Vision Transformer with Sparse Scan Prior

Qihang Fan [1,2]**, Huaibo Huang**[1]***Mingrui Chen**[1,2]**, Ran He**[1,2]
[1]MAIS & CRIPAC, Institute of Automation, Chinese Academy of Sciences, Beijing, China
[2]School of Artificial Intelligence, University of Chinese Academy of Sciences, Beijing, China
fanqihang.159@gmail.com, huaibo.huang@cripac.ia.ac.cn,
charmier2003@gmail.com, rhe@nlpr.ia.ac.cn

Abstract

In recent years, Transformers have achieved remarkable progress in computer vision tasks. However, their global modeling often comes with substantial computational overhead, in stark contrast to the human eye's efficient information processing. Inspired by the human eye's sparse scanning mechanism, we propose a **S**parse **S**can **S**elf-**A**ttention mechanism (S^3A). This mechanism predefines a series of Anchors of Interest for each token and employs local attention to efficiently model the spatial information around these anchors, avoiding redundant global modeling and excessive focus on local information. This approach mirrors the human eye's functionality and significantly reduces the computational load of vision models. Building on S^3A, we introduce the **S**parse **S**can **Vi**sion **T**ransformer (SSViT). Extensive experiments demonstrate the outstanding performance of SSViT across a variety of tasks. Specifically, on ImageNet classification, without additional supervision or training data, SSViT achieves top-1 accuracies of **84.4%/85.7%** with **4.4G/18.2G** FLOPs. SSViT also excels in downstream tasks such as object detection, instance segmentation, and semantic segmentation. Its robustness is further validated across diverse datasets. Code will be available at `https://github.com/qhfan/SSViT`.

1 Introduction

Since its inception, the Vision Transformer (ViT) [12] has attracted considerable attention from the research community, primarily owing to its exceptional capability in modeling long-range dependencies. However, the self-attention mechanism [61], as the core of ViT, imposes significant computational overhead, thus constraining its broader applicability. Several strategies have been proposed to alleviate this limitation of self-attention. For instance, methods such as Swin-Transformer [40, 11] group tokens for attention, reducing computational costs and enabling the model to focus more on local information. Techniques like PVT [63, 64, 18, 16, 29] down-sample tokens to shrink the size of the QK matrix, thus lowering computational demands while retaining global information. Meanwhile, approaches such as UniFormer [35, 47] forgo attention operations in the early stages of visual modeling, opting instead for lightweight convolution. Furthermore, some models [50] enhance computational efficiency by pruning redundant tokens.

Despite these advancements, the majority of methods primarily focus on reducing the token count in self-attention operations to boost ViT efficiency, often neglecting the manner in which human eyes process visual information. The human visual system operates in a notably less intricate yet highly efficient manner compared to ViT models. Unlike the fine-grained local spatial information modeling in models like Swin [40], NAT [20], LVT [69], or the indistinct global information modeling seen in models like PVT [63], PVTv2 [64], CMT [18], human vision employs a sparse scanning

---

*Huaibo Huang is the corresponding author.

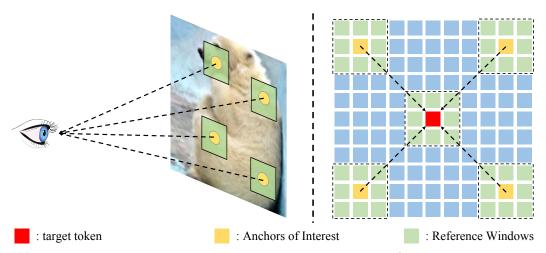: target token   : Anchors of Interest   : Reference Windows

Figure 1: Illustration of sparse scan mechanism in human eye and our S^3A. **Left:** The human eye exhibits a sparse scan mechanism when observing visual information, focusing only on a few anchors of interest and the local information surrounding these anchors. It doesn't soly model global/local information. **Right:** Our proposed S^3A mimics the sparse scan mechanism illustrated in the left figure, focusing on modeling the local information surrounding the Anchors of Interest.

mechanism, as substantiated by numerous biological studies [44, 55, 54]. As illustrated in Fig 1, our eyes swiftly move between points of interest, delving into detailed information processing solely at these anchor points [43, 14, 48]. This selective attention mechanism enables the brain to efficiently process essential visual information, rather than being solely focused on local details or vague global information. Given that the human retina's fovea has a fixed size, each shift in the eye's focal point results in a fixed-size receptive field being sensed [31, 45].

As depicted in Fig 1, we introduce a novel Self-Attention mechanism, termed **S**parse **S**can **S**elf-**A**ttention (S^3A), inspired by the sparse scanning mechanism of the human eye. For each target token, we design a set of uniformly distributed **A**nchors **o**f **I**nterest (AoI). We apply local attention to these AoIs, processing the surrounding visual information and utilizing this local data to update the AoI tokens. The size of each local window remains constant, reflecting the fixed foveal size in human vision. We subsequently aggregate the information from all AoIs to update the target token. The S^3A modeling approach harmonizes fine-grained local modeling and sparse modeling of interest anchors, closely mirroring the functioning of the human eye. S^3A's methodology surpasses previous Self-Attention mechanisms, offering a more human-like, efficient, and effective model.
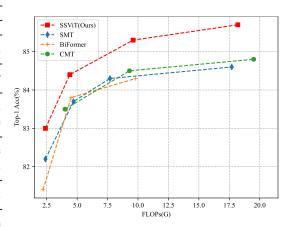

Figure 2: Top-1 accuracy v.s. FLOPs on ImageNet-1K of recent SOTA models. Our SSViT outperforms all the counterparts in all settings.

Building on S^3A, we develop the Sparse Scan Vision Transformer (SSViT). SSViT effectively replicates the human eye's visual information processing and demonstrates remarkable effectiveness across a spectrum of visual tasks. As demonstrated in Fig 2, SSViT outperforms previous state-of-the-art models in image classification accuracy, achieving 83.0% top-1 accuracy with a mere 15M parameters and 2.4G FLOPs, without the need for additional training data or supervision. This performance advantage is sustained even when the model scales up, with our SSViT-L achieving 85.7% top-1 accuracy with only 100M parameters. Beyond classification tasks, SSViT also excels in downstream tasks such as object detection, instance segmentation, and semantic segmentation. The robustness of SSViT is further corroborated by its superior performance across a variety of datasets.
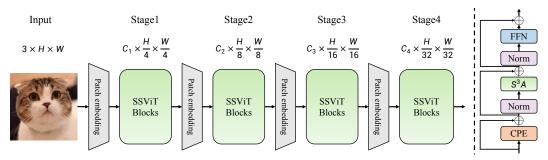
Figure 3: Illustration of the SSViT. SSViT consists of multiple SSViT blocks. A single SSViT block is composed of CPE, S^3A, and FFN.

