Agent Attention: On the Integration of Softmax and Linear Attention

Dongchen Han[1⋆], Tianzhu Ye[1⋆], Yizeng Han[1], Zhuofan Xia[1], Siyuan Pan[2], Pengfei Wan[2], Shiji Song[1], and Gao Huang[1⋆⋆]

[1] Department of Automation, Tsinghua University
[2] Kuaishou Technology

Abstract. The attention module is the key component in Transformers. While the global attention mechanism offers high expressiveness, its excessive computational cost restricts its applicability in various scenarios. In this paper, we propose a novel attention paradigm, **Agent Attention**, to strike a favorable balance between computational efficiency and representation power. Specifically, the Agent Attention, denoted as a quadruple (Q, A, K, V), introduces an additional set of agent tokens A into the conventional attention module. The agent tokens first act as the agent for the query tokens Q to aggregate information from K and V, and then broadcast the information back to Q. Given the number of agent tokens can be designed to be much smaller than the number of query tokens, agent attention is significantly more efficient than the widely adopted Softmax attention, while preserving global context modelling capability. Interestingly, we show that the proposed agent attention is equivalent to a generalized form of linear attention. Therefore, agent attention seamlessly integrates the powerful Softmax attention and the highly efficient linear attention. Extensive experiments demonstrate the effectiveness of agent attention with various vision Transformers and across diverse vision tasks, including image classification, object detection, semantic segmentation and image generation. Notably, agent attention has shown remarkable performance in high-resolution scenarios, owning to its linear attention nature. For instance, when applied to Stable Diffusion, our agent attention accelerates generation and substantially enhances image generation quality without any additional training. Code is available at https://github.com/LeapLabTHU/Agent-Attention.

Keywords: Attention mechanism · Agent attention · Vision Transformer

1 Introduction

Originating from natural language processing, Transformer models have rapidly gained prominence in the field of computer vision in recent years, achieving significant success in image classification [10, 12, 15, 35], object detection [5, 37], semantic segmentation [6, 43], and multimodal tasks [27, 28, 40].

⋆ Equal contribution.
⋆⋆ Corresponding Author.

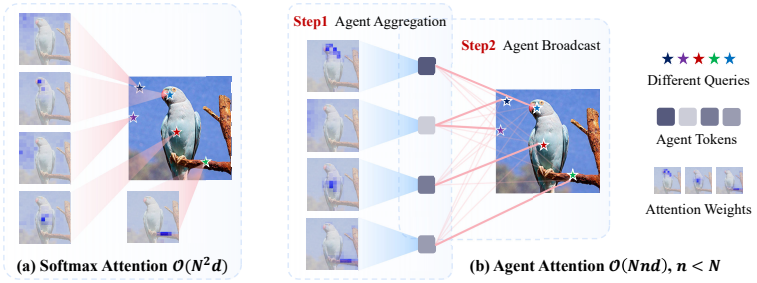


Fig. 1: An illustration of the motivation of our agent attention. (a) In Softmax attention, each query aggregates information from all features, incurring quadratic complexity. (b) Leveraging the redundancy between attention weights, agent attention uses a small number of agent tokens to act as the "agent" for queries, capturing diverse semantic information from all features, and then presenting it to each query. The attention weights are derived from DeiT-T and Agent-DeiT-T.

Nevertheless, incorporating Transformers and self-attention into the visual domain presents formidable challenges. Modern Transformer models commonly employ Softmax attention [36], which computes the similarity between each query-key pair, resulting in quadratic computation complexity with respect to the number of tokens. As a result, directly applying Softmax attention with global receptive fields to the visual tasks can lead to unmanageable computational demands. To tackle this issue, existing works [13, 14, 16, 24, 37, 41, 50] attempt to reduce computation complexity by designing efficient attention patterns. As two representatives, Swin Transformer [24] reduces the receptive field and confines self-attention calculations to local windows. PVT [37] employs a sparse attention pattern to alleviate the computational burden by reducing the number of keys and values. Despite their effectiveness, these methods inevitably compromise the capability to model long-range relationships, and are still inferior to global self-attention mechanism.

In this paper, in contrast to restricting receptive field or introducing sparsity, we propose a novel quadruplet attention paradigm (Q, A, K, V), dubbed **Agent Attention**, which exploits redundancy between attention weights to achieve both high model expressiveness and low computation complexity. As shown in Fig. 1, in Softmax attention, each query aggregates information from all features, incurring quadratic complexity. In fact, many queries, such as those denoting sky in Fig. 1a, require similar information. Therefore, our motivation is to eliminate the direct contact between each query and key, and instead use a small number of agent tokens A to act as the "agent" for queries, capturing diverse semantic information from all features, and then presenting it to each query. As illustrated in Fig. 1b and Fig. 2c, the resulting agent attention is composed of two conventional Softmax attention operations. The first Softmax attention treats agent tokens A as *queries* to aggregate agent features V_A from all values V, and the second utilizes agent tokens A as *keys*, broadcasting the global information

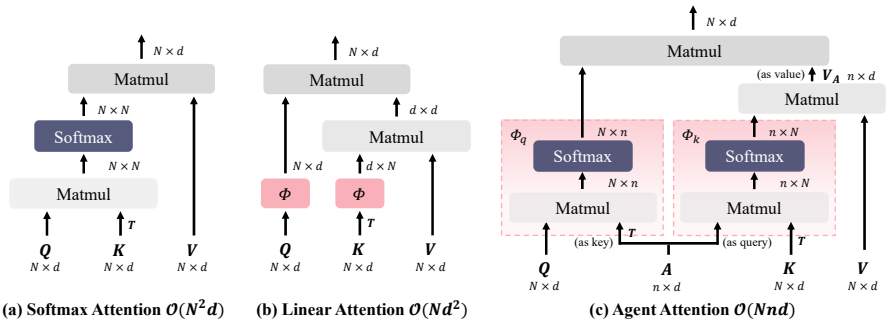


Fig. 2: Difference between Softmax attention, Linear attention and Agent attention. (a) Softmax attention computes the similarities between all query-key pairs, resulting in quadratic complexity. (b) Linear attention applies mapping function $\phi(\cdot)$ to Q and K respectively to change the computation order, reducing complexity but suffering from insufficient expressive capability. (c) Our Agent attention employs a small group of agent tokens to aggregate and broadcast global information, leading to an elegant integration of Softmax and linear attention and naturally enjoying the advantages of both high expressiveness and low computation complexity.

from agent features V_A to each query and forming the final output. Intuitively, the newly introduced tokens A serve as "agent" for the query tokens Q, as they directly collect information from K and V, and then deliver the result to Q.

