quad (8)$$

where w_i is a randomly generated vector that represents weights of each feature value, while W_i^{mu} is the corresponding diagonal matrix representation; \odot means the point-wise multiplication operation for each position.

For the FFN module in Transformer, each patch embedding carries on directional feature transformation through cascaded linear layers (c.f., Eq. 2). Getting rid of complex nonlinear transformations, we only take one linear layer as an example:

$$\hat{x}_i = x_i W^{FFN}, \quad (9)$$

where W^{FFN} is the weight of the linear layer, and it is applied to each embedding separately and identically. By analyzing the calculation process of Eqs. (refeq:mutation) and (9), Mutation and FFN operations share a unified form of matrix multiplication, so they are supposed to own a consistent function essentially. Besides, at the microcosmic level, the weight of FFN change dynamically during the training process, so the output of the individual differs among different iterations (similar to the random process of mutation). At the macroscopic objective of the algorithm, the mutation in EA is optimized into one potential direction under the constraint of the objective function (statistically speaking, only partial mutation individuals are retained, that is, the mutation also has a determinate meaning in the whole training process). In comparison, the trained FFN can be regarded as a directional mutation under the constraint of loss functions. Finally, note that we are only discussing the comparison with the mutation on one linear layer of FFN, and W^{FFN} is more expressive than diagonal W_i^{mu} in fact because it contains cascaded linear layers and the non-linear ReLU activation is interspersed between adjacent linear layers, as depicted in Eq. (2).

• Population Succession vs RC Operation

In the evolution of the biological population, individuals at the current iteration have a certain probability of inheriting to the next iteration, where a partial population of the current iteration will be combined with the selected individuals. Similarly, the above pattern is expressed by Transformer structure in the form of Residual Connection (RC), i.e., patch embeddings of the previous layer are directly mapped to the next layer. Specifically, partial-selection can be viewed as a dropout technique in Transformer, while population succession can be formulated as a concatenation operation that has a consistent mathematical expression with residual concatenation, whereas addition operation can be regarded as a particular case of the concatenation operation that shares some partial weights.

• Best Individual vs Task-Related Token

Generally speaking, the Transformer-based model chooses an enhanced task-related token (e.g., classification token) that combines information of all patch embeddings as the output feature, while the EA-based method chooses the individual with the best fitness score among the population as the output.

• Necessity of Modules in Transformer

As described in the work (Hassanat et al., 2019), the absence of the crossover operator or mutation operator will significantly damage the model's performance. Similarly, Dong et al. (2021a) explore the effect of MLP in the Transformer and find that MLP stops the output from degeneration, and removing MSA in Transformer would also significantly damage the effectiveness of the model. Thus we can conclude that global information interaction and individual evolution are necessary for Transformer, just like the global crossover and individual mutation in EA.

4.2 Short Description of Previous Columnar EAT

We explore the relationship among operators in naive EA and modules in naive Transformer in the previous NeurIPS'21 conference (Zhang et al., 2021) and analogically improve a columnar EAT based on ViT model. Figure 4a shows the structure of EAT model that is stacked of with N improved Transformer blocks inspired by local population concept in some EA works (Kolen & Pesch, 1994; Land, 1998; Moscato, 1989), where a local path is introduced in parallel with global MSA operation. Also, this work designs a Task-Related Head to deal with various tasks more flexibly, i.e., classification and distillation.

However, the columnar structure is naturally inadequate for downstream dense prediction tasks, and it is inferior in terms of accuracy compared with contemporaneous works (Wang et al., 2021a; Liu et al., 2021), which limits the use-

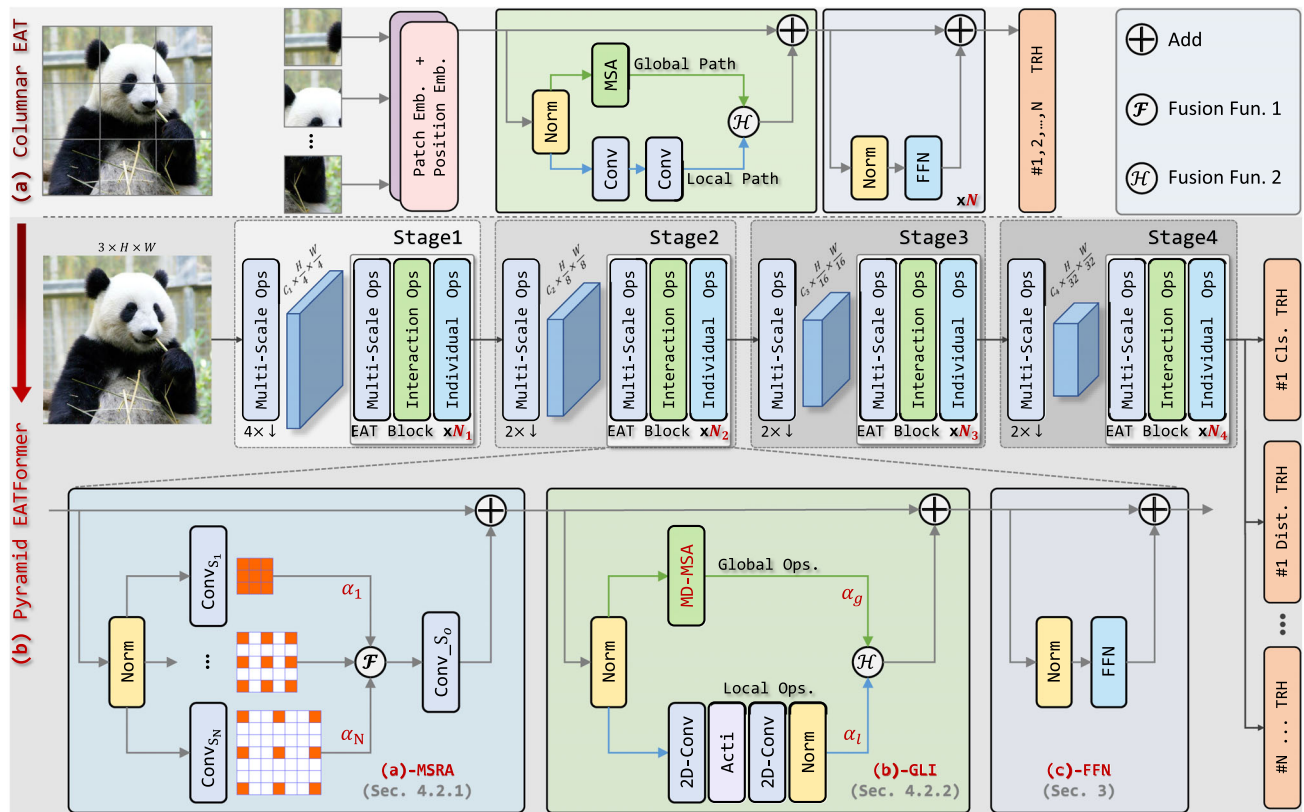


Fig. 4 Structure of EA-inspired columnar EAT model (Zhang et al., 2021) and improved pyramid EATFormer. **a** The *top* part shows the architecture of the previous EAT model, where the basic block consists of parallel global and local paths as well as an FFN module. **b** The *middle* part illustrates overall architecture of EATFormer that contains four stages with i -th stage consisting of N_i basic EAT blocks; The *bot-*

tom part illustrates the structure of serial modules in EAT block, i.e., MSRA (c.f., Sect. 4.3.1), GLI (c.f., Sect. 4.3.2), and FFN (c.f., Sect. 3) from left to right, and a MD-MSA is proposed to effectively improve the model performance; The right part shows the designed Task-Related Head module docked with transformer backbone for specific tasks

fulness of the model in some scenarios. To address the above weaknesses, this paper further explores analogies between EA and Transformer and improves the previous work to a pyramid EATFormer, consisting of the newly designed EAT block inspired by the effective EA variants.

4.3 Methodology of Pyramid EATFormer Architecture

Architecture of the improved EATFormer is illustrated in Fig. 4b, which contains four stages of different resolutions following PVT (Wang et al., 2021a). Specifically, the model is made up of EAT blocks that contains three mixed-paradigm $y = f(x) + x$ residuals: (a) Multi-Scale Region Aggregation (MSRA), (b) Global and Local Interaction (GLI), and (c) Feed-Forward Network (FFN) modules, and the down-sampling procedure between two stages is realized by MSRA with stride greater than 1. Besides, we propose a novel Modulated Deformable MSA (MD-MSA) to advance global modeling and a Task-Related Head (TRH) to complete different tasks more elegantly and flexibly.

4.3.1 Multi-scale Region Aggregation

Inspired by some multi-population-based EA methods (Chen & Kang, 2005; Li et al., 2021b) that would adopt different searching regions for obtaining a better model performance, we analogically extend this concept to multiple sets of spatial positions for the 2D image and design a novel *Multi-Scale Region Aggregation* (MSRA) module for the studied vision transformer. As shown in Fig. 4a, MSRA contains N local convolution operations (i.e., Conv_{S_n} , $1 \leq n \leq N$) with different strides to aggregate information from different receptive fields, which simultaneously *play* the role of providing inductive bias without extra position embedding procedures. Specifically, the n -th dilation operation o_n that transforms input feature map x can be formulated as:

$$o_n(x) = \text{Conv}_{S_n}(\text{Norm}(x)) \quad (10)$$

$$\text{s.t. } n \in \{1, 2, \dots, N\},$$

Weighted Operation Mixing (WOM) mechanism is further proposed to mix all operations by a softmax function over

a set of learnable weights $\alpha_1, \dots, \alpha_N$, and the intermediate representation \mathbf{x}_o is calculated by the mixing function \mathcal{F} as follows:

$$\mathbf{x}_o = \sum_{n=1}^N \frac{\exp(\alpha_n)}{\sum_{n'=1}^N \exp(\alpha_{n'})} o_n(\mathbf{x}), \quad (11)$$

where \mathcal{F} in the above formula is the addition function, and other fusion functions like concatenation are also available for a better effect at the cost of more parameters. The paper chooses the addition function acquiescently. Then, a convolution layer Conv_{S_o} maps \mathbf{x}_o to the same number of channels as the input \mathbf{x} , and the final output of the module is obtained after a residual connection. Also, the MSRA module serves as the *model stem* and *Patch Embedding* that makes the EATFormer more uniform and elegant. Note that this paper does not use any form of position embedding since CNN-based MSRA can provide a natural inductive bias for the next GLI module.

4.3.2 Global and Local Interaction

Motivated by EA variants (Moscato, 1989; Hart et al., 2005; Kumar et al., 2014) that introduce local search procedures besides conventional global search for converging higher-quality solutions faster and effectively (c.f., Fig. 1c for a better intuitive explanation), we improve a MSA-based global module to a novel *Global and Local Interaction* (GLI) module. As shown in Fig. 4b, GLI contains an extra local path in parallel with the global path, where the former aims to mine more discriminative locality-relevant information like the above-mentioned local population idea, while the latter is retained to model global information. Specifically, the input features are divided into global features (marked green) and local features (marked blue) at the channel level with ratio p , which are then fed into global and local paths to conduct feature interactions, respectively. Note that we also apply the proposed *Weighted Operation Mixing* mechanism in 4.3.1 to balance two branches, i.e., global weight α_g and local weight α_l . The outputs of the two paths recover the original data dimension by concatenation operation \mathcal{H} . Thus the improved module is very flexible and can be viewed as a plug-and-play module for the current transformer structure. In detail, the local operation can be traditional convolution layer or other improved modules, e.g., DCN (Dai et al., 2017; Zhu et al., 2019), local MSA, etc, while global operation can be MSA (Vaswani et al., 2017; Dosovitskiy et al., 2021), D-MSA (Chen et al., 2021g), Performer (Choromanski et al., 2021), etc.

In this paper, we choose naive convolution with MSA modules as basic compositions of GLI, and it owns $O(1)$ maximum path length between any two positions for keep-

ing global modeling capability besides enhancing locality, as shown in Table 1. Therefore, the proposed structure maintains the same parallelism and efficiency as the original vision transformer. Also, the selection of feature separation ratio p is crucial to the effect and efficiency of the model because different ratios bring different parameters, FLOPs, and precision of the model. In detail, the local path contains a group of point-wise and $k \times k$ depth-wise convolutions. Assume that the feature map in $\mathbb{R}^{C \times H \times W} = \mathbb{R}^{C \times L}$, and both paths have $C_g = p \times C$ and $C_l = C - C_g$ channels, respectively. Here we present an analysis process about the number of parameters and computation of the improved GLI module as follows:

(1) **Overall Params** equals $4(C_g + 1)C_g + (k^2 + 1)C_l + (C_l + 1)C_l$ according to Table 1, and it is factorized based on $C_l = C - C_g$:

$$\begin{aligned} Params &= 5C_g^2 + (2 - 2C - K^2)C_g \\ &\quad + (k^2 + 2 + C)C. \end{aligned} \quad (12)$$

Applying the minimum value formula of a quadratic function, Eq. (12) obtains the minimum value when $C_g^{min_p} = 0.2C + 0.1(k^2 - 2)$. Given that the channel number are integers and latter term can be ignored, we obtains $C_g^{min_p} = 0.2C$, i.e., p^{min_p} equals 0.2.