2 Related works

Vision Transformers. The Vision Transformer (ViT)[12] has attracted significant attention since its inception, largely due to its superior performance. Numerous studies[40, 11, 16, 15] have explored ways to optimize ViT by refining its central operator, Self-Attention, with the dual objective of reducing its quadratic computational complexity and enhancing its performance. A range of methods [11, 40, 76] have been proposed to reduce the computational burden of Self-Attention. These techniques limit the region that each token can attend to by grouping tokens. The Swin-Transformer [40], for instance, divides all tokens into separate windows and performs Self-Attention operations within these windows. The BiFormer [76], in contrast, dynamically determines the windows that each token can attend to. Additionally, some methods [63, 64, 18, 29, 22, 47, 4] reduce the number of tokens involved in Self-Attention operations through token downsampling. The PVT [63, 64] employs average pooling for direct downsampling, thus decreasing the number of tokens. The CMT [18] and PVTv2 [64] supplement token downsampling with convolution to enhance the model's ability to learn local features. The STViT [29] effectively captures global dependencies by sampling super tokens, applying self-attention to them, and subsequently mapping them back to the original token space. Certain approaches [39, 47, 35] choose to forego the computationally demanding Self-Attention in the early layers of the model, instead employing more efficient convolutions to learn local features. Self-Attention is then deployed in the deeper layers of the model to learn global features. While these aforementioned methods exhibit promising results and reduce computational complexity, it is important to note that their operational mechanisms significantly deviate from the functioning of the human eye.

Sparse Scan in Human Eye. Sparse scanning, a critical mechanism in human vision, facilitates the efficient processing of visual stimuli in the face of sensory limitations. Neuroimaging investigations have pinpointed key neural structures, such as the superior colliculus, that govern sparse scanning behaviors [58]. Furthermore, microsaccades, minute ocular movements considered a variant of sparse scanning, are instrumental in preserving visual stability and guiding attention [48]. Impairments in sparse scanning are linked to cognitive deficits observed in disorders like ADHD and schizophrenia [44]. The dynamism of sparse scanning in relation to task requirements and attentional preferences has also been a focus of research. Evidence suggests that sparse scanning exhibits adaptability to task-specific demands, flexibly distributing visual resources to enhance processing efficiency [14]. This adaptive characteristic underscores the complex interplay between bottom-up sensory inputs and top-down cognitive influences in visual processing. In essence, sparse scanning is a fundamental mechanism that impacts not only basic perceptual functions but also more complex cognitive processes [55, 54, 43].

3 Method

3.1 Overall Architecture.

The overall architecture of Sparse Scan Vision Transformer (SSViT) is shown in Fig. 3. To process the input image $x \in \mathbb{R}^{3 \times H \times W}$, we feed it into a patch embedding composed of convolutions, obtaining tokens with a shape of $C_1 \times \frac{H}{4} \times \frac{W}{4}$. Following the previous hierarchical designs [40, 15, 16], we
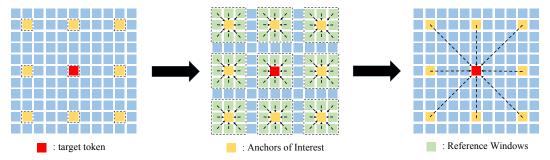
3

Figure 4: Illustration of Sparse Scan Self-Attention.

divide SSViT into four stages. The multi-resolution representations brought by hierarchical structures can be utilized for downstream tasks such as object detection and semantic segmentation.

An SSViT block consists of three key components: Conditional Positional Encoding (CPE) [6], Sparse Scan Self-Attention (S^3A), and Feed-Forward Network (FFN) [61, 12]. The complete SSViT block can be defined as Eq. 1:

$$\begin{aligned} X &= \text{CPE}(X_{in}) + X_{in}, \\ Y &= \text{S}^3\text{A}(\text{LN}(X)) + X, \\ Z &= \text{FFN}(\text{LN}(Y)) + Y. \end{aligned} \tag{1}$$

For each block, the input tensor $X_{in} \in \mathbb{R}^{C \times H \times W}$ is fed into the CPE to introduce the positional information for each token. After CPE, S^3A is employed to scan sparse regions of interest for each token. The final FFN is utilized to integrate channel-wise information of tokens.

3.2 Sparse Scan Self-Attention

The Sparse Scan Self-Attention (S^3A) is inspired by the sparse scan mechanism of the human eye in processing visual information. It can be decomposed into three sub-processes. **Firstly**, the selection of Anchors of Interest (AoI) for each token. **Secondly**, the extraction of background local information within the reference windows (RWins) determined by the AoIs. **Finally**, the interaction among AoIs. The whole process can be seen in Fig. 4.

Anchors of Interests. Ideally, each token should select its own suitable AoIs based on its characteristics. However, this approach would grant the model excessive freedom, resulting in a cumbersome implementation process and low efficiency. Therefore, we have abandoned this practice and instead opt to artificially define the AoIs for each token. Specifically, assuming the stride of the AoIs is (S_h, S_w), and the number of AoIs chosen for each token is (N_h, N_w). For a token located at position (i, j), its selected AoIs can be represented as Eq. 2:

$$\begin{aligned} \mathbb{A}_{i,j} =& \{X_{m,n} | m = i + k_h \times S_h, n = j + k_w \times S_w\}, \\ & k_h \in (-\lfloor N_h/2 \rfloor, \lfloor N_h/2 \rfloor), k_h \in \mathbb{Z}, \\ & k_w \in (-\lfloor N_w/2 \rfloor, \lfloor N_w/2 \rfloor), k_w \in \mathbb{Z}. \end{aligned} \tag{2}$$

where X is the input feature map, and $\mathbb{A}_{i,j}$ is the set of AoIs for X_{ij}. For the sake of simplicity, in Eq. 2, we do not consider the situation of boundary points. In practice, when dealing with boundary points, there is a certain offset in the range of k_h and k_w to ensure that all AoIs remain within the boundaries of the feature map.

Background Local Information. As shown in Fig. 1, when the human eye observes certain anchors, it also processes the surrounding background information. Based on this consideration, for each anchor point in the AoI from the previous step, we select its corresponding reference window (RWin), which is the background of the anchor point. Consistent with the previous definition, we assume the size of a single RWin is (W_h, W_w). Since the size of the fovea on the human eye's retina remains constant, the receptive field perceived by the human eye is the same for each AoI. Translated into model design, this means that each AoI has a RWin of the same size. For an AoI located at position

4

(m, n), its RWin is defined as Eq. 3:

$$\begin{aligned}
\mathbb{R}_{m,n} &= \{X_{p,q}|p = m + r_h, q = n + r_w\}, \\
r_h &\in [-\lfloor W_h/2 \rfloor, \lfloor W_h/2 \rfloor], r_h \in \mathbb{Z}, \\
r_w &\in [-\lfloor W_w/2 \rfloor, \lfloor W_w/2 \rfloor], r_w \in \mathbb{Z}, \\
X_{m,n} &\in \mathbb{A}_{i,j}
\end{aligned} \tag{3}$$

Where $\mathbb{R}_{m,n}$ is the set of tokens in RWin of $X_{m,n}$. Similar to the definition of AoI, in Eq. 3, we omit the situation of boundary points. For each AoI, within its determined RWin, we utilize Self-Attention to update the AoI. For an AoI with the position of (m, n), this process can be represented as Eq. 4:

$$\begin{aligned}
X_{m,n}^* &= \text{Attn}(W_q X_{m,n}, W_k \mathbb{R}_{m,n}, W_v \mathbb{R}_{m,n}), \\
\mathbb{A}_{i,j}^* &= \{X_{m,n}^* | X_{m,n} \in \mathbb{A}_{i,j}\},
\end{aligned} \tag{4}$$

where $\text{Attn}(q, k, v)$ denotes the standard Self-Attention operation. W_q, W_k, W_v are learnable matrices. $X_{m,n}^*$ is the updated $X_{m,n}$. $\mathbb{A}_{i,j}^*$ is the set of updated AoIs for $X_{i,j}$.