Due to the intrinsic redundancy in global self-attention, the number of agent tokens can be designed to be much smaller than the number of query tokens. This property endows agent attention with high efficiency, reducing the quadratic complexity of Softmax attention to linear complexity while preserving global context modelling capability. Interestingly, as illustrated in Fig. 2, the proposed agent attention practically forms an elegant integration of Softmax and linear attention, which explains how it achieves both high efficiency and high expressiveness from a novel perspective.

We empirically verify the effectiveness of our model across diverse vision tasks, including image classification, object detection, semantic segmentation and image generation. Our method yields substantial improvements in various tasks, particularly in high-resolution scenarios. Noteworthy, our agent attention can be directly plugged into pre-trained large diffusion models, and without any additional training, it not only accelerates the generation process, but also notably improves the generation quality.

2 Related Works

Vision Transformer. Since the inception of Vision Transformer [10], self-attention has made notable strides in the realm of computer vision. However, the quadratic complexity of the prevalent Softmax attention [36] poses a challenge in applying self-attention to visual tasks. Previous works proposed various remedies

for this computational challenge. PVT [37] introduces sparse global attention, curbing computation cost by reducing the resolution of K and V. Swin Transformer [24] restricts self-attention computations to local windows and employs shifted windows to model the entire image. NAT [16] emulates convolutional operations and calculates attention within the neighborhood of each feature. DAT [41] designs a deformable attention module to achieve a data-dependent attention pattern. BiFormer [50] uses bi-level routing attention to dynamically determine areas of interest for each query. GRL [21] employs a mixture of anchored stripe attention, window attention, and channel attention to achieve efficient image restoration. However, these approaches inherently limit the global receptive field of self-attention or are vulnerable to specifically designed attention patterns, hindering their plug-and-play adaptability for general purposes.

Linear Attention. In contrast to the idea of restricting receptive fields, linear attention directly addresses the computational challenge by reducing computation complexity. The pioneer work [18] discards the Softmax function and replaces it with a mapping function ϕ applied to Q and K, thereby reducing the computation complexity to $\mathcal{O}(N)$. However, such approximations led to substantial performance degradation. To tackle this problem, Efficient Attention [33] applies the Softmax function to both Q and K. SOFT [26] and Nyströmformer [44] employ matrix decomposition to further approximate Softmax operation. Castling-ViT [47] uses Softmax attention as an auxiliary training tool and fully employs linear attention during inference. FLatten Transformer [11] proposes focused function and adopts depthwise convolution to preserve feature diversity. While these methods are effective, they continue to struggle with the limited expressive power of linear attention. In the paper, rather than enhancing Softmax or linear attention, we propose agent attention which integrates these two attention types, achieving superior performance in various tasks.

3 Preliminaries

In this section, we first review the general form of self-attention in vision Transformers and briefly analyze the pros and cons of Softmax and linear attention.

3.1 General Form of Self-Attention

With an input of N tokens represented as $x \in \mathbb{R}^{N \times C}$, self-attention can be formulated as follows in each head:

$$Q = xW_Q, K = xW_K, V = xW_V, \quad O_i = \sum_{j=1}^{N} \frac{\mathrm{Sim}(Q_i, K_j)}{\sum_{j=1}^{N} \mathrm{Sim}(Q_i, K_j)} V_j, \tag{1}$$

where $W_{Q/K/V} \in \mathbb{R}^{C \times d}$ are projection matrices, C and d are the channel dimension of module and each head, and $\mathrm{Sim}(\cdot, \cdot)$ denotes the similarity function.

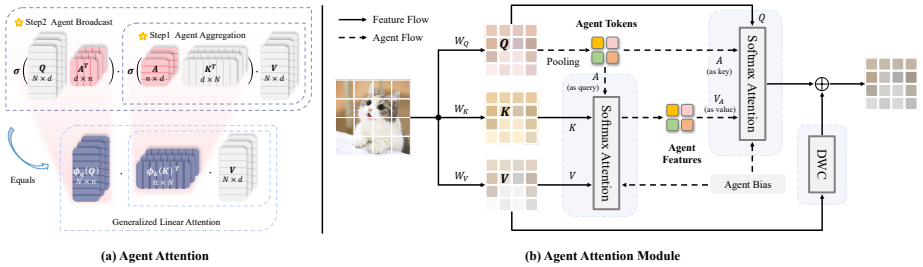


Fig. 3: An illustration of our agent attention and agent attention module. (a) Agent attention uses agent tokens to aggregate global information and distribute it to individual image tokens, resulting in a practical integration of Softmax and linear attention. $\sigma(\cdot)$ represents Softmax function. In (b), we depict the information flow of agent attention module. As a showcase, we acquire agent tokens through pooling. Subsequently, agent tokens are utilized to aggregate information from V, and Q queries features from the agent features. In addition, agent bias and DWC are adopted to add positional information and maintain feature diversity.

3.2 Softmax Attention and Linear Attention

When using $\mathrm{Sim}\,(Q,K) = \exp(QK^T/\sqrt{d})$ in Eq. (1), it becomes Softmax attention [36], which has been highly successful in modern vision Transformer designs. However, Softmax attention compels to compute the similarity between all query-key pairs, resulting in $\mathcal{O}(N^2)$ complexity. Consequently, using Softmax attention with a global receptive field leads to overwhelming computation complexity. To tackle this problem, previous works attempted to reduce the number of tokens N by designing sparse global attention [37, 38] or window attention [9, 24] patterns. While effective, these strategies unavoidably compromise the self-attention's capability for long-range modeling.

Comparably, linear attention [18] efficiently addresses the computation challenge with a linear complexity of $\mathcal{O}(N)$. Specifically, carefully designed mapping functions are applied to Q and K respectively, *i.e.*, $\mathrm{Sim}\,(Q,K) = \phi(Q)\phi(K)^T$. This gives us the opportunity to change the computation order from $(\phi(Q)\phi(K)^T)V$ to $\phi(Q)(\phi(K)^TV)$ based on the associative property of matrix multiplication. As illustrated in Fig. 2, by doing so, the computation complexity with respect to token number is reduced to $\mathcal{O}(N)$. However, designing effective mapping function $\phi(\cdot)$ proves to be a nontrivial task. Simple functions [33] such as ReLU lead to significant performance drop, whereas more intricate designs [7] or matrix decomposition methods [26, 44] may introduce extra computation overhead. In general, current linear attention approaches are still inferior to Softmax attention, limiting their practical application.