(2) **Overall FLOPs** equals $8C_g^2L + 4C_gL^2 + 3L^2 + (2k^2)LC_l + 2C_lLC_l$ according to Table 1, and it is factorized based on $C_l = C - C_g$:

$$\begin{aligned} FLOPs &= 10LC_g^2 + (4L^2 - 2k^2L - 4LC)C_g \\ &\quad + (3L + 2k^2C + 2C^2)L. \end{aligned} \quad (13)$$

Applying the minimum value formula of a quadratic function, Eq. (13) obtains the minimum value when $C_g^{min_f} = 0.2C + 0.1(k^2 - 2L)$. Also, ignoring the latter term, we obtains $C_g^{min_f} = 0.2C$ that follows the same trend with $C_g^{min_p}$. Therefore, we can draw two conclusions: ① The parameters and calculations of GLI are much lower than single-path MSA ($p < 1$), and the minimum value can be obtained when using both paths ($p > 0$); ② According to Eqs. (12) and (13), there is not much difference about the total parameters and calculations when p lies in the range $[0, 0.5]$, so p is set 0.5 for all layers in this paper for simplicity and efficiency. Also, experiments in Sect. 5.5.1 demonstrate that $p = 0.5$ is the most economical and efficient option. Note that the number of convolution parameters and computation of the current local path are smaller compared with the global path, while the stronger local structure will make ratio

Table 1 Properties of convolution and MSA layers with Parameters (Params), floating point operations (FLOPs), and Maximum Path Length (MPL). Assume that the input and output feature maps in $\mathbb{R}^{C \times H \times W}$, $L = H \times W$, $H = W$, k and G are kernel size and group number for convolution layers

Type	Params	FLOPs	MPL
MSA	$4(C+1)C$	$8C^2L + 4CL^2 + 3L^2$	$O(1)$
Conv	$(Ck^2/G + 1)C$	$(2Ck^2/G)LC$	$O(H/k)$

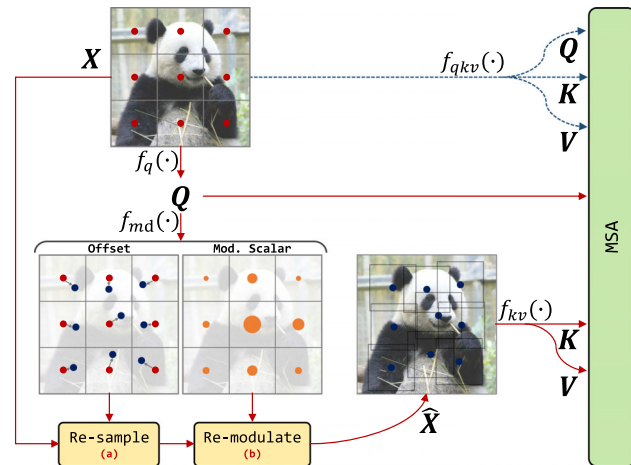


Fig. 5 Structure of the proposed MD-MSA

p change larger and this paper will not elaborate on the details.

Furthermore, we advance the global path by designing a Modulated Deformable MSA (MD-MSA in Sect. 4.3.3) module, which improves the model performance with negligible parameters and GFLOPs increasing, and a comparison study to explore combinations of different operations is further conducted in the experimental section.

4.3.3 Modulated Deformable MSA

Inspired by the irregular spatial distribution among real individuals that are not as horizontal and vertical as the image, we improve a novel Modulated Deformable MSA (MD-MSA) module that considers position fine-tuning and re-weighting of each spatial patch. As shown in Fig. 5, the blue dotted line represents naive MSA procedure that QKV features are obtained by the input feature map X from function $f_{qkv}(\cdot)$, i.e., $QKV = f_{qkv}(X)$ and $f_{qkv} = f_q \oplus f_k \oplus f_v$ (\oplus denotes concatenation operation), while the red solid line shows the procedure of MD-MSA. And the main difference between the proposed MD-MSA and original MSA lies in the query-aware access of fine-tuned feature map \hat{X} to extract KV features further. Specifically, given the input feature map X with L positions, Q is obtained by function f_q , i.e.,

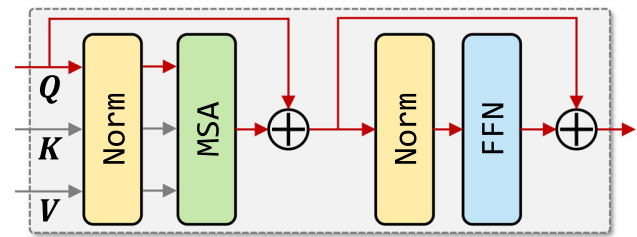


Fig. 6 Structure of the proposed Task-Related Head

$Q = f_q(X)$, which is then used to predict deformable offset Δl and modulation scalar Δm for all positions:

$$\Delta l, \Delta m = f_{md}(Q). \quad (14)$$

For the l -th position, the re-sampled and re-weighted feature \hat{X}_l is calculated by:

$$\hat{X}_l = S(X_l, \Delta l) \cdot \Delta m, \quad (15)$$

where Δl is the relative coordinate with an unconstrained range for the l -th position, while Δm lies in the range $[0, 1]$, and S represents the bilinear interpolation function. Then KV is obtained with the new feature map \hat{X} , i.e., $KV = f_{kv}(\hat{X})$. It is worth mentioning that the main difference between MD-MSA and recent similar work (Xia et al., 2022) lies in the modulation operation, where MD-MSA could apply appropriate attention to different position features to obtain better results. Also, any form of position embedding is *not used* since it makes no contribution to results, and detailed comparative experiments can be viewed in Sect. 5.4.3.

4.3.4 Task-Related Head

Current transformer-based vision models would initialize different tokens for different tasks Touvron et al. (2021a) or use the pooling operation to obtain global representation Liu et al. (2021). However, both manners are potentially incompatible: the former treats the task token and image patches coequally as unreasonable and clumsy, because the task token and image tokens have different feature distributions while additional computation is required from $O(n^2)$ to $O((n+1)^2)$; the latter uses only one pooling result for multiple tasks that is also inappropriate and harmful for losing wealthy information. Inspired by multi-objective EAs (Coello & Lamont, 2004; Khare et al., 2003; Toffolo et al., 2003) that find a set of solutions for different targets, we design a *Task-Related Head* (TRH) docked with transformer backbone to obtain the corresponding task output through the final features. As shown in Fig. 6, we employ a cross-attention paradigm to implement this module: K and V (gray lines) are output features extracted by the transformer backbone, while

Table 2 Overall analogical correlations between EA and EATFormer

	EA	EATFormer	
Basics	Population size	Patch number	(Sect. 1)
	(Discrete) Individual	(Continuous) Patch token	
	(Sparse) Crossover operation	(Dense) Global MSA module	
	(Sparse) Mutation operation	(Dense) Individual FFN module	
	(Partial) Population succession	(Integral) Residual connection	
	(Global) Best individual	(Aggregated) Task-related token	
Improvements	Multi-scale population	Multi-scale region aggregation module	(Sect. 4.3.1)
	Global and local population	Global and Local interaction module	(Sect. 4.3.2)
	Self-adapting parameters	Weighted operation mixing	(Sect. 4.3.2)
	Irregular population distribution	Modulated deformable MSA	(Sect. 4.3.3)
	Multi-objective EA	Task-related head	(Sect. 4.3.4)
	Dynamic population	Pyramid architecture	(Fig. 4)

Q (red line) is the task-related token to integrate global information. Note that this design is more effective and flexible for different tasks learning simultaneously while consuming a negligible computation amount compared to the backbone, and more analytical experiments can be viewed in the following Sect. 5.4.8. For a more fair comparison, TRH presented in the former conference version (Zhang et al., 2021) is not used by default because this plug-and-play module can easily be added to other methods, and we will conduct an ablation experiment in Sect. 5.4.8 to verify the validity of TRH.

4.3.5 Overall Congruent Relationships

To more clearly show the design inspirations of different modules, we summarize the analogies between the improved EATFormer research and homologous concepts (ideas) from EA variants in Table 2.

4.4 EATFormer Variants

In the former conference version (Zhang et al., 2021), we improve the columnar ViT by introducing a local path in parallel with global MSA operation, denoted as EAT-Ti, EAT-S, and EAT-B in the top part of Table 3. In this paper, we extend the columnar structure to a pyramid architecture and carefully re-design a novel EATFormer model, which has a series of scales for different practical applications, and these variants can be viewed in the bottom part of Table 3. Except for the depth and dimension of the model, other parameters remain consistent for all models: the head dimension of MSA is 32; window size is set to 7; kernel size of all convolution is 3×3 ; dilations of the MSRA module for four stages are [1], [1], [1,2,3], and [1,2], respectively; low-level stage1–2 only use local path while high-level stage3–4 employ hybrid GLI

module for efficiency. More detailed structures and implementations can be viewed in the attached source code.

4.5 Further Discussion

Compared with EAT in the former conference version, the improved EATFormer has better inspirations, finer analogical designs, and more sufficient experiments. And we prove the effectiveness and integrity of the proposed method through a series of following experiments, such as comparison with SOTA methods, downstream task transferring, ablation studies, and explanatory experiments. It is worth noting that the backbone of EATFormer in this paper only contains one unified EAT block, which fully considers three aspects of modeling: (1) multi-scale information aggregation, (2) feature interactions among tokens, and (3) individual enhancement. Also, the architecture recipes of EATFormer variants in this paper are mainly given by our intuition and proved by experiments, but the alterable configure parameters can be used as the search space for NAS that is worth further exploration in our future works, e.g., embedding dimension, dilations of MSRA, kernel size of MSRA, fusion function of MSRA, down-sampling mode of MSRA, separation ratio of GLI, normalization types, window size, operation combinations of GLI, etc.

5 Experiments

In this section, to evaluate the effectiveness and superiority of our improved EATFormer architecture, we experiment for mainstream vision tasks with models of different volumes as the backbone and orderly conduct down-stream tasks, i.e., *image-level classification* (ImageNet-1K, Deng et al., 2009), *object-level detection and instance segmenta-*

Table 3 Detailed settings of different EATFormer variants. Top part shows previous columnar EAT models (Zhang et al., 2021)

	Network	Depth	Dimension	Params. (M)	FLOPs (G)	Inf. Mem. (G)	Top-1
Col	EAT-Ti	12	192	5.7	1.01	2.2	72.7
	EAT-S	12	384	22.1	3.83	2.9	80.4
	EAT-B	12	768	86.6	14.83	4.5	82.0
Pyramid	EATFormer-Mobile	[1, 1, 4, 1]	[48, 64, 160, 256]	1.8	0.36	2.2	69.4
	EATFormer-Lite	[1, 2, 6, 1]	[64, 128, 192, 256]	3.5	0.91	2.7	75.4
	EATFormer-Tiny	[2, 2, 6, 2]	[64, 128, 192, 256]	6.1	1.41	3.1	78.4
	EATFormer-Mini	[2, 3, 8, 2]	[64, 128, 256, 320]	11.1	2.29	3.6	80.9
	EATFormer-Small	[3, 4, 12, 3]	[64, 128, 320, 448]	24.3	4.32	4.9	83.1
	EATFormer-Medium	[4, 5, 14, 4]	[64, 160, 384, 512]	39.9	7.07	6.2	83.6
	EATFormer-Base	[5, 6, 20, 7]	[96, 160, 384, 576]	63.5	10.89	8.7	83.9

tion (COCO 2017, Lin et al., 2014), and *pixel-level semantic segmentation* (ADE20K, Zhou et al., 2019). Massive ablation and explanatory experiments are further conducted to prove the effectiveness of EATFormer and its components.

5.1 Image Classification

5.1.1 Experimental Setting

All of our EATFormer variants are trained for 300 epochs from scratch *without* pre-training, extra datasets, pre-trained models, token labeling (Jiang et al., 2021b) alike strategy, and exponential moving average. We employ the same training recipe as Deit (Touvron et al., 2021a) to all EATFormer variants for fair comparisons with different SOTA methods: AdamW (Loshchilov and Hutter, 2019) optimizer is used for training with betas and weight decay equaling (0.9, 0.999) and $5e^{-2}$, respectively; Batch size is set to 2048, while learning rate is $5e^{-4}$ by default with a linear increasing compared with batch size divided by 512; Standard cosine learning rate scheduler, data augmentation strategies, warm-up, and stochastic depth are used during the training phase (Touvron et al., 2021a). EATFormer is built on PyTorch (Paszke et al., 2019) and relies on the TIMM interface (Wightman, 2019).