Interaction among Anchors. In practice, the human eye does not process each anchor independently. Instead, it performs interactive modeling of the information observed in each anchor, thereby inferring deep semantic information from the image. We also model this process by Self-Attention. For the target token $X_{i,j}$, after all its AoIs have been updated, we utilize $\mathbb{A}_{i,j}$ and $\mathbb{A}_{i,j}^*$ to update $X_{i,j}$, as shown specifically in Eq. 5:

$$X_{i,j} = \text{Attn}(W_q X_{i,j}, W_k \mathbb{A}_{i,j}, \mathbb{A}_{i,j}^*). \tag{5}$$

The above three-step process constitutes the complete S^3A. After the completion of S^3A, to further enhance the model's ability to capture local information, we employ a local context enhancement module to model local information:

$$X = S^3A(X) + \text{LCE}(W_v X), \tag{6}$$

where LCE is a simple depth-wise convolution. It is worth noting that although S^3A performs two rounds of Self-Attention calculations, in practice, the projection of q, k, and v is completed in a single operation. During the two self-attention computations (Eq. 4 and Eq. 5), q and k are reused, thus no additional computational or parameter overhead is introduced.

4 Experiments

We conducted experiments on a wide range of vision tasks, including image classification on ImageNet-1k [9], object detection and instance segmentation on COCO [38], and semantic segmentation on ADE20K [74]. We also evaluate the SSViT's robustness on ImageNet-v2 [52], ImageNet-A [26], ImageNet-R [25]. All models can be trained with 8 A100 80G GPUs.

4.1 ImageNet Classification

Settings. We train our models from scratch on ImageNet-1k [9]. For a fair comparison, we adopt the same training strategy as in [40], with classification loss serving as the sole supervision. The maximum rates for increasing stochastic depth [28] are set to 0.1, 0.15, 0.4, and 0.5 for SSViT-T, SSViT-S, SSViT-B, and SSViT-L, respectively.

Comparison with SOTA. We benchmark our SSViT against numerous state-of-the-art models, with results presented in Tab.1. SSViT consistently outperforms preceding models across all scales. Notably, SSViT-T attains a Top1-accuracy of **83.0%** with a mere 15M parameters and **2.4G** FLOPs, exceeding the previous state-of-the-art (SMT[39]) by **0.8%**. For larger models, SSViT-L achieves a Top1-accuracy of **85.7%** with **100M** parameters and **18.2G** FLOPs.

Strict Comparison with General/Efficient Models. To guarantee a fair comparison, we select two baselines: the general-purpose backbone, Swin-Transformer [40], and the efficiency-oriented backbone, FasterViT [21]. We compare these with SSViT. In the comparison models (SS-Swin and

Table 1: Comparison with the state-of-the-art on ImageNet-1K classification.

Cost	Model	Parmas (M)	FLOPs (G)	Top1-acc (%)
tiny model ~ 2.5G	QuadTree-B-b1 [57]	14	2.3	80.0
	RegionViT-T [4]	14	2.4	80.4
	MPViT-XS [34]	11	2.9	80.9
	VAN-B1 [19]	14	2.5	81.1
	BiFormer-T [76]	13	2.2	81.4
	Conv2Former-N [27]	15	2.2	81.5
	CrossFormer-T [65]	28	2.9	81.5
	NAT-M [20]	20	2.7	81.8
	FAT-B2 [16]	14	2.0	81.9
	QnA-T [1]	16	2.5	82.0
	GC-ViT-XT [22]	20	2.6	82.0
	SMT-T [39]	12	2.4	82.2
	SSViT-T	15	2.4	**83.0**
small model ~ 4.5G	Swin-T [40]	29	4.5	81.3
	CrossViT-15 [3]	27	5.8	81.5
	RVT-S [42]	23	4.7	81.9
	ConvNeXt-T [41]	29	4.5	82.1
	Focal-T [70]	29	4.9	82.2
	MPViT-S [34]	23	4.7	83.0
	SG-Former-S [17]	23	4.8	83.2
	Ortho-S [30]	24	4.5	83.4
	InternImage-T [62]	30	5.0	83.5
	GC-ViT-T [22]	28	4.7	83.5
	CMT-S [18]	25	4.0	83.5
	FAT-B3 [16]	29	4.4	83.6
	SMT-S [39]	20	4.8	83.7
	BiFormer-S [76]	26	4.5	83.8
	SSViT-S	27	4.4	**84.4**

Cost	Model	Parmas (M)	FLOPs (G)	Top1-acc (%)
base model ~ 9.0G	ConvNeXt-S [41]	50	8.7	83.1
	CrossFormer-B [65]	52	9.2	83.4
	NAT-S [20]	51	7.8	83.7
	Quadtree-B-b4 [57]	64	11.5	84.0
	ScaleViT-B [71]	81	8.6	84.1
	MOAT-1 [68]	42	9.1	84.2
	InternImage-S [62]	50	8.0	84.2
	DaViT-S [10]	50	8.8	84.2
	BiFormer-B [76]	57	9.8	84.3
	MViTv2-B [36]	52	10.2	84.4
	CMT-B [18]	46	9.3	84.5
	iFormer-B [56]	48	9.4	84.6
	STViT-B [29]	52	9.9	84.8
	SSViT-B	57	9.6	**85.3**
large model ~ 18.0G	DeiT-B [59]	86	17.5	81.8
	LITv2 [47]	87	13.2	83.6
	CrossFormer-L [65]	92	16.1	84.0
	Ortho-L [30]	88	15.4	84.2
	CSwin-B [11]	78	15.0	84.2
	SMT-L [39]	81	17.7	84.6
	DaViT-B [10]	88	15.5	84.6
	SG-Former-B [17]	78	15.6	84.7
	iFormer-L [56]	87	14.0	84.8
	CMT-L [18]	75	19.5	84.8
	InterImage-B [62]	97	16.0	84.9
	MaxViT-B [60]	120	23.4	84.9
	GC-ViT-B [22]	90	14.8	85.0
	SSViT-L	100	18.2	**85.7**

SS-FasterViT), we merely substitute the attention mechanism in the original Swin-Transformer and FasterViT with S^3A, without introducing any other modifications (such as CPE, Conv Stem, etc.). As shown in Tab. 2, the simple replacement of the attention mechanism with S^3A yields significant advantages in both performance and efficiency. Specifically, SS-Swin achieves or even surpasses a **2.0%** improvement over Swin across all model sizes. Meanwhile, SS-FasterViT attains higher accuracy than FasterViT while utilizing fewer parameters.