4 Agent Transformer

As discussed in Sec. 3, Softmax and linear attention suffer from either excessive computation complexity or insufficient model expressiveness. Previous research

commonly treated these two attention paradigms as distinct approaches and attempted to either reduce the computation cost of Softmax attention or enhance the performance of linear attention. In this section, we propose a new attention paradigm named **Agent Attention**, which practically forms an elegant integration of Softmax and linear attention, enjoying benefits from both linear complexity and high expressiveness.

4.1 Agent Attention

To simplify, we abbreviate Softmax and linear attention as:

$$O^{\mathrm{S}}=\sigma(QK^T)V \triangleq \mathrm{Attn}^{\mathrm{S}}(Q,K,V),\ O^{\phi}=\phi(Q)\phi(K)^TV \triangleq \mathrm{Attn}^{\phi}(Q,K,V), \quad (2)$$

where $Q, K, V \in \mathbb{R}^{N\times C}$ denote query, key and value matrices and $\sigma(\cdot)$ represents Softmax function. Then our agent attention can be written as:

$$O^{\mathrm{A}} = \underbrace{\mathrm{Attn}^{\mathrm{S}}(Q, A, \underbrace{\mathrm{Attn}^{\mathrm{S}}(A,K,V)}_{\text{Agent Aggregation}})}_{\text{Agent Broadcast}}. \quad (3)$$

It is equivalent to:

$$O^{\mathrm{A}} = \sigma(QA^T)\,\sigma(AK^T)\,V = \phi_q(Q)\phi_k(K)^TV = \underbrace{\mathrm{Attn}^{\phi_{\mathrm{q}/\mathrm{k}}}(Q,K,V)}_{\text{Generalized Linear Attn}}, \quad (4)$$

where $A \in \mathbb{R}^{n\times C}$ is our newly defined agent tokens.

As shown in Eq. (3) and Fig. 3a, our agent attention consists of two Softmax attention operations, namely agent aggregation and agent broadcast. Specifically, we initially treat agent tokens A as *queries* and perform attention calculations between A, K, and V to aggregate agent features V_A from all values. Subsequently, we utilize A as *keys* and V_A as *values* in the second attention calculation with the query matrix Q, broadcasting the global information from agent features to every query token and obtaining the final output O. In this way, we avoid the computation of pairwise similarities between Q and K while preserving information exchange between each query-key pair through agent tokens.

The newly defined agent tokens A essentially serve as the *agent* for Q, aggregating global information from K and V, and subsequently broadcasting it back to Q. Practically, we set the number of agent tokens n as a small hyperparameter, achieving a linear complexity of $\mathcal{O}(Nnd)$ relative to the number of input features N while maintaining global context modeling capability. Interestingly, as shown in Eq. (4) and Fig. 3a, we practically integrate the powerful Softmax attention and efficient linear attention, establishing a generalized linear attention paradigm by employing two Softmax attention operations, with the equivalent mapping function defined as $\phi_q(Q)=\sigma(QA^T), \phi_k(K)=\left(\sigma(AK^T)\right)^T$.

In practice, agent tokens can be acquired through different methods, such as simply setting as a set of learnable parameters or extracting from input features

through pooling or convolution. It is worth noticing that more advanced techniques like deformed points [41] or token merging [1] can also be used to obtain agent tokens. In the default setting, we employ the simple pooling strategy to obtain agent tokens, which already works surprisingly well.

4.2 Agent Attention Module

Agent attention inherits the merits of both Softmax and linear attention. In practical use, we further make two improvements to maximize its potential.

Agent Bias. In order to better utilize positional information, we present a carefully designed *Agent Bias* for our agent attention. Specifically, inspired by RPB [32], we introduce agent bias within the attention calculation, i.e.,

$$O^{\mathrm{A}} = \sigma(QA^T + B_2)\ \sigma(AK^T + B_1)\ V, \tag{5}$$

where $B_1 \in \mathbb{R}^{n\times N}, B_2 \in \mathbb{R}^{N\times n}$ are our agent biases. For parameter efficiency, we construct each agent bias using three bias components rather than directly setting B_1, B_2 as learnable parameters (see Appendix). Agent bias augments the vanilla agent attention with spatial information, helping different agent tokens to focus on diverse regions. As shown in Tab. 6, significant improvements can be observed upon the introduction of our agent bias terms.

Diversity Restoration Module. Although agent attention benefits from both low computation cost and high expressiveness, as generalized linear attention, it also suffers from insufficient feature diversity [11]. As a remedy, we follow [11] and adopt a depthwise convolution (DWC) module to preserve feature diversity.

Agent Attention Module. Building upon these designs, we propose a novel attention module named *Agent Attention Module*. As illustrated in Fig. 3(b), our module is composed of three parts, namely pure agent attention, agent bias and the DWC module. Our module can be formulated as:

$$O = \sigma(QA^T + B_2)\ \sigma(AK^T + B_1)\ V + \mathrm{DWC}(V), \tag{6}$$

where $Q, K, V \in \mathbb{R}^{N\times C}$, $A \in \mathbb{R}^{n\times C}$, $B_1 \in \mathbb{R}^{n\times N}$ and $B_2 \in \mathbb{R}^{N\times n}$. In the default setting, agent tokens A is obtained through pooling, i.e., $A = \mathrm{Pooling}(Q)$. The overall module complexity is expressed as:

$$\Omega = \underbrace{4NC^2}_{\text{Proj}} + \underbrace{NC}_{\text{Get Agents}} + \underbrace{2nNC + 2NnC}_{\text{Agent Attention}} + \underbrace{k^2NC}_{\text{DWC}}, \tag{7}$$

where N, n are the number of input features and agent tokens, and $k = 3$ is the kernel size of DWC. Notably, our model exhibits linear complexity for N.

Combining the merits of Softmax and linear attention, our module offers the following advantages:

(1) **Efficient computation and high expressive capability**. Previous work usually viewed Softmax attention and linear attention as two different attention paradigms, aiming to address their respective limitations. As a seamless

Table 1: ImageNet-1K classification results. The default input resolution is 224^2, except $^{\uparrow 384}$ denotes results on 384^2 resolution. Check the Appendix for full results.