5.1.2 Experimental Results

In this work, we design EATFormer variations at different scales to meet different application requirements, and comparison results with SOTA methods are shown in Table 4. To fully evaluate the effects of different methods, we choose the number of parameters (Params.), FLOPs, Top-1 accuracy on ImageNet-1K, as well as throughput of GPU (with basic batch size equaling 128 by a single V100 SXM2 32GB, and the batch size will be reduced to the maximum that memory requires for large models) and CPU (with batch size equaling 128 by Xeon 8255C CPU @ 2.50GHz) as evalu-

ation indexes. Our smallest EATFormer-Mobile obtains 69.4 that is much higher than MobileNetV3-Small $0.75\times$ counterpart, i.e., 65.4, while the largest EATFormer-Base obtains a very competitive result with only 49.0M parameters, and it further achieves 84.9 at 384×384 resolution. Comparatively, although our approach obtains a slight improvement over recent SOTA MPViT-T/-XS/-S by $+0.2\%/+0.0\%/+0.1\%$, EATFormer features significantly fewer FLOPs by $-0.21\text{G}/-0.68\text{G}/-0.48\text{G}$, faster GPU speed by $+2.1\times/+1.7\times/+1.5\times$, and CPU speed by $+1.33\times/+1.27\times/+1.07\times$. At the highest 50M-level model, our EATFormer-B still achieves a throughput of 329 that is $1.8\times$ \uparrow faster than MPViT-B, and this efficiency increase is also considerable. Meaning that EATFormer is more user-friendly than MPViT on general-purpose GPU and CPU devices, and our EATFormer can better trade-off parameters, computation, and precision. At the same time, our tiny, small, and base models improve by $+5.7\uparrow$, $+2.7\uparrow$, and $+1.9\uparrow$ compared with the previous conference version. Interestingly, we find that the Top-1 accuracy of different methods with 50–80M parameters would be approximately saturated to 84.0 without external data, token labeling, larger resolution, etc., so it is worth future exploration to alleviate this problem.

5.2 Object Detection and Instance Segmentation

5.2.1 Experimental Setting

To further evaluate the effectiveness and superiority of our method, ImageNet-1K (Deng et al., 2009) pre-trained EATFormer is benchmarked as the feature extractor for downstream object detection and instance segmentation tasks on COCO2017 dataset (Lin et al., 2014), and its window size increases from 7 to 12 without global attention and other changes. For fair comparisons, we employ MMDetection library (Chen et al., 2019) for experiments and follow the same training recipe as Swin-Transformer (Liu et al., 2021):

Table 4 Comparison with SOTA methods on ImageNet-1K. Reported results are from corresponding papers. Bold face shows CNN-based and columnar methods, while the proposed EATFormer variants are italic

Network	Params. ↓ (M)	FLOPs ↓ (G)	Images/s ↑		Resolution	Top-1	Pub.
			GPU	CPU			
MNetV3-Small 0.75x (Howard et al., 2019)	2.0	0.05	9872	589.1	224²	65.4	ICCV'19
<i>EATFormer-Mobile</i>	<i>1.8</i>	<i>0.36</i>	<i>3926</i>	<i>456.3</i>	<i>224²</i>	<i>69.4</i>	–
MobileNetV3 0.75x (Howard et al., 2019)	4.0	0.16	5585	315.4	224²	73.3	ICCV'19
PVTv2-B0 (Wang et al., 2021a)	3.6	0.57	1711	104.2	224 ²	70.5	ICCV'21
XCiT-N12 (Ali et al., 2021)	3.1	0.56	2736	290.9	224 ²	69.9	NeurIPS'21
VAN-Tiny (Guo et al., 2023)	4.1	0.87	1706	107.9	224 ²	75.4	arXiv'22
<i>EATFormer-Lite</i>	<i>3.5</i>	<i>0.91</i>	<i>2168</i>	<i>246.3</i>	<i>224²</i>	<i>75.4</i>	–
DeiT-Ti (Touvron et al., 2021a)	5.7	1.25	2342	417.7	224²	72.2	ICML'21
EAT-Ti (Zhang et al., 2021)	5.8	1.01	2356	436.8	224²	72.7	NeurIPS'21
EfficientNet-B0 (Tan and Le, 2019)	5.3	0.39	2835	225.1	224²	77.1	ICML'19
Coat-Lite Tiny (Xu et al., 2021b)	5.7	1.59	1055	143.5	224 ²	77.5	ICCV'21
ViTAE-6M (Xu et al., 2021c)	6.6	2.16	921	152.6	224 ²	77.9	NeurIPS'21
XCiT-T12 (Ali et al., 2021)	6.7	1.25	1750	259.5	224 ²	77.1	NeurIPS'21
MPViT-T (Lee et al., 2022)	5.8	1.65	755	125.9	224 ²	78.2	CVPR'22
<i>EATFormer-Tiny</i>	<i>6.1</i>	<i>1.41</i>	<i>1549</i>	<i>167.5</i>	<i>224²</i>	<i>78.4</i>	–
<i>EATFormer-Tiny-384</i>	<i>6.1</i>	<i>4.23</i>	<i>536</i>	<i>56.9</i>	<i>384²</i>	<i>80.1</i>	–
EfficientNet-B2 (Tan and Le, 2019)	9.1	0.88	1440	143.4	256²	80.1	ICML'19
PVTv2-B1 (Wang et al., 2021a)	14.0	2.12	1006	79.2	224 ²	78.7	ICCV'21
ViTAE-13M (Xu et al., 2021c)	10.8	3.05	698	114.3	224 ²	81.0	NeurIPS'21
XCiT-T24 (Ali et al., 2021)	12.1	2.35	933	146.6	224 ²	79.4	NeurIPS'21

Table 4 continued

Network	Params. ↓ (M)	FLOPs ↓ (G)	Images/s ↑		Resolution	Top-1	Pub.
			GPU	CPU			
Coat-Lite Mini (Xu et al., 2021b)	11.0	1.99	968	164.8	224 ²	79.1	ICCV'21
PoolFormer-S12 (Yu et al., 2022)	11.9	1.82	1858	218.7	224 ²	77.2	CVPR'22
MPViT-XS (Lee et al., 2022)	10.5	2.97	612	95.8	224 ²	80.9	CVPR'22
VAN-Small (Guo et al., 2023)	13.8	2.50	992	95.2	224 ²	81.1	arXiv'22
EATFormer-Mini	<i>11.1</i>	<i>2.29</i>	<i>1055</i>	<i>122.1</i>	<i>224²</i>	<i>80.9</i>	–
DeiT-S (Touvron et al., 2021a)	22.0	4.60	937	163.5	224 ²	79.8	ICML'21
EAT-S (Zhang et al., 2021)	22.1	3.83	964	175.6	224 ²	80.4	NeurIPS'21
ResNet-50 (He et al., 2016; Wightman et al., 2021)	25.5	4.11	1192	123.8	224 ²	80.4	CVPR'16
EfficientNet-B4 (Tan and Le, 2019)	19.3	3.13	495	38.2	320 ²	82.9	ICML'19
Coat-Lite Small (Xu et al., 2021b)	19.8	3.96	542	93.8	224 ²	81.9	ICCV'21
PVTv2-B2 (Wang et al., 2021a)	25.3	4.04	585	36.5	224 ²	82.0	ICCV'21
Swin-T (Liu et al., 2021)	28.2	4.50	664	88.0	224 ²	81.3	ICCV'21
XCiT-S12 (Ali et al., 2021)	26.2	18.92	187	30.1	224 ²	82.0	NeurIPS'21
ViTAE-S (Xu et al., 2021c)	24.0	6.20	399	69.9	224 ²	82.0	NeurIPS'21
UniFormer-S (Li et al., 2022)	21.5	3.64	844	121.1	224 ²	82.9	ICLR'22
CrossFormer-T (Wang et al., 2022c)	27.7	2.86	948	185.4	224 ²	81.5	ICLR'22
DAT-T (Xia et al., 2022)	28.3	4.58	581	75.6	224 ²	82.0	CVPR'22
PoolFormer-S36 (Yu et al., 2022)	30.8	5.00	656	76.4	224 ²	81.4	CVPR'22
MPViT-S (Lee et al., 2022)	22.8	4.80	417	68.7	224 ²	83.0	CVPR'22

Table 4 continued

Network	Params. ↓ (M)	FLOPs ↓ (G)	Images/s ↑		Resolution	Top-1	Pub.
			<i>GPU</i>	<i>CPU</i>			
Shunted-S (Ren et al., 2022)	22.4	5.01	461	48.6	224 ²	83.7	CVPR'22
PoolFormer-S24 (Yu et al., 2022)	21.3	3.41	971	111.9	224 ²	80.3	CVPR'22
CSWin-T (Dong et al., 2022)	22.3	4.34	592	66.6	224 ²	82.7	CVPR'22
iFormer-S (Si et al., 2022)	19.9	4.85	528	58.8	224 ²	83.4	NeurIPS'22
MaxViT-T (Tu et al., 2022)	25.1	4.56	367	36.6	224 ²	83.6	ECCV'22
VAN-Base (Guo et al., 2023)	26.5	5.00	542	51.9	224 ²	82.8	arXiv'22
ViTAEv2-S (Zhang et al., 2023b)	19.3	5.78	464	72.8	224 ²	82.6	IJCV'23
NAT-Tiny (Hassani et al., 2023)	27.9	4.11	401	< 1	224 ²	83.2	arXiv'22
EATFormer-Small	24.3	4.32	615	73.3	224 ²	83.1	–
EATFormer-Small-384	24.3	12.92	198	24.1	384 ²	84.3	–
ResNet-101 (He et al., 2016; Wightman et al., 2021)	44.5	7.83	675	81.5	224²	81.5	CVPR'16
EfficientNet-B5 (Tan and Le, 2019)	30.3	10.46	163	18.6	456²	83.6	ICML'19
PVTv2-B3 (Wang et al., 2021a)	45.2	6.92	392	45.7	224²	83.2	ICCV'21
Coat-Lite Medium (Xu et al., 2021b)	44.5	9.80	275	42.2	224 ²	83.6	ICCV'21
XCiT-S24 (Ali et al., 2021)	47.6	36.04	99	16.0	224 ²	82.6	NeurIPS'21
CSWin-S (Dong et al., 2022)	34.6	6.83	360	51.2	224 ²	83.6	CVPR'22
EATFormer-Medium	39.9	7.05	425	53.4	224 ²	83.6	–
ViT-B/16 (Dosovitskiy et al., 2021)	86.5	17.58	293	53.0	384²	77.9	ICLR'21
DeiT-B (Touvron et al., 2021a)	86.5	17.58	293	53.7	224²	81.8	ICML'21
EAT-B (Zhang et al., 2021)	86.6	14.83	331	71.5	224²	82.0	NeurIPS'21

Table 4 continued

Network	Params. ↓ (M)	FLOPs ↓ (G)	Images/s ↑		Resolution	Top-1	Pub.
			GPU	CPU			
ResNet-152 (He et al., 2016; Wightman et al., 2021)	60.1	11.55	470	57.5	224²	82.0	CVPR'16
EfficientNet-B7 (Tan and Le, 2019)	66.3	38.32	47	5.6	600²	84.3	ICML'19
PVTv2-B5 (Wang et al., 2021a)	81.9	11.76	256	33.9	224²	83.8	ICCV'21
Swin-B (Liu et al., 2021)	87.7	15.46	258	38.6	224 ²	83.5	ICCV'21
Twins-SVT-L (Chu et al., 2021)	99.2	15.14	271	43.1	224 ²	83.2	NeurIPS'21
UniFormer-B (Li et al., 2022)	49.7	8.27	378	58.7	224 ²	83.9	ICLR'22
DAT-B (Xia et al., 2022)	87.8	15.78	217	30.8	224 ²	84.0	CVPR'22
Shunted-B (Ren et al., 2022)	39.6	8.18	290	33.7	224 ²	84.0	CVPR'22
PoolFormer-M48 (Yu et al., 2022)	73.4	11.59	301	37.5	224 ²	82.5	CVPR'22
MPViT-B (Lee et al., 2022)	74.8	16.44	181	29.1	224 ²	84.0	CVPR'22
CSWin-B (Dong et al., 2022)	77.4	15.00	204	32.5	224 ²	84.2	CVPR'22
iFormer-B (Si et al., 2022)	47.9	9.38	262	38.9	224 ²	84.6	NeurIPS'22
MaxViT-S (Tu et al., 2022)	55.8	9.41	231	32.1	224 ²	84.4	ECCV'22
NAT-Small (Hassani et al., 2023)	50.7	7.50	260	< 1	224 ²	83.7	arXiv'22
ViTAEv2-48M (Zhang et al., 2023b)	48.6	13.38	251	38.6	224 ²	83.8	IJCV'23
EATFormer-Base	49.0	8.94	329	43.7	224²	83.9	–
EATFormer-Base-384	49.0	26.11	112	14.2	384²	84.9	–

Table 5 Object detection and instance segmentation with Mask R-CNN on COCO (Lin et al., 2014) dataset for $1\times$ and $3\times$ schedules. All backbones are pre-trained on ImageNet-1K (Deng et al., 2009)