Model	Parmas (M)	FLOPs (G)	Throughput (imgs/s)	Top1-acc (%)
Swin-T [40]	29	4.5	1723	81.3
SS-Swin-T	29	4.8	1282	**83.8**(+2.5)
Swin-S [40]	50	8.8	1062	83.0
SS-Swin-S	50	8.9	724	**84.7**(+1.7)
Swin-B [40]	88	15.4	798	83.3
SS-Swin-B	88	15.7	538	**85.1**(+1.8)

Model	Parmas (M)	FLOPs (G)	Throughput (imgs/s)	Top1-acc (%)
FasterViT-0 [21]	31	3.3	3551	82.1
SS-FasterViT-0	25	3.4	3021	**82.8**(+0.7)
FasterViT-1 [21]	53	5.3	2619	83.2
SS-FasterViT-1	41	5.4	2282	**83.7**(+0.5)
FasterViT-2 [21]	76	8.7	1988	84.2
SS-FasterViT-2	58	8.8	1783	**84.6**(+0.4)

Table 2: Strict comparison with the baslines on ImageNet-1K classfication. The speed of the models are measured on A100 GPU with the batch size of 64.

4.2 Object Detection and Instance Segmentation

Settings. We utilize MMDetection [5] to implement Mask-RCNN [24], Cascade Mask R-CNN [2], and RetinaNet [37] for evaluating our SSViT. For Mask R-CNN and Cascade Mask R-CNN, we adhere to the commonly used "3 × + MS" setting, and for Mask R-CNN and RetinaNet, we apply the "1×" setting. Following [40], during training, we resize the images such that the shorter side is 800 pixels while keeping the longer side within 1333 pixels. We employ the AdamW optimizer for model optimization.

Results. Tab. 3 and Tab. 4 present the performance of SSViT across different detection frameworks. The results highlight that SSViT consistently outperforms its counterparts in all comparisons. Under

Table 3 (Mask R-CNN 3×+MS)

Backbone	Params (M)	FLOPs (G)	AP^b	AP^b_{50}	AP^b_{75}	AP^m	AP^m_{50}	AP^m_{75}
Focal-T [70]	49	291	47.2	69.4	51.9	42.7	66.5	45.9
NAT-T [20]	48	258	47.8	69.0	52.6	42.6	66.0	45.9
GC-ViT-T [22]	48	291	47.9	70.1	52.8	43.2	67.0	46.7
MPViT-S [34]	43	268	48.4	70.5	52.6	43.9	67.6	47.5
SMT-S [39]	40	265	49.0	70.1	53.4	43.4	67.3	46.7
CSWin-T [11]	42	279	49.0	70.7	53.7	43.6	67.9	46.6
InternImage-T [62]	49	270	49.1	70.4	54.1	43.7	67.3	47.3
SSViT-S	46	266	**51.2**	**72.0**	**56.0**	**45.4**	**69.7**	**49.0**
ConvNeXt-S [41]	70	348	47.9	70.0	52.7	42.9	66.9	46.2
NAT-S [20]	70	330	48.4	69.8	53.2	43.2	66.9	46.4
Swin-S [40]	69	359	48.5	70.2	53.5	43.3	67.3	46.6
InternImage-S [62]	69	340	49.7	71.1	54.5	44.5	68.5	47.8
SMT-B [39]	52	328	49.8	71.0	54.4	44.0	68.0	47.3
CSWin-S [11]	54	342	50.0	71.3	54.7	44.5	68.4	47.7
SSViT-B	76	382	**52.6**	**73.2**	**57.7**	**46.4**	**70.9**	**50.3**

Table 3 (Cascade Mask R-CNN 3×+MS)

Backbone	Params (M)	FLOPs (G)	AP^b	AP^b_{50}	AP^b_{75}	AP^m	AP^m_{50}	AP^m_{75}
NAT-T [20]	85	737	51.4	70.0	55.9	44.5	67.6	47.9
GC-ViT-T [22]	85	770	51.6	70.4	56.1	44.6	67.8	48.3
SMT-S [39]	78	744	51.9	70.5	56.3	44.7	67.8	48.6
UniFormer-S [35]	79	747	52.1	71.1	56.6	45.2	68.3	48.9
Ortho-S [30]	81	755	52.3	71.3	56.8	45.3	68.6	49.2
HorNet-T [51]	80	728	52.4	71.6	56.8	45.6	69.1	49.6
CSWin-T [11]	80	757	52.5	71.5	57.1	45.3	68.8	48.9
SSViT-S	84	745	**53.8**	**72.4**	**58.1**	**46.6**	**70.1**	**50.4**
Swin-S [40]	107	838	51.9	70.7	56.3	45.0	68.2	48.8
NAT-S [20]	108	809	51.9	70.4	56.2	44.9	68.3	48.9
GC-ViT-S [22]	108	866	52.4	71.0	57.1	45.4	68.5	49.3
DAT-S [66]	107	857	52.7	71.7	57.2	45.5	69.1	49.3
CSWin-S [11]	92	820	53.7	72.2	58.4	46.4	69.6	50.6
UniFormer-S [35]	107	878	53.8	72.8	58.5	46.4	69.9	50.4
SSViT-B	114	861	**54.9**	**73.7**	**59.7**	**47.6**	**71.6**	**51.5**

Table 3: Comparison with other backbones using "3 × +MS" schedule on COCO.

the "3 × +MS" schedule, SSViT surpasses the recent SMT, achieving a **+2.2** box AP and **+2.0** mask AP improvement with the Mask R-CNN framework. For Cascade Mask R-CNN, SSViT still maintains a significant performance edge over SMT. Regarding the "1×" schedule, SSViT exhibits remarkable performance. Specifically, SSViT-S attains an improvement of **+2.2** box AP and **+1.5** mask AP over InternImage-T within the Mask-RCNN framework.