Method	#Params	FLOPs	Top-1 Acc.
DeiT-T [35]	5.7M	1.2G	72.2
Agent-DeiT-T	6.0M	1.2G	**74.9** (+2.7)
DeiT-S	22.1M	4.6G	79.8
Agent-DeiT-S	22.7M	4.4G	**80.5** (+0.7)
PVT-T [37]	13.2M	1.9G	75.1
Agent-PVT-T	11.6M	2.0G	**78.4** (+3.3)
PVT-S	24.5M	3.8G	79.8
Agent-PVT-S	20.6M	4.0G	**82.2** (+2.4)
PVT-L	61.4M	9.8G	81.7
Agent-PVT-L	48.7M	10.4G	**83.7** (+2.0)

Method	#Params	FLOPs	Top-1 Acc.
Swin-T [24]	29M	4.5G	81.3
Agent-Swin-T	29M	4.5G	**82.6** (+1.3)
Swin-S	50M	8.7G	83.0
Agent-Swin-S	50M	8.7G	**83.7** (+0.7)
Swin-B	88M	15.4G	83.5
Agent-Swin-B	88M	15.4G	**84.0** (+0.5)
CSwin-B [9]	78M	15.0G	84.2
Agent-CSwin-B	73M	14.9G	**84.7** (+0.5)
CSwin-B$^{\uparrow 384}$	78M	47.0G	85.4
Agent-CSwin-B$^{\uparrow 384}$	73M	46.3G	**85.8** (+0.4)

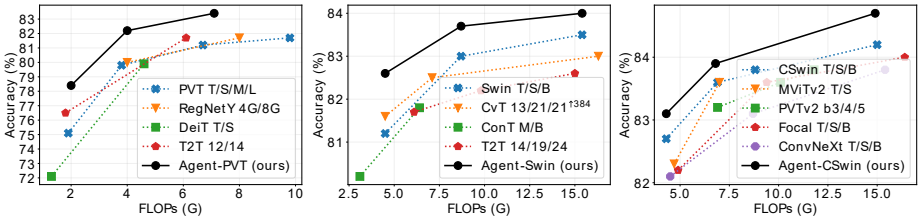


Fig. 4: Comparison with SOTA models [20, 25, 29, 35, 38, 39, 45, 46, 48] on ImageNet-1K.

integration of these two attention forms, our agent attention naturally inherits the merits of the two, enjoying both low computation complexity and high model expression ability at the same time.

(2) **Large receptive field**. Our module can adopt a large receptive field while maintaining the same amount of computation. Modern vision Transformer models typically resort to sparse attention [37, 38] or window attention [9, 24] to mitigate the computation burden of Softmax attention. Benefited from linear complexity, our model can enjoy the advantages of a large, even global receptive field while maintaining the same computation.

4.3 Implementation

Our agent attention module can serve as a plug-in module and can be easily adopted on a variety of modern vision Transformer architectures. As a showcase, we empirically apply our method to four advanced and representative Transformer models including DeiT [35], PVT [37], Swin [24] and CSwin [9]. We also apply agent attention to Stable Diffusion [30] to accelerate image generation. Detailed model architectures are shown in Appendix.

5 Experiments

To verify the effectiveness of our method, we conduct experiments on ImageNet-1K classification [8], ADE20K semantic segmentation [49], and COCO object

Table 2: Results on COCO dataset. The FLOPs are computed over backbone, FPN and detection head with an input resolution of 1280×800. See full results in Appendix.

(a) Mask R-CNN Object Detection

Method	FLOPs	Sch.	AP^b	AP^b_{50}	AP^b_{75}	AP^m	AP^m_{50}	AP^m_{75}
PVT-T	240G	1x	36.7	59.2	39.3	35.1	56.7	37.3
Agent-PVT-T	230G	1x	41.4	64.1	45.2	38.7	61.3	41.6
PVT-M	392G	1x	42.0	64.4	45.6	39.0	61.6	42.1
Agent-PVT-M	400G	1x	45.9	67.8	50.4	42.0	65.0	45.4
PVT-L	494G	1x	42.9	65.0	46.6	39.5	61.9	42.5
Agent-PVT-L	510G	1x	46.9	69.2	51.4	42.8	66.2	46.2
Swin-S	358G	1x	45.7	67.9	50.4	41.1	64.9	44.2
Agent-Swin-S	364G	1x	47.2	69.6	52.3	42.7	66.6	45.8

(b) Cascade Mask R-CNN Object Detection

Method	FLOPs	Sch.	AP^b	AP^b_{50}	AP^b_{75}	AP^m	AP^m_{50}	AP^m_{75}
Swin-T	745G	1x	48.1	67.1	52.2	41.7	64.4	45.0
Agent-Swin-T	755G	1x	49.2	68.6	53.2	42.7	65.6	45.9
Swin-T	745G	3x	50.4	69.2	54.7	43.7	66.6	47.3
Agent-Swin-T	755G	3x	51.4	70.2	55.9	44.5	67.6	48.4
Swin-S	837G	3x	51.9	70.7	56.3	45.0	68.2	48.8
Agent-Swin-S	843G	3x	52.6	71.3	57.1	45.5	68.9	49.2
Swin-B	981G	3x	51.9	70.5	56.4	45.0	68.1	48.9
Agent-Swin-B	990G	3x	52.6	71.1	57.1	45.3	68.6	49.2

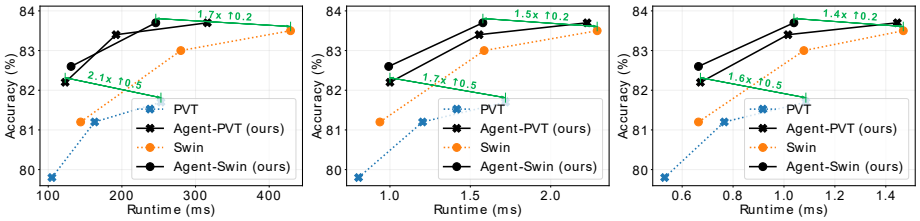


Fig. 5: Accuracy-Runtime curve on ImageNet. Runtime is tested with resolution 224^2.

detection [23]. Additionally, we integrate agent attention into the state-of-the-art generation model, Stable Diffusion [30]. Furthermore, we construct high-resolution models with large receptive fields to maximize the benefits of agent attention. In addition, sufficient ablation experiments are conducted to show the effectiveness of each design.