Backbone	Mask R-CNN $1\times$						Mask R-CNN $3\times$						Params. (M)↓	FLOPs (G)↓	Pub.
	A	P^b	A_{50}^b	A_{75}^b	A^m	A_{50}^m	A	P^b	A_{50}^b	A_{75}^b	A^m	A_{50}^m			
PVT-Tiny (Wang et al., 2021a)	36.7	59.2	39.3	35.1	56.7	37.3	39.8	62.2	43.0	37.4	59.3	39.9	33	–	ICCV'21
PVT-v2-B0 (Wang et al., 2022b)	38.2	60.5	40.7	36.2	57.8	38.6	–	–	–	–	–	–	23	195	CVM'22
XCiT-T12 (Ali et al., 2021)	–	–	–	–	–	–	44.5	66.4	48.8	40.4	63.5	43.3	26	266	NeurIPS'21
PFormer-S12 (Yu et al., 2022)	37.3	59.0	40.1	34.6	55.8	36.9	–	–	–	–	–	–	31	–	CVPR'22
MPViT-T (Lee et al., 2022)	42.2	64.2	45.8	39.0	61.4	41.8	44.8	66.9	49.2	41.0	64.2	44.1	28	216	CVPR'22
EATFormer-Tiny	42.3	64.7	46.2	39.0	61.5	42.0	45.4	67.5	49.5	41.4	64.8	44.6	25	198	–
ResNet-50 (He et al., 2016)	38.0	58.6	41.4	34.4	55.1	36.7	41.0	61.7	44.9	37.1	58.4	40.1	44	260	CVPR'16
Swin-T (Liu et al., 2021)	43.7	66.6	47.7	39.8	63.3	42.7	46.0	68.1	50.3	41.6	65.1	44.9	48	267	ICCV'21
Twins-S (Chu et al., 2021)	43.4	66.0	47.3	40.3	63.2	43.4	46.8	69.2	51.2	42.6	66.3	45.8	44	228	NeurIPS'21
PFormer-S24 (Yu et al., 2022)	40.1	62.2	43.4	37.0	59.1	39.6	–	–	–	–	–	–	41	–	CVPR'22
DAT-T (Xia et al., 2022)	44.4	67.6	48.5	40.4	64.2	43.1	47.1	69.2	51.6	42.4	66.1	45.5	48	272	CVPR'22
MPViT-S (Lee et al., 2022)	46.4	68.6	51.2	42.4	65.6	45.7	48.4	70.5	52.6	43.9	67.6	47.5	43	268	CVPR'22
EATFormer-Small	46.1	68.4	50.4	41.9	65.3	44.8	47.4	69.3	51.9	42.9	66.4	46.3	44	258	–
ResNet-101 (He et al., 2016)	40.4	61.1	44.2	36.4	57.7	38.8	42.8	63.2	47.1	38.5	60.1	41.3	63	336	CVPR'16
Swin-S (Liu et al., 2021)	45.7	67.9	50.4	41.1	64.9	44.2	48.5	70.2	53.5	43.3	67.3	46.6	69	359	ICCV'21
Twins-B (Chu et al., 2021)	45.2	67.6	49.3	41.5	64.5	44.8	48.0	69.5	52.7	43.0	66.8	46.6	76	340	NeurIPS'21
PFormer-S36 (Yu et al., 2022)	41.0	63.1	44.8	37.7	60.1	40.0	–	–	–	–	–	–	51	–	CVPR'22
DAT-S (Xia et al., 2022)	47.1	69.9	51.5	42.5	66.7	45.4	49.0	70.9	53.8	44.0	68.0	47.5	69	378	CVPR'22
MPViT-B (Lee et al., 2022)	48.2	70.0	52.9	43.5	67.1	46.8	49.5	70.9	54.0	44.5	68.3	48.3	95	503	CVPR'22
EATFormer-Base	47.2	69.4	52.1	42.8	66.4	46.5	49.0	70.3	53.6	44.2	67.7	47.6	68	349	–

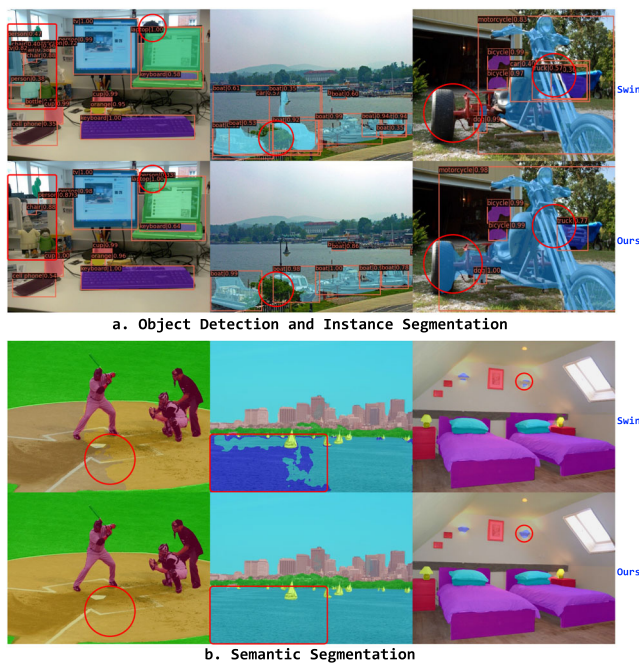


Fig. 7 Intuitive visualizations of two downstream tasks compared with Swin Transformer. Distinct differences are highlighted with red circles and rectangles

$1\times$ schedule for 12 epochs and $3\times$ schedule with a multi-scale training strategy for 36 epochs. AdamW (Loshchilov & Hutter, 2019) optimizer is used for training with learning rate and weight decay equaling $1e^{-4}$ and $5e^{-2}$, respectively.

5.2.2 Experimental Results

Comparison results of box mAP (AP^b) and mask mAP (AP^m) are reported in Table 5, and our improved EATFormer obtains competitive results over recent approaches. Specifically, our tiny model obtains $+5.6\uparrow/+5.6\uparrow$ AP^b and $+3.9\uparrow/+4.0\uparrow$ AP^m improvements over PVT-Tiny Wang et al. (2021a) on both $1\times$ and $3\times$ schedules, while achieves higher results over MPViT (Lee et al., 2022) with less parameters and FLOPs, i.e., $+0.6\uparrow$ and $+0.4\uparrow$ on $3\times$ schedule. For larger EATFormer-small and EATFormer-base models, we consistently get better results than recent counterparts, which surpass Swin-T by $+2.4\uparrow/+2.1\uparrow$ and Swin-S by $+1.5\uparrow/+1.7\uparrow$ with $1\times$ schedule, while by $+1.4\uparrow/+1.3\uparrow$ and by $+0.5\uparrow/+0.9\uparrow$ with $3\times$ schedule. Also, we obtain slightly higher results than DAT (Xia et al., 2022) with computation amount going down by 29 G↓. Although EATFormer is slightly lower than SOTA MPViT-S/B in the downstream task metrics, our method has obvious advantages in the number of parameters and computation, e.g., -10 G↓ FLOPs decreasing than MPViT-S for Mask R-CNN, while -154 G↓ (-30.6%) FLOPs and -27 M↓ (-28.4%) parameters decreasing than MPViT-B, effectively balancing the trade-off between effectiveness and perfor-

mance. For MPViT-T, our EATFormer-Tiny has obvious metrics, parameter numbers, and computation advantages. Qualitative visualizations on validation dataset compared with Swin-S (Liu et al., 2021) are shown in the top part of Fig. 7. Results indicate that our EATFormer can obtain more accurate detection accuracy, fewer false positives, and finer segmentation results than Swin Transformer.

5.3 Semantic Segmentation

5.3.1 Experimental Setting

We further conduct semantic segmentation experiments on the ADE20K (Zhou et al., 2019) dataset, and pre-trained EATFormer with window size equaling 12 is integrated into UperNet (Xiao et al., 2018) architecture to obtain pixel-level predictions. In detail, we follow the same setting of Swin-Transformer (Liu et al., 2021) to train the model for 160k iterations. AdamW (Loshchilov & Hutter, 2019) optimizer is also used with learning rate and weight decay equaling $1e^{-4}$ and $5e^{-2}$, respectively.

5.3.2 Experimental Results

Segmentation results compared with contemporary SOTA works under three main model scales are reported in Table 6. Our EATFormer-Tiny obtains a significantly $+3.4\uparrow$ improvement than recent VAN-Tiny (Guo et al., 2023), while EATFormer-Small achieves a higher mIoU with fewer FLOPs over SOTA methods. For larger EATFormer-Base, it consistently obtains competitive results, i.e., $+1.7\uparrow$ and $+1.0\uparrow$ than Swin-S (Liu et al., 2021) and DAT-S (Xia et al., 2022), respectively. Compared with SOTA MPViT, we obtain a better trade-off among parameters, computation, and precision. E.g., our EATFormer-Base has 26M fewer parameters and 156G fewer FLOPs compared to MPViT-B. Our approach generally has excellent overall precision and computation performance than counterpart. Also, intuitive visualizations of the validation dataset compared with Swin-S (Liu et al., 2021) are shown in the bottom part of Fig. 7. Qualitative results consistently demonstrate the robustness and effectiveness of the proposed approach, where our EATFormer has more accurate segmentation results.

5.4 Ablation Study

To fully evaluate the effectiveness of each designed module, we conduct a series of ablation studies in the following sections. By default, EATFormer-Tiny is used for all experiments, and we follow the same training recipe as mentioned in Sect. 5.1.1.

Table 6 Semantic segmentation results compared with SOTAs on ADE20K (Zhou et al., 2019) by Upernet (Xiao et al., 2018)

Backbone	Params.	GFLOPs	mIoU	Pub
XCiT-T12 (Ali et al., 2021)	34	–	43.5	NeurIPS'21
VAN-Tiny (Guo et al., 2023)	32	858	41.1	CVPR'22
<i>EATFormer-Tiny</i>	34	870	44.5	–
Swin-T (Liu et al., 2021)	60	945	44.5	ICCV'21
XCiT-S12 (Ali et al., 2021)	52	–	46.6	NeurIPS'21
DAT-T (Xia et al., 2022)	60	957	45.5	CVPR'22
ViTAEv2-S (Zhang et al., 2023b)	49	–	45.0	IJCV'23
MPViT-S (Lee et al., 2022)	52	943	48.3	CVPR'22
UniFormer-S (Li et al., 2023a)	52	955	47.0	arXiv'22
<i>EATFormer-Small</i>	53	934	47.3	–
Swin-S (Liu et al., 2021)	81	1038	47.6	ICCV'21
XCiT-M24 (Ali et al., 2021)	109	–	48.4	NeurIPS'21
DAT-S (Xia et al., 2022)	81	1079	48.3	CVPR'22
MPViT-B (Lee et al., 2022)	105	1186	50.3	CVPR'22
UniFormer-B (Li et al., 2023a)	80	1106	49.5	arXiv'22
<i>EATFormer-Base</i>	79	1030	49.3	–

Table 7 Ablation study for different component combinations in EAT block

MSRA	GLI	FFN	Params	FLOPs	Top-1
✓	×	×	2.4	0.45	62.9
×	✓	×	2.6	0.51	64.4
×	×	✓	5.2	1.17	71.4
✓	✓	×	2.9	0.60	67.7
✓	×	✓	5.5	1.26	76.0
×	✓	✓	5.8	1.32	77.4
✓	✓	✓	6.1	1.41	78.4

5.4.1 Component of EAT Block

As afore-mentioned in Sect. 4.3, our proposed EAT block contains: (1) MSRA, (2) GLI, and (3) FFN modules that are responsible for aggregating multi-scale information, interacting global and local features, and enhancing the features of each location, respectively. To verify the validity of each module in the EAT block, we conduct an ablation experiment in Table 7 that contains different component combinations. Results indicate that each component contributes to the model performance, and our EATFormer obtains the best result when using all three parts. Since FFN takes up most of the parameters and calculations, we can conduct further research on optimizing this module to obtain better-integrated model performance.

5.4.2 Separation Ratio of GLI

We deduce from Eqs. (12) and (13) in Sect. 4.3.2 that EATFormer has the lowest number of parameters and calculation

amount when separation ratio p of GLI equals 0.2, and there is not much difference about the total parameters and calculations when p lies in the range $[0, 0.5]$. To further prove the above analysis and verify the validity of the GLI, we conduct a set of experiments with equal interval sampling of p in range $[0, 1]$ for the classification task. As shown in Fig. 8, the x-coordinate represents different proportions, and the left y-ordinate represents Top-1 accuracy of the modified EATFormer-Tiny with embedding dims equaling $[64, 128, 230, 320]$ for divisible channels. The right y-ordinate shows the model's running speed and relative computation amount. Results in the figure are consistent with the foregoing derivation, and p equaling 0.5 is the most economical and efficient choice, where the model has relatively high precision, fast speed, and low computational cost. All GLI layers in this article use the same ratio, and exploring different ratios for different layers should lead to further improvements based on the above analysis.