Table 4

Backbone	Params (M)	FLOPs (G)	AP^b	AP^b_{50}	AP^b_{75}	AP^m	AP^m_{50}	AP^m_{75}	Params (M)	FLOPs (G)	AP^b	AP^b_{50}	AP^b_{75}	AP^b_S	AP^b_M	AP^b_L
			Mask R-CNN 1×								RetinaNet 1×					
PVT-T [63]	33	240	39.8	62.2	43.0	37.4	59.3	39.9	23	221	39.4	59.8	42.0	25.5	42.0	52.1
PVTv2-B1 [64]	33	243	41.8	54.3	45.9	38.8	61.2	41.6	23	225	41.2	61.9	43.9	25.4	44.5	54.3
MPViT-XS [34]	30	231	44.2	66.7	48.4	40.4	63.4	43.4	20	211	43.8	65.0	47.1	28.1	47.6	56.5
SSViT-T	34	223	**47.3**	**69.1**	**51.7**	**42.6**	**66.2**	**45.8**	24	205	**45.6**	**66.5**	**49.3**	**28.6**	**50.1**	**60.5**
CMT-S [18]	45	249	44.6	66.8	48.9	40.7	63.9	43.4	44	231	44.3	65.5	47.5	27.1	48.3	59.1
ScalableViT-S [71]	46	256	45.8	67.6	50.0	41.7	64.7	44.8	36	238	45.2	66.5	48.4	29.2	49.1	60.3
InternImage-T [62]	49	270	47.2	69.0	52.1	42.5	66.1	45.8	–	–	–	–	–	–	–	–
STViT-S [29]	44	252	47.6	70.0	52.3	43.1	66.8	46.5	–	–	–	–	–	–	–	–
SMT-S [39]	40	265	47.8	69.5	52.1	43.0	66.6	46.1	–	–	–	–	–	–	–	–
BiFormer-S [76]	–	–	47.8	69.8	52.3	43.2	66.8	46.5	–	–	45.9	66.9	49.4	30.2	49.6	61.7
SSViT-S	46	266	**49.4**	**70.8**	**54.1**	**44.0**	**67.7**	**47.3**	36	248	**47.5**	**68.6**	**50.8**	**30.1**	**52.2**	**63.3**
Swin-S [40]	69	359	45.7	67.9	50.4	41.1	64.9	44.2	60	339	44.5	66.1	47.4	29.8	48.5	59.1
ScalableViT-B [71]	95	349	46.8	68.7	51.5	42.5	65.8	45.9	85	330	45.8	67.3	49.2	29.9	49.5	61.0
InternImage-S [62]	69	340	47.8	69.8	52.8	43.3	67.1	46.7	–	–	–	–	–	–	–	–
CSWin-S [11]	54	342	47.9	70.1	52.6	43.2	67.1	46.2	–	–	–	–	–	–	–	–
BiFormer-B [76]	–	–	48.6	70.5	53.8	43.7	67.6	47.1	–	–	47.1	68.5	50.4	31.3	50.8	62.6
SSViT-B	76	382	**51.0**	**72.5**	**55.8**	**45.4**	**69.7**	**48.9**	66	363	**49.0**	**70.2**	**52.9**	**32.4**	**53.4**	**64.8**
Swin-B [40]	107	496	46.9	69.2	51.6	42.3	66.0	45.5	98	477	45.0	66.4	48.3	28.4	49.1	60.6
Focal-B [70]	110	533	47.8	70.2	52.5	43.2	67.3	46.5	101	514	46.3	68.0	49.8	31.7	50.4	60.8
MPViT-B [34]	95	503	48.2	70.0	52.9	43.5	67.1	46.8	85	482	47.0	68.4	50.8	29.4	51.3	61.5
CSwin-B [11]	97	526	48.7	70.4	53.9	43.9	67.8	47.3	–	–	–	–	–	–	–	–
InternImage-B [62]	115	501	48.8	70.9	54.0	44.0	67.8	47.4	–	–	–	–	–	–	–	–
SSViT-L	119	572	**51.6**	**72.9**	**56.6**	**46.0**	**70.1**	**49.8**	109	553	**50.0**	**71.4**	**53.8**	**33.2**	**54.6**	**65.0**

Table 4: Comparison to other backbones using "1×" schedule on COCO.

4.3 Semantic Segmentation

Settings. We utilize Semantic FPN [33] and UperNet [67] to assess SSViT's performance, implementing these frameworks via MMSegmentation [7]. We mirror PVT's [63] training settings for Semantic FPN, training the model for 80k iterations. All models use an input resolution of 512×512, and during testing, the image's shorter side is resized to 512 pixels. UperNet is trained for 160K iterations, following Swin's [40] settings. We employ the AdamW optimizer with a weight decay of 0.01, including a 1500 iteration warm-up.

Results. The results of semantic segmentation are detailed in Tab. 5. All FLOPs are evaluated using an input resolution of 512×2048, with the exception of the SSViT-T group, which employs

Semantic FPN					UperNet			
Backbone	Params(M)	FLOPs(G)	mIoU(%)		Backbone	Params(M)	FLOPs(G)	mIoU(%)
PVTv2-B1 [64]	18	34	42.5		DAT-T [66]	60	957	45.5
FAT-B2 [16]	17	32	45.4		NAT-T [20]	58	934	47.1
EdgeViT-S [46]	17	32	45.9		InternImage-T [62]	59	944	47.9
SSViT-T	18	35	**46.8**		MPViT-S [34]	52	943	48.3
					SMT-S [39]	50	935	49.2
DAT-T [66]	32	198	42.6		SSViT-S	56	941	**50.1**
CSWin-T [11]	26	202	48.2		DAT-S [66]	81	1079	48.3
Shuted-S [53]	26	183	48.2		SMT-B [39]	62	1004	49.6
FAT-B3 [16]	33	179	48.9		InterImage-S [62]	80	1017	50.2
SSViT-S	30	184	**49.6**		MPViT-B [34]	105	1186	50.3
DAT-S [66]	53	320	46.1		CSWin-S [11]	65	1027	50.4
RegionViT-B+ [4]	77	459	47.5		SSViT-B	86	1060	**52.2**
UniFormer-B [35]	54	350	47.7		Swin-B [40]	121	1188	48.1
CSWin-S [11]	39	271	49.2		GC ViT-B [22]	125	1348	49.2
SSViT-B	60	303	**51.0**		DAT-B [66]	121	1212	49.4
DAT-B [66]	92	481	47.0		InternImage-B [62]	128	1185	50.8
CrossFormer-L [65]	95	497	48.7		CSWin-B [11]	109	1222	51.1
CSWin-B [11]	81	464	49.9		SSViT-L	130	1256	**53.3**
SSViT-L	103	497	**51.5**					

Table 5: Comparison with the state-of-the-art on ADE20K.

a 512×512 resolution. Across all settings, SSViT delivers superior performance. Notably, within the Semantic FPN framework, our SSViT-S exceeds FAT-B3 by a significant **+0.7** mIoU margin. SSViT-L further outperforms CSWin-L by a remarkable **+1.6** mIoU. Within the UperNet framework, SSViT-S outstrips the recent SMT-S by **+0.9** mIoU. Both SSViT-B and SSViT-L also surpass their respective counterparts.

4.4 Robustness Evaluation

Settings. In line with previous studies [42, 75, 68], we assess SSViT's robustness using ImageNet-V2 [52], ImageNet-A [26], and ImageNet-R [25]. The models used for this evaluation are pretrained on ImageNet-1k [9].