5.1 ImageNet-1K Classification

ImageNet [8] comprises 1000 classes, with 1.2 million training images and 50,000 validation images. We implement our module on four representative vision Transformers and compare the top-1 accuracy on the validation split with state-of-the-art models. See Appendix for **training settings**.

Results. As depicted in Tab. 1, substituting Softmax attention with agent attention in various models results in significant performance improvements. For instance, Agent-PVT-S surpasses PVT-L while using just 30% of the parameters and 40% of the FLOPs. Additionally, we provide a comprehensive comparison with various state-of-the-art methods in Fig. 4. Our models clearly achieve a better trade-off between computation cost and model performance. These results unequivocally prove that our approach has robust advantages and is adaptable to diverse architectures.

Inference Time. We further conduct real speed measurements by deploying the models on various devices. As Fig. 5 illustrates, our models attain inference speeds 1.7 to 2.1 times faster on the CPU while simultaneously improving per-

Table 3: Results of semantic segmentation on ADE20K. The FLOPs are computed over encoders and decoders with an input image at the resolution of 512×2048.

SemanticFPN Semantic Segmentation					
Backbone	Method	FLOPs	#Params	mIoU	mAcc
PVT-T	S-FPN	158G	17M	36.57	46.72
Agent-PVT-T	S-FPN	147G	15M	**40.18**	51.76
PVT-S	S-FPN	225G	28M	41.95	53.02
Agent-PVT-S	S-FPN	211G	24M	**44.18**	56.17
PVT-L	S-FPN	420G	65M	43.49	54.62
Agent-PVT-L	S-FPN	434G	52M	**46.52**	58.50

UperNet Semantic Segmentation					
Backbone	Method	FLOPs	#Params	mIoU	mAcc
Swin-T	UperNet	945G	60M	44.51	55.61
Agent-Swin-T	UperNet	954G	61M	**46.68**	58.53
Swin-S	UperNet	1038G	81M	47.64	58.78
Agent-Swin-S	UperNet	1043G	81M	**48.08**	59.78
Swin-B	UperNet	1188G	121M	48.13	59.13
Agent-Swin-B	UperNet	1196G	121M	**48.73**	60.01

formance. On RTX3090 GPU and A100 GPU, our models also achieve 1.4x to 1.7x faster inference speeds.

5.2 Object Detection

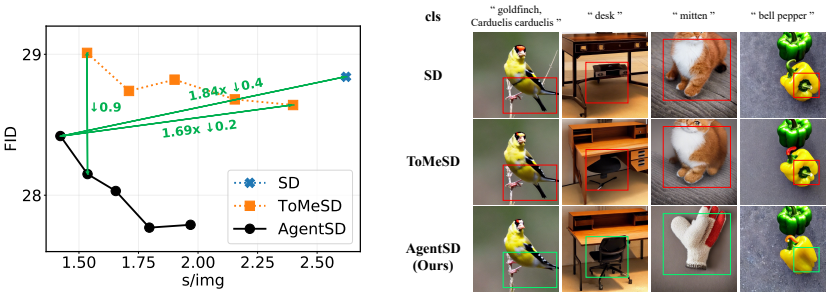
COCO [23] object detection and instance segmentation dataset has 118K training and 5K validation images. We apply our model to RetinaNet [22], Mask R-CNN [17] and Cascade Mask R-CNN [4] frameworks to evaluate the performance of our method. A series of experiments are conducted utilizing both 1x and 3x schedules with different detection heads. As depicted in Tab. 2, our model exhibits consistent enhancements across all configurations. Agent-PVT outperforms PVT models with an increase in box AP ranging from +**3.9** to +**4.7**, while Agent-Swin surpasses Swin models by up to +**1.5** box AP. These substantial improvements can be attributed to the large receptive field brought by our design, proving the effectiveness of agent attention in high-resolution scenarios.

5.3 Semantic Segmentation

ADE20K [49] is a well-established benchmark for semantic segmentation which encompasses 20K training images and 2K validation images. We apply our model to two exemplary segmentation models, namely SemanticFPN [19] and UperNet [42]. The results are presented in Tab. 3. Remarkably, our Agent-PVT-T and Agent-Swin-T achieve +**3.61** and +**2.17** higher mIoU than their counterparts. The results show that our model is compatible with various segmentation backbones and consistently achieves improvements.

5.4 Agent Attention for Stable Diffusion

The advent of diffusion models makes it possible to generate high-resolution and high-quality images. However, current diffusion models mainly use the original Softmax attention with a global receptive field, resulting in huge computation cost and slow generation speed. In the light of this, we apply our agent attention to Stable Diffusion [30], hoping to improve the generation speed of the model. Surprisingly, after simple adjustments, the Stable Diffusion model using agent



(a) Quantitative results **(b)** Samples

Fig. 6: (a) Quantitative Results of Stable Diffusion (SD), ToMeSD and our AgentSD. For ToMeSD, we take the merging ratios $\{0.1, 0.2, 0.3, 0.4, 0.5\}$ to construct five different models. Furthermore, we apply agent attention to each ToMeSD model to obtain the corresponding AgentSD model. (b) Samples generated by SD, ToMeSD ($r = 40\%$) and AgentSD ($r = 40\%$). The prompt is "A high quality photograph of a {cls}.".

attention, dubbed **AgentSD**, shows a significant improvement in generation speed and produces even better image quality *without any extra training*.

Applying agent attention to Stable Diffusion. We practically apply agent attention to ToMeSD model [3]. ToMeSD reduces the number of tokens before attention calculation in Stable Diffusion, enhancing generation speed. Nonetheless, the post-merge token count remains considerable, resulting in continued complexity and latency. Hence, we replace the Softmax attention employed in ToMeSD model with our agent attention to further enhance speed. We experimentally find that when producing agent tokens through token merging [1], our agent attention can be directly applied to Stable Diffusion and ToMeSD model without any extra training. However, we are unable to apply the agent bias and DWC in this way. As a remedy, we make two simple adjustments to the agent attention, which are described in detail in Appendix. In addition, we get a significant boost by applying agent attention during early diffusion generation steps and keeping the later steps unchanged.