5.4.3 Component Ablation of EATFormer

Following the core idea of paralleling global and local modelings, this paper extends a pyramid architecture over the previous columnar EAT model (Zhang et al., 2021). Specifically, EAT block-based EATFormer can be seen as evolving from the naive baseline, which employs: (1) patch embedding for down-sampling; (2) MSRA with only one scale; (3) naive MSA; (4) simple addition operation with $\alpha_i, i = 1, \dots, N, g, l$ equaling 1, instead of: (1) MSRA for down-sampling; (2) MSRA with multiple scale; (3) improved MD-MSA; (4) weighted operation mixing (WOM) with learnable $\alpha_i, i = 1, \dots, N, g, l$. Detail ablation experiment

Table 8 Ablation study for different component combinations in EATFormer. ✓ means choice while ✗ is the opposite, and + represents D-MSA that abandons the modulation operation

MSRA Down	MSRA	MD-MSA	WOM	Param	FLOPs	Top-1
✗	✗	✗	✗	4.792	1.232	77.4
✓	✗	✗	✗	5.202	1.300	77.8
✗	✓	✗	✗	5.208	1.283	77.9
✗	✗	+	✗	4.804	1.236	77.5
✗	✗	✓	✗	4.805	1.236	77.7
✓	✓	✗	✗	6.109	1.412	78.2
✓	✗	✓	✗	5.214	1.304	78.0
✗	✓	✓	✗	5.220	1.288	78.1
✓	✓	✓	✗	6.122	1.416	78.2
✓	✓	✗	✓	6.109	1.412	78.1
✗	✓	✓	✓	5.221	1.288	78.0
✓	✓	✓	✓	6.122	1.416	78.4

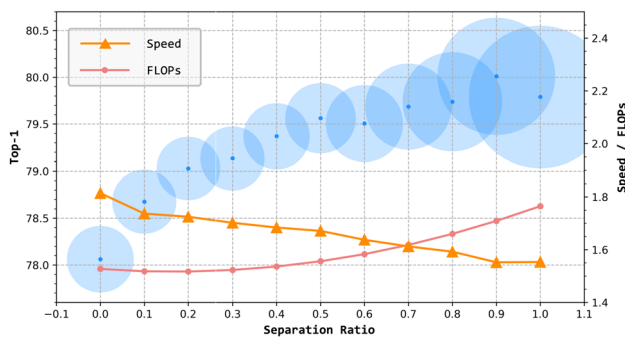


Fig. 8 Separation ratio analysis of GLI. Blue circle represents the Top-1 accuracy of EATFormer at different separation ratios (c.f., left axis), and the radius represents the relative number of parameters. Orange and pink lines represent the running speed and relative FLOPs of the model (c.f., right axis)

based on EATFormer-tiny can be viewed in Table 8, and the results indicate that each individual component has a role, and different components combination can complement each other to help the model achieve higher results. Note that WOM can only be applied if multi-path-based MSRA is used.

5.4.4 Composition of GLI

By default, the global path in GLI employs the designed MD-MSA module inspired by the dynamic population concept, while the local branch uses conventional CNN to model static feature extraction. To further assess the potential of the GLI module, different combinations of global (i.e., MSA and MD-MSA) and local (i.e., CNN and DCNv2, Zhu et al., 2019) operators are used for experiments. As shown in Table 9, MD-MSA improves the model effect by 0.3 ↑ only with negligible parameters and computation, while DCNv2 can further boost performance by a large margin at the cost of higher storage and computation. Theoretically, MD-MSA has no significant impact on the speed, but the naive PyTorch implementation

Table 9 Ablation study for compositions of GLI

Global	Local	Param	FLOPs	GPU	Top-1
MSA	CNN	6.1	1.412	1896	78.1
MSA	DCNv2	9.0	1.522	1567	79.0
MD-MSA	CNN	6.1	1.416	1549	78.4
MD-MSA	DCNv2	9.0	1.526	1333	79.2

without CUDA acceleration leads to a obvious decrease in GPU speed. Therefore, the running speed of our model could be improved after further optimization for MD-MSA.

5.4.5 Normalization Type

Transformer-based vision models generally use *Layer Normalization* (LN) to achieve better results rather than *Batch Normalization* (BN). Nevertheless, considering that LN requires slightly more computation than BN and the proposed hybrid EATFormer contains many convolutions that are usually combined with *Batch Normalization* (BN) layers, we conduct an ablation study to evaluate which normalization would be better. Table 10 shows the results on three EATFormer variants, and BN-normalized EATFormer achieves slightly better results while owing an significantly faster GPU inference speed. Note that merging convolution and BN layers is not used here, and this technique can further improve the inference speed.

5.4.6 MSRA at Different Stages

Different network depths may have different requirements for the MSRA module, so we explore the introduction of MSRA at different stages. As shown in Table 11, our model obtains the best result when MSRA is used in [2, 3, 4] stages, and the model effect decreases sharply when only used in the

Table 10 Effects of different normalization types

Network	Params	FLOPs	GPU	Top-1
Tiny-LN	6.1	1.425	963	78.2
Tiny-BN	6.1	1.416	1549	78.4
Small-LN	24.3	4.337	448	82.8
Small-BN	24.3	4.320	615	83.1
Base-LN	49.0	8.775	240	83.7
Base-BN	49.0	8.744	345	83.9

Table 11 Ablation study of MSRA on different stages

Stages	Params	FLOPs	GPU	Top-1
[1, 2, 3, 4]	6.3	1.541	1291	78.2
[2, 3, 4]	6.3	1.533	1434	78.5
[3, 4]	6.1	1.416	1549	78.4
[4]	5.6	1.326	1695	77.9

Table 12 Ablation study on kernel size of MSRA

Size	Params	FLOPs	GPU	Top-1
3×3	6.1	1.416	1549	78.4
5×5	9.0	1.845	1342	78.5
7×7	13.4	2.487	1087	78.5

fourth stage. Considering the model accuracy and efficiency, using this module in [3, 4] stages is a better choice.

5.4.7 Kernel Size of MSRA

The MSRA module for multi-scale modeling adopts CNN as its primary component so that the convolution kernel may influence the model results. As shown in Table 12, a larger kernel size can only slightly increase the model effect, but the number of parameters and the amount of calculation could increase dramatically. Therefore, we employ efficient 3×3 kernel size in MSRA for EATFormer at all scales.

5.4.8 Layer Number of TRH

The Plug-and-play TRH module can easily be docked with the transformer backbone to obtain the task-related feature representation, and we take the classification task as an example to explore the effect of this module. As shown in Table 13, Top-1 accuracy is significantly improved by gradually increasing the number of TRH layers in the EATFormer-Tiny model, and the performance tends to saturation after two layers. Therefore, using two-layer TRH is the recommended choice to balance model effectiveness and efficiency. However, there is no noticeable improvement in

Table 13 Quantitative ablation study for the layer number of TRH

Network	Params	FLOPs	GPU	Top-1
Tiny	6.1	1.416	1549	78.4
Tiny +1	6.9 _{+0.8}	1.423	1495	78.7
Tiny +2	7.7 _{+1.6}	1.430	1461	79.1
Tiny +3	8.4 _{+2.3}	1.438	1423	79.2
Small +2	29.1 _{+4.8}	4.363	589	83.2
Base +2	55.3 _{+6.3}	9.001	316	83.9

the larger models, so the multi-task advantage of TRH for the larger model is more important than accuracy improvement.

5.5 EATFormer Explanation

5.5.1 Alpha Distribution of Different Depths

The weighted operation mixing mechanism can improve the model performance and objectively represent the model's attention to different branches at different depths. Based on EATFormer-tiny, we use 3-path MSRA along with 2-path GLI for each EAT block, and the alpha-indicated weight distribution after training is shown in Fig. 9. (1) For the MSRA module, the proportion of α_1 (i.e., dilation equals 1) in the same stage shows an increasing trend while the larger α_3 is the opposite, indicating that local feature extraction with stronger correlation (i.e., smaller scale) is more critical for the network. And weight mutation between adjacent stages is caused by a down-sampling operation that changes the feature distribution. In the last stage4, large scale paths have more weight because they need to model as much global information as possible to get proper classification results. But in general, the proportion of each branch is balanced, meaning that feature learning at all scales contributes to the network. Considering the amount of computation and the number of parameters, this also supports the experimental result about why only using MSRA for stage3/4 described in above Sect. 5.4.6. (2) For the GLI module, the global branch has more and more weight than the local branches as the network deepens, indicating that both branches are effective and complement each other: local CNN is more suitable for low-level feature extraction while the global transformer is better at high-level information fusion.

5.5.2 Attention Visualization

To better illustrate which parts of the image the model focuses on, Grad-CAM (Selvaraju et al., 2017) is applied to highlight concerning regions by our small model. As shown in Fig. 10, we visualize different images by column for ResNet-50 (He et al., 2016), Swin-B (Liu et al., 2021), and our

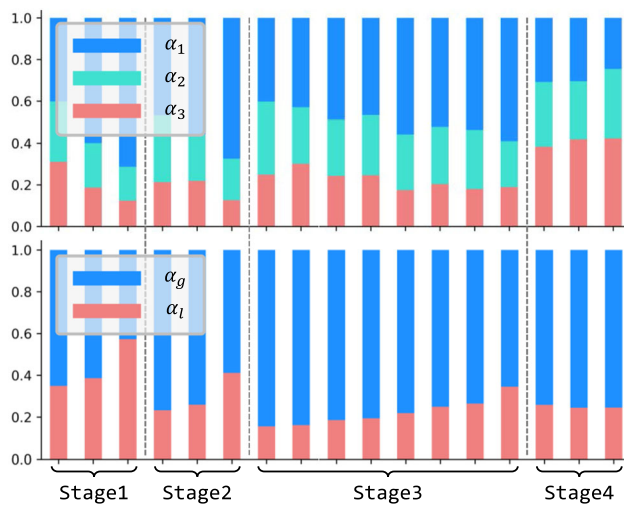


Fig. 9 Alpha distribution of the trained EATFormer for different depths. Top and bottom parts represent α_i , $i = 1, \dots, N$ in MSRA and α_i , $i = g, l$ in GLI, respectively

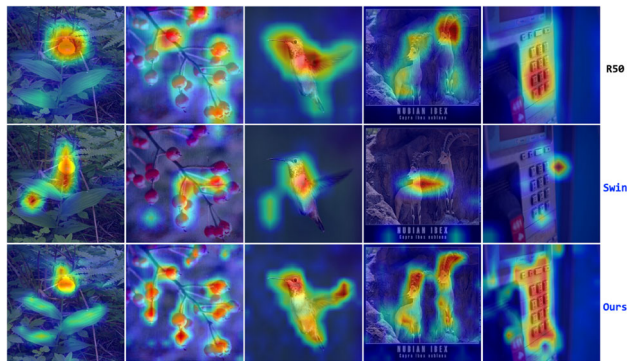


Fig. 10 Attention visualizations by Grad-CAM of our EATFormer compared to CNN-based ResNet-50 (He et al., 2016) and transformer-based Swin-B (Liu et al., 2021)

EATFormer-Base models, respectively. Results indicate that: (1) CNN-based ResNet tends to focus on as many regions as possible but ignores edges; (2) Transformer-based Swin pays more attention to sparse local areas; (3) Thanks to the design of MSRA and GLI modules, our EATFormer has more discriminative attention to subject targets that own very sharp edges.

5.5.3 Attention Distance of Global Path in GLI

We design the GLI module to explicitly model global and local information separately, so the local branch could undertake part of the short-distance modeling of the global branch. To verify this, we visualize the modeling distance of the global branch for our previous columnar EAT model (Zhang et al., 2021) and current studied EATFormer in Fig. 11: 1-Top) Compared with DeiT without local modeling, our EAT pays more attention to global information fusion (choosing layer

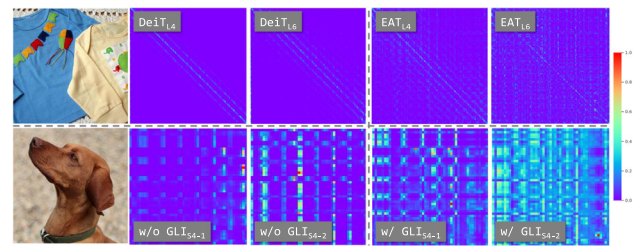


Fig. 11 Attention scope for the transformer-based global branch in GLI. The top part shows the attention maps for columnar DeiT (Touvron et al., 2021a) (w/o local modeling) and our previous EAT model (Zhang et al., 2021) (w/ local modeling) in different depths; the bottom part shows results of EATFormer w/ and w/o local modelings in the last stage

4/6 for examples), where more significant values are found at off-diagonal locations. 2-Bottom) Attention maps in the last stage are visualized because the window size equals the feature size that could cover overall information. When using global modeling alone (w/o GLI), the model only focuses on sparse regions but will pay attention to more regions when GLI is used. Results indicate that the designed parallel local path takes responsibility for some local modelings that should be the responsibility of the global path. We can find differences in feature modeling between columnar-alike and pyramid-aware architectures.