Results. Tab. 6 presents the robustness evaluation outcomes. On ImageNet-V2 (IN-V2), SSViT outperforms all competitors. For instance, SSViT-B exceeds BiFormer-B by **+1.7**, maintaining similar parameters and FLOPs. SSViT's advantages are further amplified on ImageNet-A (IN-A) and ImageNet-R (IN-R). Specifically, SSViT-L, pretrained solely on ImageNet-1k, achieves accuracies of **55.0** on IN-A and **59.2** on IN-R, markedly outpacing FAN-Hybrid-L (IN-A: **+13.2**, IN-R: **+6.0**). This underscores SSViT's robustness.

Backbone	Params(M)	FLOPs(G)	IN-V2(%)		Backbone	Params(M)	FLOPs(G)	IN-A(%)	IN-R(%)
tiny-MOAT-2 [68]	10	2.3	70.1		SSViT-T	15	2.4	**32.6**	**45.6**
BiFormer-T [76]	13	2.2	70.7		Swin-T [40]	29	4.5	21.6	41.3
SMT-T [39]	12	2.4	71.0		ConvNeXt-T [41]	29	4.5	24.2	47.2
SSViT-T	15	2.4	**72.3**		RVT-S* [42]	23	4.7	25.7	47.7
XCiT-S12 [13]	26	4.8	72.5		FAN-S-Hybrid [75]	26	6.7	33.9	50.7
SMT-S [39]	21	4.7	73.3		SSViT-S	27	4.4	**41.6**	**51.0**
BiFormer-S [76]	26	4.5	73.6		ConvNeXt-S [41]	50	8.7	31.2	49.5
SSViT-S	27	4.4	**74.1**		LV-ViT-M [32]	56	16.0	35.2	47.2
XCiT-S24 [13]	48	9.1	73.3		FAN-B-Hybrid [75]	50	11.3	39.6	52.9
BiFormer-B [76]	57	9.8	74.0		SSViT-B	57	9.6	**49.4**	**55.6**
MOAT-1 [68]	42	9.1	74.2		Swin-B [40]	88	15.4	35.8	46.6
SSViT-B	57	9.6	**75.7**		MAE-ViT-B [23]	86	17.5	35.9	48.3
DeiT-B [59]	86	17.5	71.5		RVT-B* [42]	92	17.7	28.5	48.7
MOAT-2 [68]	73	17.2	74.3		FAN-L-Hybrid [75]	77	16.9	41.8	53.2
SSViT-L	100	18.2	**76.1**		SSViT-L	100	18.2	**55.0**	**59.2**

Table 6: Evaluation of the model's robustness.

Model	Params(M)	FLOPs(G)	Top1-acc(%)	AP^b	AP^m	mIoU(%)
DeiT-S [59]	22	4.6	79.8	–	–	–
SS-DeiT-S	22	4.3(-0.3)	81.3(+1.5)	–	–	–
Swin-T [40]	29	4.5	81.3	43.7	39.8	44.5
SS-Swin-T	29	4.8	83.8(+2.5)	47.8(+4.1)	43.3(+3.5)	49.3(+4.8)
SSViT-T	15	2.4	**83.0**	**47.3**	**42.6**	**46.8**
$S^3A \rightarrow$WSA [40]	15	2.3	81.3	44.0	39.8	42.7
$S^3A \rightarrow$CSWSA [11]	15	2.3	81.6	44.6	40.3	43.7
w/o LCE	15	2.4	82.8	47.0	42.4	46.5
w/o CPE	15	2.4	82.9	47.3	42.4	46.7
w/o Stem	15	2.1	82.8	47.1	42.4	46.4

Table 7: Ablation study.

4.5 Ablation Study

SSViT v.s. DeiT& Swin. As illustrated in Tab. 7, we position SSViT in comparison with DeiT [59] and Swin [40]. By exclusively substituting the Self-Attention/Window Self-Attention module in DeiT/Swin with S^3A, we formulate SS-DeiT/SS-Swin-T. Remarkably, SS-DeiT-S, despite demanding fewer FLOPs, outperforms DeiT-S, registering a notable performance gain of **+1.5**, thereby underlining the potency of S^3A. In contrast to Swin-T, SS-Swin-T exhibits substantial enhancements across a broad range of downstream tasks.

S^3A v.s. WSA. The Window Self-Attention (WSA), integral to the Swin Transformer [40], is widely employed. In Tab. 7, we contrast S^3A with WSA, revealing S^3A's significant superiority in both classification and downstream tasks. Specifically, using SSViT-T as the benchmark, S^3A achieves a substantial **+1.7** boost in classification accuracy and a **+3.3** increase in box AP over WSA.

S^3A v.s. CSWSA. CSWSA, an enhancement of WSA introduced in CSwin-Transformer [11], outperforms its predecessor. Yet, our S^3A still surpasses CSWSA in key performance metrics. Notably, S^3A attains a classification accuracy **+1.4** points higher than CSWSA. Within the Semantic FPN segmentation framework, S^3A exceeds CSWSA by a substantial **+3.1** mIoU.

LCE. LCE, a straightforward depth-wise convolutional component, is employed to amplify the model's capacity to capture local features. We perform ablation studies on LCE, revealing its contribution to the model's performance enhancement. The results, presented in Tab. 7, indicate that LCE increases the model's classification accuracy by **+0.2**, box AP by **+0.3**, and mask AP by **+0.2**.

CPE. CPE [6] is a versatile, plug-and-play positional encoding strategy, frequently employed to impart positional information to the model. Comprising only a 3x3 depth-wise convolution in a residual block, CPE provides modest performance improvements as illustrated in Tab. 7, with an approximate increase of **+0.1** in classification accuracy.

Conv Stem. The Conv Stem, deployed in the initial stages of the model, aids in extracting refined local features. Tab. 7 suggests that the Conv Stem somewhat bolsters the model's performance in both classification and downstream tasks, specifically enhancing classification accuracy by **+0.2** and the mean Intersection over Union (mIoU) by **+0.4**.

5 Conclusion

Motivated by the human eye's efficient sparse scanning mechanism for visual information processing, we propose the Sparse Scan Self-Attention mechanism (S^3A). This mechanism emulates the human eye's procedural operation: initially selecting anchors of interest, subsequently extracting local information around these anchors, and ultimately aggregating this information. Harnessing the power of S^3A, we develop the Sparse Scan Vision Transformer (SSViT), a robust vision backbone designed for a variety of vision tasks. We assess SSViT across a range of common visual tasks such

as image classification, object detection, instance segmentation, and semantic segmentation, where it consistently showcases impressive performance. Notably, SSViT also exhibits remarkable robustness towards out-of-distribution (OOD) data.

References

[1] Moab Arar, Ariel Shamir, and Amit H. Bermano. Learned queries for efficient local attention. In *CVPR*, 2022.

[2] Zhaowei Cai and Nuno Vasconcelos. Cascade r-cnn: Delving into high quality object detection. In *CVPR*, 2018.

[3] Chun-Fu (Richard) Chen, Quanfu Fan, and Rameswar Panda. CrossViT: Cross-Attention Multi-Scale Vision Transformer for Image Classification. In *ICCV*, 2021.