Quantitative Results. We follow [3] and quantitatively compare AgentSD with Stable Diffusion and ToMeSD. As displayed in Fig. 6a, ToMeSD accelerates Stable Diffusion while maintaining similar image quality. AgentSD not only further accelerates ToMeSD but also significantly enhances image generation quality. Specifically, while maintaining superior image generation quality, AgentSD achieves 1.84x and 1.69x faster generation speeds compared to Stable Diffusion and ToMeSD, respectively. At an equivalent generation speed, AgentSD produces samples with a 0.9 lower FID score compared to ToMeSD. See the experimental details and full comparison table in Appendix.

Visualization. We present some visualizations in Fig. 6b. AgentSD noticeably reduces ambiguity and generation errors in comparison to Stable Diffusion and ToMeSD. For instance, in the first column, Stable Diffusion and ToMeSD produce birds with one leg and two tails, while AgentSD's sample does not exhibit

Table 4: Ablation on window size based on Agent-Swin-T.

	Window	FLOPs	#Param	Acc.	Diff.
Agent-Swin-T	7^2	4.5G	29M	82.0	-0.6
	14^2	4.5G	29M	82.2	-0.4
	28^2	4.5G	29M	82.4	-0.2
	56^2	4.5G	29M	**82.6**	**Ours**
Swin-T	7^2	4.5G	29M	81.3	-1.3

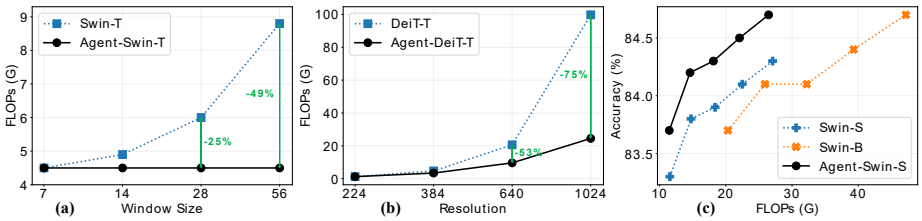


Fig. 7: (a) Comparison of FLOPs between Swin and our Agent-Swin as window size increases. (b) FLOPs comparison between DeiT and our Agent-DeiT in high-resolution scenarios. (c) Increasing resolution to $\{256^2, 288^2, 320^2, 352^2, 384^2\}$. All these models are finetuned for 30 epochs from the corresponding 224^2 resolution models.

this issue. In the third column, when provided with the prompt "A high quality photo of a mitten.", Stable Diffusion and ToMeSD erroneously generate a cat, whereas AgentSD produces the correct image.

AgentSD for finetuning. We apply agent attention to SD-based Dreambooth [31] to verify its performance under finetuning. When finetuned, agent attention can be integrated into all diffusion steps, reaching 2.2x acceleration in generation speed compared to the original Dreambooth. Refer to Appendix for details.

5.5 Large Receptive Field and High Resolution

Large Receptive Field. Modern vision Transformers often confine self-attention calculation to local windows to reduce computation complexity, such as Swin [24]. In Tab. 4, we gradually enlarge the window size of Agent-Swin-T, ranging from 7^2 to 56^2. Clearly, as the receptive field expands, the model's performance consistently improves. This indicates that while the window attention pattern is effective, it inevitably compromises the long-range modeling capability of self-attention and remains inferior to global attention. As shown in Fig. 7a, unlike the quadratic complexity of Softmax attention, the linear complexity of agent attention enables us to benefit from a global receptive field while preserving identical computation complexity.

High Resolution. Limited by the quadratic complexity of Softmax attention, current vision Transformer models usually scale up by increasing model depth and width. Building on insights from [34], we discover that enhancing resolution might be a more effective approach for scaling vision Transformers, particularly

Table 5: Scaling up by increasing resolution. All these models are trained from scratch.

Method	Reso	#Params	Flops	Top-1
DeiT-B [35]	224^2	86.6M	17.6G	81.8
DeiT-S	416^2	22.2M	18.8G	82.9 (+1.1)
Agent-DeiT-B	224^2	87.2M	17.6G	**82.0** (+0.2)
Agent-DeiT-S	448^2	23.1M	17.7G	**83.1** (+1.3)

Method	Reso	#Params	Flops	Top-1
PVT-L [37]	224^2	61.4M	9.8G	81.7
PVT-M	256^2	44.3M	8.8G	82.2 (+0.5)
Agent-PVT-L	224^2	48.7M	10.4G	**83.7** (+2.0)
Agent-PVT-M	256^2	36.1M	9.2G	**83.8** (+2.1)

Table 6: Ablation on each module of agent attention.

	FLOPs	#Param	Acc.	Diff.
Vanilla Linear Attention	4.5G	29M	77.8	-4.8
Agent Attention	4.5G	29M	79.0	-3.6
+ Agent Bias	4.5G	29M	81.1	-1.5
+ DWC	4.5G	29M	**82.6**	**Ours**
Swin-T w/o PE	4.5G	29M	80.1	-2.5
+ RPE	4.5G	29M	81.3	-1.3
+ DWC	4.5G	29M	81.6	-1.0

those employing agent attention with global receptive fields. As shown in Tab. 5, Agent-DeiT-B achieves a 0.2 accuracy gain compared to DeiT-B, whereas Agent-DeiT-S at 448^2 resolution attains an accuracy of 83.1 with only a quarter of the parameters. We observed analogous trends when scaling the resolution of Agent-PVT-M and Agent-Swin-S (see Appendix). Fig. 7b shows the FLOPs comparison between Agent-DeiT and DeiT, with Agent-DeiT saving 75% of FLOPs for 1024^2 resolution images. In Fig. 7c, we progressively increase the resolution of Agent-Swin-S, Swin-S, and Swin-B. It is evident that in high-resolution scenarios, our model consistently delivers notably superior outcomes.

5.6 Ablation Study

In this section, we ablate the key components in our agent attention module to verify the effectiveness of these designs. We report the results on ImageNet-1K classification based on Agent-Swin-T.

Ablation on key designs. We substitute Softmax attention in Swin-T with vanilla linear attention, followed by a gradual introduction of agent attention, agent bias, and DWC to create Agent-Swin-T. The results are depicted in Tab. 6. Three key findings emerge: (1) Agent attention boosts accuracy by 1.2, proving its effectiveness. (2) Agent bias serves as an effective position embedding for agent attention, similar to RPE in Swin. (3) DWC is a crucial complement to unlock the capabilities of agent attention. When applying DWC to Swin-T, a modest gain of 0.3 is observed. In contrast, with DWC preserving feature diversity, agent attention delivers a much better result (+1.5).