Relationship with EA. Motivated by EA variants (Moscato, 1989; Hart et al., 2005; Kumar et al., 2014) that introduce local search procedures besides conventional global search for converging higher-quality solutions, we analogically improve the novel GLI module. When GLI is not used (only global modeling), the model tends to correlate local regional features, which is consistent with the concept of local population in biological evolution due to geographical constraints, i.e., just like the concept of local search in EA. With GLI, explicit local modeling unlocks global modeling potential, forcing global branches to associate more distant features for better results, just as the global/local concept in EA (Moscato, 1989; Hart et al., 2005; Kumar et al., 2014) that improves performance.

5.5.4 Δl and Δm Distribution of MD-MSA

Figure 12 visualizes the learned offset (the longer the arrow, the farther the deformable distance, and the arrow direction indicates sampling direction) and modulation (the brighter the color, the greater the weight) of MD-MSA in stage4. There are differences in offset and modulation of each location in different depths, and the model unexpectedly tends to give more weight to the main object that could describe the main parts of the object. Since we set align_corners to true when resampling, it has a gradually increasing bias from 0 to 0.5 from the center to the edge. Therefore, the visualization results behave as a whole spreading outwards that may visu-

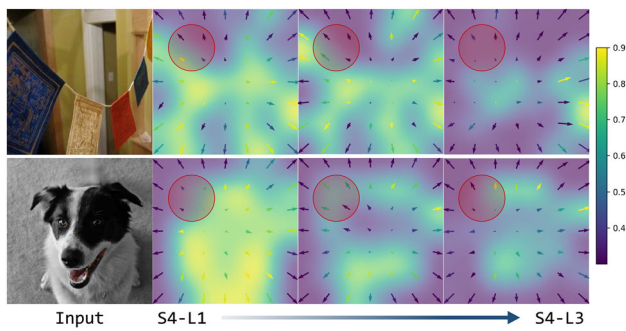


Fig. 12 Visualization of deformable offsets (denoted as arrows) and modulation scalar (denoted as color) for MD-MSA of our small model in the last stage. Please zoom in on the red circle for more details

ally weaken changes in each learned position. Please zoom in for better visualization.

Relationship with EA. Inspired by the irregular spatial distribution among real individuals that are not as horizontal and vertical as the image, we improve the novel MD-MSA module that considers the offset of each spatial position. As show in Fig. 12, different positions (individuals) prefer different offsets and modulation (i.e., direction and scale), just as individuals have different preferences in different regions of the biological world. This modeling method has also been verified in EA, e.g., the improved works (Liu & Lampinen, 2005; Opara & Arabas, 2019; Pant et al., 2020) adopt the similar parameter adaption and feature scaling idea to conduct global feature interaction.

5.5.5 Visualization of Attention Map in TRH

Taking the classification task as an example, we visualize the attention map in the two-layer TRH that contains multiple heads in the inner cross-attention layer. As shown in Fig. 13, we normalize values of attention maps to [0, 1] and draw them on the right side of the image. Results indicate that different heads focus on different regions, and the deeper TRH₂ focuses on a broader area than TRH₁ to form the final feature.

5.5.6 Parameters and FLOPs Distribution

Taking the designed EATFormer-Tiny as an example, we analyze the distribution of parameters and FLOPs in different layers, where the model contains a stem for resolution reduction, four stages for feature extraction, and a head for target output. As shown in Fig. 14, the number of parameters is mainly distributed in the deep stage3/4, while FLOPs concentrate in the early stages, and FFN occupies the majority of parameter number calculation. Therefore, we can focus on the optimization of the FFN structure to better balance the comprehensive model efficiency in future work.

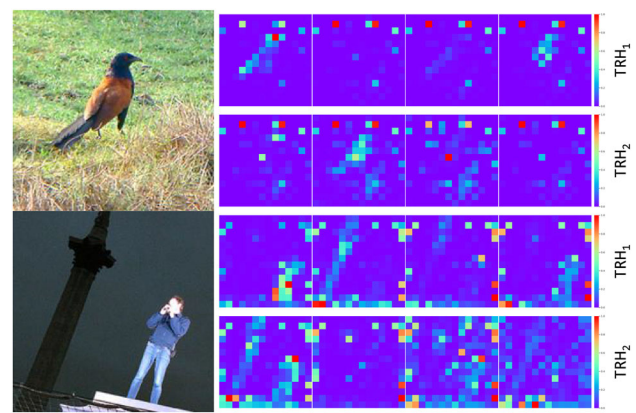


Fig. 13 Attention map visualization in TRH for the classification task. The model contains two TRHs, and four head attentions are displayed in each TRH

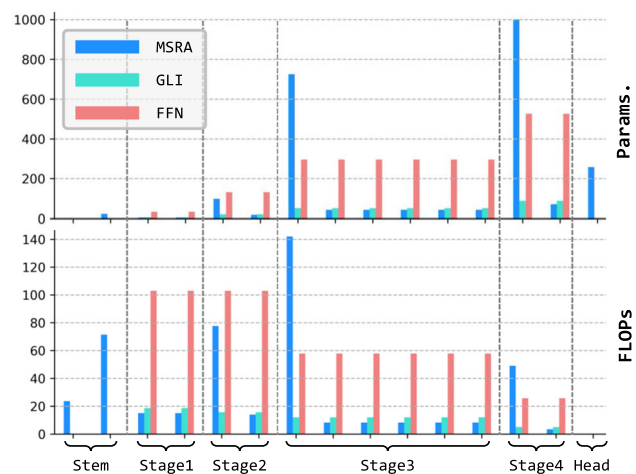


Fig. 14 Analyses of Params. and FLOPs distribution

5.5.7 Works Comparison with Local/Global Concepts

In this paper, the locality in ViT refers to the introduction of CNN with inductive bias into the Transformer structure, and we design the GLI block as a parallel structure that introduces a different local branch beside the global branch. This idea is motivated by some EA variants (Moscato, 1989; Hart et al., 2005; Kumar et al., 2014) that employ local search procedures besides conventional global search for converging higher-quality solutions. Also, global/local concept is only an idea in the macro sense, and the specific way varies from method to method. E.g., global/local concept in MPViT (Lee et al., 2022) is expressed as parallelism between blocks rather than within each block, while CMT (Guo et al., 2022) cascades local information into the FFN module rather than the MSA as previous works (Srinivas et al., 2021; Yuan et al., 2021a; Li et al., 2023b). Comparatively, our GLI block consists of local convolution and global MD-MSA operations, and Weighted Operation Mixing (WOM) mechanism is fur-

Table 14 Global/local idea comparisons among recent methods. ①: Whether the local/global concept is instantiated as a parallel block; ②: Whether the local operation is combined with MSA; ③: Whether only one kind of block is used for the model; ④: Whether feature split is employed for local/global paths; ⑤: Whether feature importance is considered for local/global paths. ✓: Satisfied; ✗: Unsatisfied

Method vs. Criterion	①	②	③	④	⑤
MPViT (Lee et al., 2022)	✗	✗	✓	✗	✗
CMT (Guo et al., 2022)	✗	✗	✓	✗	✗
UniFormer (Li et al., 2022)	✗	✓	✓	✗	✗
ViTAEv2 (Zhang et al., 2023b)	✓	✓	✗	✗	✗
MobileViT (Mehta & Rastegari, 2022)	✗	✓	✗	✗	✗
EdgeNeXt (Maaz et al., 2023)	✗	✓	✗	✗	✗
EfficientFormerv2 (Li et al., 2023b)	✗	✗	✗	✗	✗
EATFormer (Ours)	✓	✓	✓	✓	✓

ther proposed to mix all operations adaptively. So, *we argue that GLI obviously differs comparison methods*. Besides, we compare our method with some contemporary/recent works (Lee et al., 2022; Guo et al., 2022; Li et al., 2022; Zhang et al., 2023b; Mehta and Rastegari, 2022; Maaz et al., 2023; Li et al., 2023b) which incorporate the global/local concept to their model designs. To further illustrate the differences with these methods, we make comprehensive comparison with them in terms of local/global concepts by several criteria in Table 14). Results illustrates the uniqueness of GLI in the technical level.

6 Conclusion

This paper explains the rationality of vision transformer by analogy with EA and improves our previous columnar EAT to a novel pyramid EATFormer architecture inspired by effective EA variants. Specifically, the designed backbone consists only of the proposed EAT block that contains three residual parts, i.e., MSRA, GLI, and FFN modules, to model multi-scale, interactive, and individual information separately. Moreover, we propose a TRH module and *improve* an MD-MSA module to boost the effectiveness and usability of our EATFormer further. Abundant experiments on classification and downstream tasks demonstrate the superiority of our approach over SOTA methods in terms of accuracy and efficiency, while ablation and explanatory experiments further illustrate the effectiveness of EATFormer and each analogically designed component.

Nevertheless, we do not use larger models (e.g., > 100M), larger datasets (i.e., ImageNet-21K Deng et al., 2009) or stronger training strategy (i.e., token labeling Jiang et al., 2021b) for experiments due to limited amount of computation. Also, the architecture recipes are mainly given by our

intuition, and the super-parameter could be used to optimize the model structure further. We will explore the above aspects and the combination with self-supervised learning techniques in future works.

Acknowledgements This work was supported by a Grant from The National Natural Science Foundation of China (No. 62103363)

Data Availability All the datasets used in this paper are available online. ImageNet-1K (<http://image-net.org>), COCO 2017 (<https://cocodataset.org>), and ADE20K (<http://sceneparsing.csail.mit.edu>) can be downloaded from their official website accordingly.

References

- Ali, A., Touvron, H., Caron, M., Bojanowski, P., Douze, M., Joulin, A., Laptev, I., Neverova, N., Synnaeve, G., Verbeek, J., & Jegou H (2021). Xcit: Cross-covariance image transformers. In *NeurIPS*.
- Atito, S., Awais, M., & Kittler, J. (2021). Sit: Self-supervised vision transformer. arXiv preprint [arXiv:2104.03602](https://arxiv.org/abs/2104.03602)
- Baevski, A., Hsu, W.N., Xu, Q., Babu, A., Gu, J., & Auli, M. (2022). Data2vec: A general framework for self-supervised learning in speech, vision and language. In *ICML*.
- Bao, H., Dong, L., Piao, S., & Wei, F. (2022). BEit: BERT pre-training of image transformers. In *ICLR*.
- Bartz-Beielstein, T., Branke, J., Mehnen, J., & Mersmann, O. (2014). *Evolutionary algorithms*. Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery.
- Bello, I. (2021). Lambdanetworks: Modeling long-range interactions without attention. In *ICLR*.
- Bertasius, G., Wang, H., & Torresani, L. (2021). Is space-time attention all you need for video understanding? In *ICML*.
- Bhojanapalli, S., Chakrabarti, A., Glasner, D., Li, D., Unterthiner, T., & Veit, A. (2021). Understanding robustness of transformers for image classification. In *ICCV*.
- Bhowmik, P., Pantho, M. J. H., & Bobda, C. (2021). Bio-inspired smart vision sensor: Toward a reconfigurable hardware modeling of the hierarchical processing in the brain. *Journal of Real-Time Image Processing*, 18, 157–174.
- Brest, J., Greiner, S., Boskovic, B., Mernik, M., & Zumer, V. (2006). Self-adapting control parameters in differential evolution: A comparative study on numerical benchmark problems. In *TEC*.
- Brest, J., Zamuda, A., Boskovic, B., Maucec, M. S., & Zumer, V. (2008). High-dimensional real-parameter optimization using self-adaptive differential evolution algorithm with population size reduction. In *CEC*.
- Brest, J., Zamuda, A., Fister, I., & Maučec, M. S. (2010). Large scale global optimization using self-adaptive differential evolution algorithm. In *CEC*.
- Brown, T., Mann, B., Ryder, N., Subbiah, M., Kaplan, J.D., Dhariwal, P., Neelakantan, A., Shyam, P., Sastry, G., Askell, A., & Agarwal, S. (2020). Language models are few-shot learners. In *NeurIPS*.
- Carion, N., Massa, F., Synnaeve, G., Usunier, N., Kirillov, A., & Zagoruyko, S. (2020). End-to-end object detection with transformers. In: *ECCV*.
- Caron, M., Touvron, H., Misra, I., Jégou, H., Mairal, J., Bojanowski, P., & Joulin, A. (2021). Emerging properties in self-supervised vision transformers. In *ICCV*.
- Chen, B., Li, P., Li, C., Li, B., Bai, L., Lin, C., Sun, M., Yan, J., & Ouyang, W. (2021). Glit: Neural architecture search for global and local image transformer. In *ICCV*.