[4] Chun-Fu (Richard) Chen, Rameswar Panda, and Quanfu Fan. RegionViT: Regional-to-Local Attention for Vision Transformers. In *ICLR*, 2022.

[5] Kai Chen, Jiaqi Wang, Jiangmiao Pang, et al. MMDetection: Open mmlab detection toolbox and benchmark. *arXiv preprint arXiv:1906.07155*, 2019.

[6] Xiangxiang Chu, Zhi Tian, Bo Zhang, Xinlong Wang, and Chunhua Shen. Conditional positional encodings for vision transformers. In *ICLR*, 2023.

[7] MMSegmentation Contributors. Mmsegmentation, an open source semantic segmentation toolbox, 2020.

[8] Ekin D Cubuk, Barret Zoph, Jonathon Shlens, et al. Randaugment: Practical automated data augmentation with a reduced search space. In *CVPRW*, 2020.

[9] Jia Deng, Wei Dong, Richard Socher, et al. Imagenet: A large-scale hierarchical image database. In *CVPR*, 2009.

[10] Mingyu Ding, Bin Xiao, Noel Codella, et al. Davit: Dual attention vision transformers. In *ECCV*, 2022.

[11] Xiaoyi Dong, Jianmin Bao, Dongdong Chen, et al. Cswin transformer: A general vision transformer backbone with cross-shaped windows. In *CVPR*, 2022.

[12] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, et al. An image is worth 16x16 words: Transformers for image recognition at scale. In *ICLR*, 2021.

[13] Alaaeldin El-Nouby, Hugo Touvron, Mathilde Caron, Piotr Bojanowski, Matthijs Douze, Armand Joulin, Ivan Laptev, Natalia Neverova, Gabriel Synnaeve, Jakob Verbeek, et al. Xcit: Cross-covariance image transformers. *arXiv preprint arXiv:2106.09681*, 2021.

[14] Ralf Engbert and Reinhold Kliegl. Microsaccades uncover the orientation of covert attention. *Vision research*, 43(9):1035–1045, 2003.

[15] Qihang Fan, Huaibo Huang, Jiyang Guan, and Ran He. Rethinking local perception in lightweight vision transformer, 2023.

[16] Qihang Fan, Huaibo Huang, Xiaoqiang Zhou, and Ran He. Lightweight vision transformer with bidirectional interaction. In *NeurIPS*, 2023.

[17] SG-Former: Self guided Transformer with Evolving Token Reallocation. Sucheng ren, xingyi yang, songhua liu, xinchao wang. In *ICCV*, 2023.

[18] Jianyuan Guo, Kai Han, Han Wu, Chang Xu, Yehui Tang, Chunjing Xu, and Yunhe Wang. Cmt: Convolutional neural networks meet vision transformers. In *CVPR*, 2022.

[19] Meng-Hao Guo, Cheng-Ze Lu, Zheng-Ning Liu, Ming-Ming Cheng, and Shi-Min Hu. Visual attention network. *arXiv preprint arXiv:2202.09741*, 2022.

[20] Ali Hassani, Steven Walton, Jiachen Li, Shen Li, and Humphrey Shi. Neighborhood attention transformer. In *CVPR*, 2023.

[21] Ali Hatamizadeh, Greg Heinrich, Hongxu Yin, Andrew Tao, Jose M Alvarez, Jan Kautz, and Pavlo Molchanov. Fastervit: Fast vision transformers with hierarchical attention. In *ICLR*, 2024.

[22] Ali Hatamizadeh, Hongxu Yin, Greg Heinrich, Jan Kautz, and Pavlo Molchanov. Global context vision transformers. In *ICML*, 2023.

[23] Kaiming He, Xinlei Chen, Saining Xie, Yanghao Li, Piotr Dollár, and Ross Girshick. Masked autoencoders are scalable vision learners. In *CVPR*, 2022.

[24] Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross B. Girshick. Mask r-cnn. In *ICCV*, 2017.

[25] Dan Hendrycks, Steven Basart, Norman Mu, Saurav Kadavath, Frank Wang, Evan Dorundo, Rahul Desai, Tyler Zhu, Samyak Parajuli, Mike Guo, Dawn Song, Jacob Steinhardt, and Justin Gilmer. The many faces of robustness: A critical analysis of out-of-distribution generalization. In *ICCV*, 2021.

[26] Dan Hendrycks, Kevin Zhao, Steven Basart, Jacob Steinhardt, and Dawn Song. Natural adversarial examples. In *CVPR*, 2021.

[27] Qibin Hou, Cheng-Ze Lu, Ming-Ming Cheng, and Jiashi Feng. Conv2former: A simple transformer-style convnet for visual recognition. *arXiv preprint arXiv:2211.11943*, 2022.

[28] Gao Huang, Yu Sun, and Zhuang Liu. Deep networks with stochastic depth. In *ECCV*, 2016.

[29] Huaibo Huang, Xiaoqiang Zhou, Jie Cao, Ran He, and Tieniu Tan. Vision transformer with super token sampling. In *CVPR*, 2023.

[30] Huaibo Huang, Xiaoqiang Zhou, and Ran He. Orthogonal transformer: An efficient vision transformer backbone with token orthogonalization. In *NeurIPS*, 2022.

[31] David H. Hubel and Torsten N. Wiesel. Receptive fields, binocular interaction and functional architecture in the cat's visual cortex. *The Journal of physiology*, 160:106–154, 1962.

[32] Zi-Hang Jiang, Qibin Hou, Li Yuan, Daquan Zhou, Yujun Shi, Xiaojie Jin, Anran Wang, and Jiashi Feng. All tokens matter: Token labeling for training better vision transformers. In *NeurIPS*, 2021.

[33] Alexander Kirillov, Ross Girshick, Kaiming He, and Piotr Dollár. Panoptic feature pyramid networks. In *CVPR*, 2019.

[34] Youngwan Lee, Jonghee Kim, Jeffrey Willette, and Sung Ju Hwang. Mpvit: Multi-path vision transformer for dense prediction. In *CVPR*, 2022.

[35] Kunchang Li, Yali Wang, Peng Gao, Guanglu Song, Yu Liu, Hongsheng Li, and Yu Qiao. Uniformer: Unified transformer for efficient spatiotemporal representation learning, 2022.

[36] Yanghao Li, Chao-Yuan Wu, Haoqi Fan, Karttikeya Mangalam, Bo Xiong, Jitendra Malik, and Christoph Feichtenhofer. Mvitv2: Improved multiscale vision transformers for classification and detection. In *CVPR*, 2022.

[37] Tsung-Yi Lin, Priya Goyal, Ross B. Girshick, and Kaiming He andPiotr Dollár. Focal loss for dense object detection. In *ICCV*, 2017.

[38] Tsung-Yi Lin, Michael Maire, Serge Belongie, et al. Microsoft coco: Common objects in context. In *ECCV*, 2014.

[39] Weifeng Lin, Ziheng Wu, Jiayu Chen, Jun Huang, and Lianwen Jin. Scale-aware modulation meet transformer. In *ICCV*, 2023.