Ablation on number of agent tokens. The model's computation complexity can be modulated by varying the number of agent tokens. As shown in Tab. 7, shallower layers of the model have simple semantics, and judiciously decreasing

Table 7: Ablation on the number of agent tokens.

#Num of Agent Tokens				FLOPS	#Param	Acc.	Diff.
Stage1	Stage2	Stage3	Stage4				
49	49	49	49	4.7G	29M	82.6	-0.0
9	16	49	49	4.5G	29M	**82.6**	**Ours**
9	16	25	49	4.5G	29M	82.2	-0.4
4	9	49	49	4.5G	29M	82.4	-0.2
Swin-T				4.5G	29M	81.3	-1.3

Table 8: Comparison of different linear attention designs on DeiT-Tiny and Swin-Tiny.

DeiT-T Setting			
Linear Attention	FLOPs	#Param	Acc.
Hydra Attn [2]	1.1G	5.7M	68.3
Efficient Attn [33]	1.1G	5.7M	70.2
Linear Angular Attn [47]	1.1G	5.7M	70.8
Focused Linear Attn [11]	1.1G	6.1M	74.1
Ours	1.2G	6.0M	**74.9**

Swin-T Setting			
Linear Attention	FLOPs	#Param	Acc.
Hydra Attn [2]	4.5G	29M	80.7
Efficient Attn [33]	4.5G	29M	81.0
Linear Angular Attn [47]	4.5G	29M	79.4
Focused Linear Attn [11]	4.5G	29M	82.1
Ours	4.5G	29M	**82.6**

the number of agent tokens in these layers does not adversely affect performance. In contrast, deeper layers have rich semantics, and reducing agent tokens in these layers leads to performance degradation. Hence, our design principle is using fewer agent tokens in the model's shallow layers to reduce computation complexity and more agent tokens in the deep layers to better represent rich semantics. This aligns with the stripe width design principle in CSwin [9].

Comparison with Other Linear Attention. We conduct a comparison of our agent attention with other linear attention methods using DeiT-T and Swin-T. As depicted in Tab. 8, substituting the Softmax attention employed by DeiT-T and Swin-T with various linear attention methods usually results in notable performance degradation. Remarkably, our models outperform all other methods as well as the Softmax baseline.

6 Conclusion

This paper presents a new attention paradigm dubbed *Agent Attention*, which is applicable across a variety of vision Transformer models. As an elegant integration of Softmax and linear attention, agent attention enjoys both high expressive power and low computation complexity. Extensive experiments on image classification, semantic segmentation, and object detection unequivocally confirm the effectiveness of our approach, particularly in high-resolution scenarios. When integrated with Stable Diffusion, agent attention accelerates image generation and substantially enhances image quality without any extra training. Due to its linear complexity with respect to the number of tokens and its strong representation power, agent attention may pave the way for challenging tasks with super long token sequences, such as video modelling and multi-modal foundation models.

Acknowledgement

This work is supported in part by the National Key R&D Program of China under Grant 2021ZD0140407, the National Natural Science Foundation of China under Grants 42327901 and 62321005.

References

1. Bolya, D., Fu, C.Y., Dai, X., Zhang, P., Feichtenhofer, C., Hoffman, J.: Token merging: Your ViT but faster. In: ICLR (2023)
2. Bolya, D., Fu, C.Y., Dai, X., Zhang, P., Hoffman, J.: Hydra attention: Efficient attention with many heads. In: ECCVW (2022)
3. Bolya, D., Hoffman, J.: Token merging for fast stable diffusion. In: CVPRW (2023)
4. Cai, Z., Vasconcelos, N.: Cascade r-cnn: Delving into high quality object detection. In: CVPR (2018)
5. Carion, N., Massa, F., Synnaeve, G., Usunier, N., Kirillov, A., Zagoruyko, S.: End-to-end object detection with transformers. In: ECCV (2020)
6. Cheng, B., Schwing, A., Kirillov, A.: Per-pixel classification is not all you need for semantic segmentation. In: NeurIPS (2021)
7. Choromanski, K., Likhosherstov, V., Dohan, D., Song, X., Gane, A., Sarlos, T., Hawkins, P., Davis, J., Mohiuddin, A., Kaiser, L., et al.: Rethinking attention with performers. In: ICLR (2021)
8. Deng, J., Dong, W., Socher, R., Li, L.J., Li, K., Fei-Fei, L.: Imagenet: A large-scale hierarchical image database. In: CVPR (2009)
9. Dong, X., Bao, J., Chen, D., Zhang, W., Yu, N., Yuan, L., Chen, D., Guo, B.: Cswin transformer: A general vision transformer backbone with cross-shaped windows. In: CVPR (2022)
10. Dosovitskiy, A., Beyer, L., Kolesnikov, A., Weissenborn, D., Zhai, X., Unterthiner, T., Dehghani, M., Minderer, M., Heigold, G., Gelly, S., et al.: An image is worth 16x16 words: Transformers for image recognition at scale. In: ICLR (2021)
11. Han, D., Pan, X., Han, Y., Song, S., Huang, G.: Flatten transformer: Vision transformer using focused linear attention. In: ICCV (2023)
12. Han, Y., Han, D., Liu, Z., Wang, Y., Pan, X., Pu, Y., Deng, C., Feng, J., Song, S., Huang, G.: Dynamic perceiver for efficient visual recognition. In: ICCV (2023)
13. Han, Y., Huang, G., Song, S., Yang, L., Wang, H., Wang, Y.: Dynamic neural networks: A survey. TPAMI (2021)
14. Han, Y., Liu, Z., Yuan, Z., Pu, Y., Wang, C., Song, S., Huang, G.: Latency-aware unified dynamic networks for efficient image recognition. TPAMI (2024)
15. Han, Y., Pu, Y., Lai, Z., Wang, C., Song, S., Cao, J., Huang, W., Deng, C., Huang, G.: Learning to weight samples for dynamic early-exiting networks. In: ECCV (2022)
16. Hassani, A., Walton, S., Li, J., Li, S., Shi, H.: Neighborhood attention transformer. In: CVPR (2023)
17. He, K., Gkioxari, G., Dollár, P., Girshick, R.: Mask r-cnn. In: ICCV (2017)
18. Katharopoulos, A., Vyas, A., Pappas, N., Fleuret, F.: Transformers are rnns: Fast autoregressive transformers with linear attention. In: ICML (2020)
19. Kirillov, A., Girshick, R., He, K., Dollár, P.: Panoptic feature pyramid networks. In: CVPR (2019)