- Chen, H., Wang, Y., Guo, T., Xu, C., Deng, Y., Liu, Z., Ma, S., Xu, C., Xu, C., & Gao, W. (2021). Pre-trained image processing transformer. In *CVPR*.
- Chen, J., Lu, Y., Yu, Q., Luo, X., Adeli, E., Wang, Y., Lu, L., Yuille, A. L., & Zhou, Y. (2021). Transunet: Transformers make strong encoders for medical image segmentation. arXiv preprint [arXiv:2102.04306](https://arxiv.org/abs/2102.04306)
- Chen, K., Wang, J., Pang, J., Cao, Y., Xiong, Y., Li, X., Sun, S., Feng, W., Liu, Z., Xu, J., Zhang, Z., Cheng, D., Zhu, C., Cheng, T., Zhao, Q., Li, B., Lu, X., Zhu, R., Wu, Y., Dai, J., Wang, J., Shi, J., Ouyang, W., Loy, C. C., & Lin, D. (2019). MMDetection: Open mmlab detection toolbox and benchmark. arXiv preprint [arXiv:1906.07155](https://arxiv.org/abs/1906.07155)
- Chen, M., Peng, H., Fu, J., & Ling, H. (2021). Autoformer: Searching transformers for visual recognition. In *ICCV*.
- Chen, M., Wu, K., Ni, B., Peng, H., Liu, B., Fu, J., Chao, H., & Ling, H. (2021). Searching the search space of vision transformer. In *NeurIPS*.
- Chen, Q., Wu, Q., Wang, J., Hu, Q., Hu, T., Ding, E., & Cheng, J., Wang, J. (2022). Mixformer: Mixing features across windows and dimensions. In *CVPR*.
- Chen, T., Saxena, S., Li, L., Fleet, D. J., & Hinton, G. (2022). Pix2seq: A language modeling framework for object detection. In *ICLR*.
- Chen, X., Ding, M., Wang, X., Xin, Y., Mo, S., Wang, Y., Han, S., Luo, P., Zeng, G., & Wang, J. (2023). Context autoencoder for self-supervised representation learning. In *IJCV*.
- Chen, X., Xie, S., & He, K. (2021). An empirical study of training self-supervised visual transformers. In *ICCV*.
- Chen, Y., Dai, X., Chen, D., Liu, M., Dong, X., Yuan, L., & Liu, Z. (2022). Mobile-former: Bridging mobilenet and transformer. In *CVPR*.
- Chen, Z., & Kang, L. (2005). Multi-population evolutionary algorithm for solving constrained optimization problems. In *AIAl*.
- Chen, Z., Zhu, Y., Zhao, C., Hu, G., Zeng, W., Wang, J., & Tang, M. (2021). Dpt: Deformable patch-based transformer for visual recognition. In *ACM MM*.
- Cheng, B., Misra, I., Schwing, A. G., Kirillov, A., & Girdhar, R. (2022). Masked-attention mask transformer for universal image segmentation. In *CVPR*.
- Cheng, B., Schwing, A., & Kirillov, A. (2021). Per-pixel classification is not all you need for semantic segmentation. In *NeurIPS*.
- Choromanski, K. M., Likhoshervstov, V., Dohan, D., Song, X., Gane, A., Sarlos, T., Hawkins, P., Davis, J. Q., Mohiuddin, A., Kaiser, L., Belanger, D. B., Colwell, L. J., & Weller, A. (2021). Rethinking attention with performers. In *ICLR*.
- Chu, X., Tian, Z., Wang, Y., Zhang, B., Ren, H., Wei, X., Xia, H., & Shen, C. (2021). Twins: Revisiting the design of spatial attention in vision transformers. In *NeurIPS*.
- Chu, X., Tian, Z., Zhang, B., Wang, X., Wei, X., Xia, H., & Shen, C. (2023). Conditional positional encodings for vision transformers. In *ICLR*.
- Coello, C. A. C., & Lamont, G. B. (2004). *Applications of multi-objective evolutionary algorithms* (Vol. 1). World Scientific.
- Cordonnier, J. B., Loukas, A., & Jaggi, M. (2020). On the relationship between self-attention and convolutional layers. In *ICLR*.
- Dai, J., Qi, H., Xiong, Y., Li, Y., Zhang, G., Hu, H., & Wei, Y. (2017). Deformable convolutional networks. In *ICCV*.
- Das, S., & Suganthan, P. N. (2010). Differential evolution: A survey of the state-of-the-art. *TEC*.
- Deng, J., Dong, W., Socher, R., Li, L. J., Li, K., & Fei-Fei, L. (2009). Imagenet: A large-scale hierarchical image database. In *CVPR*.
- Devlin, J., Chang, M. W., Lee, K., & Toutanova, K. (2019). Bert: Pre-training of deep bidirectional transformers for language understanding. In *NAACL*.
- Dong, X., Bao, J., Chen, D., Zhang, W., Yu, N., Yuan, L., Chen, D., & Guo, B. (2022). Cswin transformer: A general vision transformer backbone with cross-shaped windows. In *CVPR*.
- Dong, X., Bao, J., Zhang, T., Chen, D., Zhang, W., Yuan, L., Chen, D., Wen, F., Yu, N., & Guo, B. (2023). Peco: Perceptual codebook for Bert pre-training of vision transformers. In *AAAI*.
- Dong, Y., Cordonnier, J. B., & Loukas, A. (2021). Attention is not all you need: Pure attention loses rank doubly exponentially with depth. In *ICML*.
- Dosovitskiy, A., Beyer, L., Kolesnikov, A., Weissenborn, D., Zhai, X., Unterthiner, T., Dehghani, M., Minderer, M., Heigold, G., Gelly, S., Uszkoreit, J., & Houlsby, N. (2021). An image is worth 16 × 16 words: Transformers for image recognition at scale. In *ICLR*.
- d'Ascoli, S., Touvron, H., Leavitt, M. L., Morcos, A. S., Biroli, G., & Sagun, L. (2021). Convit: Improving vision transformers with soft convolutional inductive biases. In *ICML*.
- Fang, Y., Liao, B., Wang, X., Fang, J., Qi, J., Wu, R., Niu, J., & Liu, W. (2021). You only look at one sequence: Rethinking transformer in vision through object detection. In *NeurIPS*.
- Felleman, D. J., & Van Essen, D. C. (1991). *Distributed hierarchical processing in the primate cerebral cortex*. Cerebral Cortex.
- Gao, P., Ma, T., Li, H., Lin, Z., Dai, J., & Qiao, Y. (2022). Mcmae: Masked convolution meets masked autoencoders. In *NeurIPS*.
- García-Martínez, C., & Lozano, M. (2008). Local search based on genetic algorithms. In *Advances in metaheuristics for hard optimization*. Springer.
- Goyal, A., & Bengio, Y. (2022). Inductive biases for deep learning of higher-level cognition. *Proceedings of the Royal Society A*, 478, 20210068.
- Guo, J., Han, K., Wu, H., Tang, Y., Chen, X., Wang, Y., & Xu, C. (2022). Cmt: Convolutional neural networks meet vision transformers. In *CVPR*.
- Guo, M. H., Lu, C. Z., Liu, Z. N., Cheng, M. M., & Hu, S. M. (2023). Visual attention network. In *CVM*.
- Han, K., Xiao, A., Wu, E., Guo, J., Xu, C., & Wang, Y. (2021). Transformer in transformer. In *NeurIPS*.
- Hao, Y., Dong, L., Wei, F., & Xu, K. (2021). Self-attention attribution: Interpreting information interactions inside transformer. In *AAAI*.
- Hart, W. E., Krasnogor, N., & Smith, J. E. (2005). Memetic evolutionary algorithms. In *Recent advances in memetic algorithms* (pp. 3–27). Springer.
- Hassanat, A., Almohammadi, K., Alkafaween, E., Abunawas, E., Hammouri, A., & Prasath, V. (2019). Choosing mutation and crossover ratios for genetic algorithms—a review with a new dynamic approach. *Information*, 10, 390.
- Hassani, A., Walton, S., Li, J., Li, S., & Shi, H. (2023). Neighborhood attention transformer. In *CVPR*.
- He, K., Chen, X., Xie, S., Li, Y., Dollár, P., & Girshick, R. (2022). Masked autoencoders are scalable vision learners. In *CVPR*.
- He, K., Zhang, X., Ren, S., & Sun, J. (2016). Deep residual learning for image recognition. In *CVPR*.
- He, R., Ravula, A., Kanagal, B., & Ainslie, J. (2020). Realformer: Transformer likes residual attention. arXiv preprint [arXiv:2012.11747](https://arxiv.org/abs/2012.11747).
- Howard, A., Sandler, M., Chu, G., Chen, L. C., Chen, B., Tan, M., Wang, W., Zhu, Y., Pang, R., Vasudevan, V., & Le, Q. V. (2019). Searching for mobilenetv3. In *ICCV*.
- Huang, Z., Ben, Y., Luo, G., Cheng, P., Yu, G., & Fu, B. (2021). Shuffle transformer: Rethinking spatial shuffle for vision transformer. arXiv preprint [arXiv:2106.03650](https://arxiv.org/abs/2106.03650).
- Hudson, D. A., & Zitnick, L. (2021). Generative adversarial transformers. In *ICML*.
- Jiang, Y., Chang, S., & Wang, Z. (2021). Transgan: Two pure transformers can make one strong gan, and that can scale up. In *NeurIPS*.
- Jiang, Z. H., Hou, Q., Yuan, L., Zhou, D., Shi, Y., Jin, X., Wang, A., & Feng, J. (2021). All tokens matter: Token labeling for training better vision transformers. In *NeurIPS*.
- Katharopoulos, A., Vyas, A., Pappas, N., & Fleuret, F. (2020). Transformers are rnns: Fast autoregressive transformers with linear attention. In *ICML*.

- Khare, V., Yao, X., & Deb, K. (2003). Performance scaling of multi-objective evolutionary algorithms. In *EMO*.
- Kim, J., Nguyen, D., Min, S., Cho, S., Lee, M., Lee, H., & Hong, S. (2022). Pure transformers are powerful graph learners. In *NeurIPS*.
- Kitaev, N., Kaiser, L., & Levskaya, A. (2020). Reformer: The efficient transformer. In *ICLR*.
- Kolen, A., & Pesch, E. (1994). Genetic local search in combinatorial optimization. *Discrete Applied Mathematics*.
- Kumar, S., Sharma, V. K., & Kumari, R. (2014). Memetic search in differential evolution algorithm. arXiv preprint [arXiv:1408.0101](https://arxiv.org/abs/1408.0101).
- Land, M. W. S. (1998). *Evolutionary algorithms with local search for combinatorial optimization*. University of California.
- Lee, Y., Kim, J., Willette, J., & Hwang, S. J. (2022). Mpvit: Multi-path vision transformer for dense prediction. In *CVPR*.
- Li, C., Tang, T., Wang, G., Peng, J., Wang, B., Liang, X., & Chang, X. (2021). Bossnas: Exploring hybrid cnn-transformers with blockwisely self-supervised neural architecture search. In *ICCV*.
- Li, K., Wang, Y., Peng, G., Song, G., Liu, Y., Li, H., & Qiao, Y. (2022). Uniformer: Unified transformer for efficient spatial-temporal representation learning. In *ICLR*.
- Li, K., Wang, Y., Zhang, J., Gao, P., Song, G., Liu, Y., Li, H., & Qiao, Y. (2023). Uniformer: Unifying convolution and self-attention for visual recognition. *TPAMI*.
- Li, X., Wang, L., Jiang, Q., & Li, N. (2021). Differential evolution algorithm with multi-population cooperation and multi-strategy integration. *Neurocomputing*, 421, 285–302.
- Li, Y., Hu, J., Wen, Y., Evangelidis, G., Salahi, K., Wang, Y., Tulyakov, S., & Ren, J. (2023). Rethinking vision transformers for mobilenet size and speed. In *ICCV*.
- Li, Y., Zhang, K., Cao, J., Timofte, R., & Van Gool, L. (2021). Localvit: Bringing locality to vision transformers. arXiv preprint [arXiv:2104.05707](https://arxiv.org/abs/2104.05707).
- Liang, J., Cao, J., Sun, G., Zhang, K., Van Gool, L., & Timofte, R. (2021). Swinir: Image restoration using swin transformer. In *ICCV*.
- Lin, T. Y., Maire, M., Belongie, S., Hays, J., Perona, P., Ramanan, D., Dollár, P., & Zitnick, C. L. (2014). Microsoft coco: Common objects in context. In *ECCV*.
- Liu, J., & Lampinen, J. (2005). A fuzzy adaptive differential evolution algorithm. *Soft Computing*, 9, 448–462.
- Liu, Y., Li, H., Guo, Y., Kong, C., Li, J., & Wang, S. (2022). Rethinking attention-model explainability through faithfulness violation test. In *ICML*.
- Liu, Z., Hu, H., Lin, Y., Yao, Z., Xie, Z., Wei, Y., Ning, J., Cao, Y., Zhang, Z., Dong, L., & Wei, F. (2022). Swin transformer v2: Scaling up capacity and resolution. In *CVPR*.
- Liu, Z., Lin, Y., Cao, Y., Hu, H., Wei, Y., Zhang, Z., Lin, S., Guo, B. (2021). Swin transformer: Hierarchical vision transformer using shifted windows. In *ICCV*.
- Loshchilov, I., & Hutter, F. (2019). Decoupled weight decay regularization. In *ICLR*.
- Lu, J., Mottaghi, R., & Kembhavi, A. (2021). Container: Context aggregation networks. In *NeurIPS*.
- Maaz, M., Shaker, A., Cholakkal, H., Khan, S., Zamir, S. W., Anwer, R. M., & Shahbaz Khan, F. (2023). Edgenext: Efficiently amalgamated cnn-transformer architecture for mobile vision applications. In *ECCVW*.
- Mehta, S., & Rastegari, M. (2022). Mobilevit: Light-weight, general-purpose, and mobile-friendly vision transformer. In *ICLR*.
- Min, J., Zhao, Y., Luo, C., & Cho, M. (2022). Peripheral vision transformer. In *NeurIPS*.
- Moscato, P., et al. (1989). On evolution, search, optimization, genetic algorithms and martial arts: Towards memetic algorithms. *Caltech Concurrent Computation Program, C3P Report*, 826, 1989.
- Motter, B. C. (1993). Focal attention produces spatially selective processing in visual cortical areas v1, v2, and v4 in the presence of competing stimuli. *Journal of Neurophysiology*, 70, 909–919.
- Nakashima, K., Kataoka, H., Matsumoto, A., Iwata, K., Inoue, N., & Satoh, Y. (2022). Can vision transformers learn without natural images? In *AAAI*.
- Neimark, D., Bar, O., Zohar, M., & Asselmann, D. (2021). Video transformer network. In *ICCV*.
- Opara, K. R., & Arabas, J. (2019). Differential evolution: A survey of theoretical analyses. *Swarm and Evolutionary Computation*, 44, 546–558.
- Padhye, N., Mittal, P., & Deb, K. (2013). Differential evolution: Performances and analyses. In *CEC*.
- Pan, X., Ge, C., Lu, R., Song, S., Chen, G., Huang, Z., & Huang, G. (2022). On the integration of self-attention and convolution. In *CVPR*.
- Pant, M., Zaheer, H., Garcia-Hernandez, L., Abraham, A., et al. (2020). Differential evolution: A review of more than two decades of research. *Engineering Applications of Artificial Intelligence*, 90, 103479.
- Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., Killeen, T., Lin, Z., Gimelshein, N., Antiga, L., & Desmaison, A. (2019). Pytorch: An imperative style, high-performance deep learning library. In *NeurIPS*.
- Peters, M. E., Neumann, M., Iyyer, M., Gardner, M., Clark, C., Lee, K., & Zettlemoyer, L. (2018). Deep contextualized word representations. In *ICLR*.
- Qiang, Y., Pan, D., Li, C., Li, X., Jang, R., & Zhu, D. (2022). Attcatt: Explaining transformers via attentive class activation tokens. In *NeurIPS*.
- Radford, A., Narasimhan, K., Salimans, T., Sutskever, I., & Sutskever, I. (2018). Improving language understanding by generative pre-training. OpenAI.
- Radford, A., Wu, J., Child, R., Luan, D., Amodei, D., Sutskever, I. (2019). Language models are unsupervised multitask learners. OpenAI blog.
- Raghu, M., Unterthiner, T., Kornblith, S., & Zhang, C. (2021). Dosovitskiy, A. Do vision transformers see like convolutional neural networks? In *NeurIPS*.
- Ren, S., Zhou, D., He, S., Feng, J., & Wang, X. (2022). Shunted self-attention via multi-scale token aggregation. In *CVPR*.
- Selvaraju, R. R., Cogswell, M., Das, A., Vedantam, R., & Parikh, D. (2017). Batra, D. Grad-cam: Visual explanations from deep networks via gradient-based localization. In *ICCV*.
- Shi, E. C., Leung, F. H., & Law, B. N. (2014). Differential evolution with adaptive population size. In *ICDSP*.
- Si, C., Yu, W., Zhou, P., Zhou, Y., Wang, X., & Yan, S. (2022). Inception transformer. In *NeurIPS*.
- Sloss, A. N., & Gustafson, S. (2020). 2019 evolutionary algorithms review. In *Genetic programming theory and practice XVII*.
- Srinivas, A., Lin, T. Y., Parmar, N., Shlens, J., Abbeel, P., & Vaswani, A. (2021). Bottleneck transformers for visual recognition. In *CVPR*.
- Storn, R., & Price, K. (1997). Differential evolution—a simple and efficient heuristic for global optimization over continuous spaces. *Journal of Global Optimization* 341–359.
- Tan, M., & Le, Q. (2019). Efficientnet: Rethinking model scaling for convolutional neural networks. In *ICML*.
- Thatipelli, A., Narayan, S., Khan, S., Anwer, R. M., Khan, F. S., & Ghanem, B. (2022). Spatio-temporal relation modeling for few-shot action recognition. In *CVPR*.
- Toffolo, A., & Benini, E. (2003). Genetic diversity as an objective in multi-objective evolutionary algorithms. *Evolutionary Computation*, 11, 151–167.
- Touvron, H., Cord, M., Douze, M., Massa, F., Sablayrolles, A., & Jégou, H. (2021). Training data-efficient image transformers & distillation through attention. In *ICML*.
- Touvron, H., Cord, M., Sablayrolles, A., Synnaeve, G., & Jégou, H. (2021). Going deeper with image transformers. In *ICCV*.

- Tu, Z., Talebi, H., Zhang, H., Yang, F., Milanfar, P., Bovik, A., & Li, Y. (2022). Maxvit: Multi-axis vision transformer. In *ECCV*.
- Valanarasu, J.M.J., Oza, P., Hacihaliloglu, I., & Patel, V. M. (2021). Medical transformer: Gated axial-attention for medical image segmentation. In *MICCAI*.
- Vaswani, A., Ramachandran, P., Srinivas, A., Parmar, N., Hechtman, B., & Shlens, J. (2021). Scaling local self-attention for parameter efficient visual backbones. In *CVPR*.
- Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, L. u., & Polosukhin, I. (2017). Attention is all you need. In *NeurIPS*.
- Vikhar, P. A. (2016). Evolutionary algorithms: A critical review and its future prospects. In *ICGTSPICC*.
- Wan, Z., Chen, H., An, J., Jiang, W., Yao, C., & Luo, J. (2022). Facial attribute transformers for precise and robust makeup transfer. In *CACV*.
- Wang, H., Wu, Z., Liu, Z., Cai, H., Zhu, L., Gan, C., & Han, S. (2020). Hat: Hardware-aware transformers for efficient natural language processing. In *ACL*.
- Wang, R., Chen, D., Wu, Z., Chen, Y., Dai, X., Liu, M., Jiang, Y. G., Zhou, L., & Yuan, L. (2022). Bevt: Bert pretraining of video transformers. In *CVPR*.
- Wang, S., Li, B., Khabsa, M., Fang, H., & Ma, H. (2020). Linformer: Self-attention with linear complexity. arXiv preprint [arXiv:2006.04768](https://arxiv.org/abs/2006.04768)
- Wang, W., Xie, E., Li, X., Fan, D.P., Song, K., Liang, D., Lu, T., Luo, P., & Shao, L. (2021). Pyramid vision transformer: A versatile backbone for dense prediction without convolutions. In *ICCV*.
- Wang, W., Xie, E., Li, X., Fan, D. P., Song, K., Liang, D., Lu, T., Luo, P., & Shao, L. (2022). Pvt v2: Improved baselines with pyramid vision transformer. *CVM*.
- Wang, W., Yao, L., Chen, L., Lin, B., Cai, D., He, X., & Liu, W. (2022). Crossformer: A versatile vision transformer hinging on cross-scale attention. In *ICLR*.
- Wang, Y., Yang, Y., Bai, J., Zhang, M., Bai, J., Yu, J., Zhang, C., Huang, G., & Tong, Y. (2021). Evolving attention with residual convolutions. In *ICML*.
- Wei, C., Fan, H., Xie, S., Wu, C. Y., Yuille, A., & Feichtenhofer, C. (2022). Masked feature prediction for self-supervised visual pre-training. In *CVPR*.
- Wightman, R. (2019). Pytorch image models. <https://github.com/wlightman/pytorch-image-models>
- Wightman, R., Touvron, H., & Jegou, H. (2021). Resnet strikes back: An improved training procedure in timm. In *NeurIPS*.
- Wu, H., Xiao, B., Codella, N., Liu, M., Dai, X., Yuan, L., & Zhang, L. (2021). Cvt: Introducing convolutions to vision transformers. In *ICCV*.
- Xia, Z., Pan, X., Song, S., Li, L.E., & Huang, G. (2022). Vision transformer with deformable attention. In *CVPR*.
- Xiao, T., Liu, Y., Zhou, B., Jiang, Y., & Sun, J. (2018). Unified perceptual parsing for scene understanding. In *ECCV*.
- Xie, E., Wang, W., Yu, Z., Anandkumar, A., Alvarez, J. M., & Luo, P. (2021). Segformer: Simple and efficient design for semantic segmentation with transformers. In *NeurIPS*.
- Xie, Z., Zhang, Z., Cao, Y., Lin, Y., Bao, J., Yao, Z., Dai, Q., & Hu, H. (2022). Simmim: A simple framework for masked image modeling. In *CVPR*.
- Xu, L., Yan, X., Ding, W., & Liu, Z. (2023). Attribution rollout: a new way to interpret visual transformer. *JAIHC*.
- Xu, M., Xiong, Y., Chen, H., Li, X., Xia, W., Tu, Z., & Soatto, S. (2021). Long short-term transformer for online action detection. In *NeurIPS*.
- Xu, W., Xu, Y., Chang, T., & Tu, Z. (2021). Co-scale conv-attentional image transformers. In *ICCV*.
- Xu, Y., Zhang, Q., Zhang, J., & Tao, D. (2021). Vitae: Vision transformer advanced by exploring intrinsic inductive bias. In *NeurIPS*.
- Yang, C., Wang, Y., Zhang, J., Zhang, H., Wei, Z., Lin, Z., & Yuille, A. (2022). Lite vision transformer with enhanced self-attention. In *CVPR*.
- Yu, W., Luo, M., Zhou, P., Si, C., Zhou, Y., Wang, X., Feng, J., & Yan, S. (2022). Metaformer is actually what you need for vision. In *CVPR*.
- Yuan, K., Guo, S., Liu, Z., Zhou, A., Yu, F., & Wu, W. (2021). Incorporating convolution designs into visual transformers. In *ICCV*.
- Yuan, L., Chen, Y., Wang, T., Yu, W., Shi, Y., Jiang, Z. H., Tay, F. E., Feng, J., & Yan, S. (2021). Tokens-to-token vit: Training vision transformers from scratch on imagenet. In *ICCV*.
- Yuan, L., Hou, Q., Jiang, Z., Feng, J., & Yan, S. (2022). Volo: Vision outlooker for visual recognition. In *TPAMI*.
- Yuan, Y., Fu, R., Huang, L., Lin, W., Zhang, C., Chen, X., & Wang, J. (2021). Hrformer: High-resolution vision transformer for dense predict. In *NeurIPS*.
- Zamir, S. W., Arora, A., Khan, S., Hayat, M., Khan, F. S., & Yang, M. H. (2022). Restormer: Efficient transformer for high-resolution image restoration. In *CVPR*.
- Zhang, J., Li, X., Li, J., Liu, L., Xue, Z., Zhang, B., Jiang, Z., Huang, T., Wang, Y., & Wang, C. (2023). Rethinking mobile block for efficient attention-based models. In *ICCV*.
- Zhang, J., Xu, C., Li, J., Chen, W., Wang, Y., Tai, Y., Chen, S., Wang, C., Huang, F., & Liu, Y. (2021). Analogous to evolutionary algorithm: Designing a unified sequence model. In *NeurIPS*.
- Zhang, Q., Xu, Y., Zhang, J., & Tao, D. (2023). Vitaev2: Vision transformer advanced by exploring inductive bias for image recognition and beyond. *IJCV*.
- Zheng, S., Lu, J., Zhao, H., Zhu, X., Luo, Z., Wang, Y., Fu, Y., Feng, J., Xiang, T., Torr, P. H., & Zhang, L. (2021). Rethinking semantic segmentation from a sequence-to-sequence perspective with transformers. In *CVPR*.
- Zhou, B., Zhao, H., Puig, X., Xiao, T., Fidler, S., Barriuso, A., & Torralba, A. (2019). Semantic understanding of scenes through the ade20k dataset. In *IJCV*.
- Zhou, D., Kang, B., Jin, X., Yang, L., Lian, X., Hou, Q., & Feng, J. (2021). Deepvit: Towards deeper vision transformer. arXiv preprint [arXiv:2103.11886](https://arxiv.org/abs/2103.11886)
- Zhu, X., Hu, H., Lin, S., & Dai, J. (2019). Deformable convnets v2: More deformable, better results. In *CVPR*.
- Zhu, X., Su, W., Lu, L., Li, B., Wang, X., & Dai, J. (2021). Deformable detr: Deformable transformers for end-to-end object detection. In *ICLR*.

Publisher's Note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.

Springer Nature or its licensor (e.g. a society or other partner) holds exclusive rights to this article under a publishing agreement with the author(s) or other rightsholder(s); author self-archiving of the accepted manuscript version of this article is solely governed by the terms of such publishing agreement and applicable law.