20. Li, Y., Wu, C.Y., Fan, H., Mangalam, K., Xiong, B., Malik, J., Feichtenhofer, C.: Mvitv2: Improved multiscale vision transformers for classification and detection. In: CVPR (2022)
21. Li, Y., Fan, Y., Xiang, X., Demandolx, D., Ranjan, R., Timofte, R., Van Gool, L.: Efficient and explicit modelling of image hierarchies for image restoration. In: CVPR (2023)
22. Lin, T.Y., Goyal, P., Girshick, R., He, K., Dollár, P.: Focal loss for dense object detection. In: ICCV (2017)
23. Lin, T.Y., Maire, M., Belongie, S., Hays, J., Perona, P., Ramanan, D., Dollár, P., Zitnick, C.L.: Microsoft coco: Common objects in context. In: ECCV (2014)
24. Liu, Z., Lin, Y., Cao, Y., Hu, H., Wei, Y., Zhang, Z., Lin, S., Guo, B.: Swin transformer: Hierarchical vision transformer using shifted windows. In: ICCV (2021)
25. Liu, Z., Mao, H., Wu, C.Y., Feichtenhofer, C., Darrell, T., Xie, S.: A convnet for the 2020s. In: CVPR (2022)
26. Lu, J., Yao, J., Zhang, J., Zhu, X., Xu, H., Gao, W., Xu, C., Xiang, T., Zhang, L.: Soft: Softmax-free transformer with linear complexity. In: NeurIPS (2021)
27. Pan, X., Ye, T., Han, D., Song, S., Huang, G.: Contrastive language-image pre-training with knowledge graphs. In: NeurIPS (2022)
28. Radford, A., Kim, J.W., Hallacy, C., Ramesh, A., Goh, G., Agarwal, S., Sastry, G., Askell, A., Mishkin, P., Clark, J., Krueger, G., Sutskever, I.: Learning transferable visual models from natural language supervision. In: ICML (2021)
29. Radosavovic, I., Kosaraju, R.P., Girshick, R., He, K., Dollár, P.: Designing network design spaces. In: CVPR (2020)
30. Rombach, R., Blattmann, A., Lorenz, D., Esser, P., Ommer, B.: High-resolution image synthesis with latent diffusion models. In: CVPR (2022)
31. Ruiz, N., Li, Y., Jampani, V., Pritch, Y., Rubinstein, M., Aberman, K.: Dreambooth: Fine tuning text-to-image diffusion models for subject-driven generation. In: CVPR (2023)
32. Shaw, P., Uszkoreit, J., Vaswani, A.: Self-attention with relative position representations. In: ACL (2018)
33. Shen, Z., Zhang, M., Zhao, H., Yi, S., Li, H.: Efficient attention: Attention with linear complexities. In: WACV (2021)
34. Tan, M., Le, Q.: Efficientnet: Rethinking model scaling for convolutional neural networks. In: ICML (2019)
35. Touvron, H., Cord, M., Douze, M., Massa, F., Sablayrolles, A., Jégou, H.: Training data-efficient image transformers & distillation through attention. In: ICML (2021)
36. Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A.N., Kaiser, Ł., Polosukhin, I.: Attention is all you need. In: NeurIPS (2017)
37. Wang, W., Xie, E., Li, X., Fan, D.P., Song, K., Liang, D., Lu, T., Luo, P., Shao, L.: Pyramid vision transformer: A versatile backbone for dense prediction without convolutions. In: ICCV (2021)
38. Wang, W., Xie, E., Li, X., Fan, D.P., Song, K., Liang, D., Lu, T., Luo, P., Shao, L.: Pvt v2: Improved baselines with pyramid vision transformer. Computational Visual Media (2022)
39. Wu, H., Xiao, B., Codella, N., Liu, M., Dai, X., Yuan, L., Zhang, L.: Cvt: Introducing convolutions to vision transformers. In: ICCV (2021)
40. Xia, Z., Han, D., Han, Y., Pan, X., Song, S., Huang, G.: Gsva: Generalized segmentation via multimodal large language models. In: CVPR (2024)
41. Xia, Z., Pan, X., Song, S., Li, L.E., Huang, G.: Vision transformer with deformable attention. In: CVPR (2022)

42. Xiao, T., Liu, Y., Zhou, B., Jiang, Y., Sun, J.: Unified perceptual parsing for scene understanding. In: ECCV (2018)
43. Xie, E., Wang, W., Yu, Z., Anandkumar, A., Alvarez, J.M., Luo, P.: Segformer: Simple and efficient design for semantic segmentation with transformers. In: NeurIPS (2021)
44. Xiong, Y., Zeng, Z., Chakraborty, R., Tan, M., Fung, G., Li, Y., Singh, V.: Nyströmformer: A nyström-based algorithm for approximating self-attention. In: AAAI (2021)
45. Yan, H., Li, Z., Li, W., Wang, C., Wu, M., Zhang, C.: Contnet: Why not use convolution and transformer at the same time? arXiv preprint arXiv:2104.13497 (2021)
46. Yang, J., Li, C., Zhang, P., Dai, X., Xiao, B., Yuan, L., Gao, J.: Focal self-attention for local-global interactions in vision transformers. In: NeurIPS (2021)
47. You, H., Xiong, Y., Dai, X., Wu, B., Zhang, P., Fan, H., Vajda, P., Lin, Y.C.: Castling-vit: Compressing self-attention via switching towards linear-angular attention at vision transformer inference. In: CVPR (2023)
48. Yuan, L., Chen, Y., Wang, T., Yu, W., Shi, Y., Jiang, Z.H., Tay, F.E., Feng, J., Yan, S.: Tokens-to-token vit: Training vision transformers from scratch on imagenet. In: ICCV (2021)
49. Zhou, B., Zhao, H., Puig, X., Xiao, T., Fidler, S., Barriuso, A., Torralba, A.: Semantic understanding of scenes through the ade20k dataset. IJCV (2019)
50. Zhu, L., Wang, X., Ke, Z., Zhang, W., Lau, R.W.: Biformer: Vision transformer with bi-level routing attention. In: CVPR (2023)