[40] Ze Liu, Yutong Lin, Yue Cao, Han Hu, Yixuan Wei, Zheng Zhang, Stephen Lin, and Baining Guo. Swin transformer: Hierarchical vision transformer using shifted windows. In *ICCV*, 2021.

[41] Zhuang Liu, Hanzi Mao, Chao-Yuan Wu, et al. A convnet for the 2020s. In *CVPR*, 2022.

[42] Xiaofeng Mao, Gege Qi, Yuefeng Chen, Xiaodan Li, Ranjie Duan, Shaokai Ye, Yuan He, and Hui Xue. Towards robust vision transformer. In *CVPR*, 2022.

[43] Susana Martinez-Conde and Stephen L Macknik. Fixational eye movements across vertebrates: comparative dynamics, physiology, and perception. *Journal of Vision*, 8(14):1–16, 2008.

[44] Susana Martinez-Conde, Stephen L Macknik, and David H Hubel. The role of fixational eye movements in visual perception. *Nature Reviews Neuroscience*, 5(3):229–240, 2004.

[45] Jonathan J. Nassi and Edward M. Callaway. Parallel processing strategies of the primate visual system. *Nature Reviews Neuroscience*, 10:360–372, 2009.

[46] Junting Pan, Adrian Bulat, Fuwen Tan, et al. Edgevits: Competing light-weight cnns on mobile devices with vision transformers. In *ECCV*, 2022.

[47] Zizheng Pan, Jianfei Cai, and Bohan Zhuang. Fast vision transformers with hilo attention. In *NeurIPS*, 2022.

[48] Martina Poletti and Michele Rucci. A compact field guide to the study of microsaccades: Challenges and functions. *Vision research*, 118:83–97, 2016.

[49] Boris T Polyak and Anatoli B Juditsky. Acceleration of stochastic approximation by averaging. *arXiv preprint arXiv:1906.07155*, 2019.

[50] Yongming Rao, Wenliang Zhao, Benlin Liu, Jiwen Lu, Jie Zhou, and Cho-Jui Hsieh. Dynamicvit: Efficient vision transformers with dynamic token sparsification. In *NeurIPS*, 2021.

[51] Yongming Rao, Wenliang Zhao, Yansong Tang, Jie Zhou, Ser-Lam Lim, and Jiwen Lu. Hornet: Efficient high-order spatial interactions with recursive gated convolutions. In *NeurIPS*, 2022.

[52] Benjamin Recht, Rebecca Roelofs, Ludwig Schmidt, and Vaishaal Shankar. Do imagenet classifiers generalize to imagenet?, 2019.

[53] Sucheng Ren, Daquan Zhou, Shengfeng He, Jiashi Feng, and Xinchao Wang. Shunted self-attention via multi-scale token aggregation. In *CVPR*, 2022.

[54] Michele Rucci, Ramona Iovin, Martina Poletti, and Francesca Santini. Miniature eye movements enhance fine spatial detail. *Nature*, 447(7146):851–854, 2007.

[55] Michele Rucci and Martina Poletti. Control and function of fixational eye movements. *Annual Review of Vision Science*, 1:499–518, 2015.

[56] Chenyang Si, Weihao Yu, Pan Zhou, Yichen Zhou, Xinchao Wang, and Shuicheng YAN. Inception transformer. In *NeurIPS*, 2022.

[57] Shitao Tang, Jiahui Zhang, Siyu Zhu, et al. Quadtree attention for vision transformers. In *ICLR*, 2022.

[58] Kelly G Thompson, Kathryn L Biscoe, and Takashi R Sato. Neuronal basis of covert spatial attention in the frontal eye field. *Journal of Neuroscience*, 25(41):9479–9487, 2005.

[59] Hugo Touvron, Matthieu Cord, Matthijs Douze, et al. Training data-efficient image transformers & distillation through attention. In *ICML*, 2021.

[60] Zhengzhong Tu, Hossein Talebi, Han Zhang, Feng Yang, Peyman Milanfar, Alan Bovik, and Yinxiao Li. Maxvit: Multi-axis vision transformer. In *ECCV*, 2022.

[61] Ashish Vaswani, Noam Shazeer, Niki Parmar, et al. Attention is all you need. In *NeurIPS*, 2017.

[62] Wenhai Wang, Jifeng Dai, Zhe Chen, Zhenhang Huang, Zhiqi Li, Xizhou Zhu, Xiaowei Hu, Tong Lu, Lewei Lu, Hongsheng Li, et al. Internimage: Exploring large-scale vision foundation models with deformable convolutions. In *CVPR*, 2023.

[63] Wenhai Wang, Enze Xie, Xiang Li, Deng-Ping Fan, Kaitao Song, Ding Liang, Tong Lu, Ping Luo, and Ling Shao. Pyramid vision transformer: A versatile backbone for dense prediction without convolutions. In *ICCV*, 2021.

[64] Wenhai Wang, Enze Xie, Xiang Li, Deng-Ping Fan, Kaitao Song, Ding Liang, Tong Lu, Ping Luo, and Ling Shao. Pvtv2: Improved baselines with pyramid vision transformer. *Computational Visual Media*, 8(3):1–10, 2022.

[65] Wenxiao Wang, Lu Yao, Long Chen, Binbin Lin, Deng Cai, Xiaofei He, and Wei Liu. Crossformer: A versatile vision transformer hinging on cross-scale attention. In *ICLR*, 2022.

[66] Zhuofan Xia, Xuran Pan, Shiji Song, Li Erran Li, and Gao Huang. Vision transformer with deformable attention. In *CVPR*, 2022.

[67] Tete Xiao, Yingcheng Liu, Bolei Zhou, Yuning Jiang, and Jian Sun. Unified perceptual parsing for scene understanding. In *ECCV*, 2018.

[68] Chenglin Yang, Siyuan Qiao, Qihang Yu, et al. Moat: Alternating mobile convolution and attention brings strong vision models. In *ICLR*, 2023.

[69] Chenglin Yang, Yilin Wang, Jianming Zhang, et al. Lite vision transformer with enhanced self-attention. In *CVPR*, 2022.

[70] Jianwei Yang, Chunyuan Li, Pengchuan Zhang, Xiyang Dai, Bin Xiao, Lu Yuan, and Jianfeng Gao. Focal self-attention for local-global interactions in vision transformers. In *NeurIPS*, 2021.

[71] Rui Yang, Hailong Ma, Jie Wu, Yansong Tang, Xuefeng Xiao, Min Zheng, and Xiu Li. Scalablevit: Rethinking the context-oriented generalization of vision transformer. In *ECCV*, 2022.

[72] Sangdoo Yun, Dongyoon Han, Seong Joon Oh, et al. Cutmix: Regularization strategy to train strong classifiers with localizable features. In *ICCV*, 2019.

[73] Hongyi Zhang, Moustapha Cisse, Yann N Dauphin, et al. mixup: Beyond empirical risk minimization. In *ICLR*, 2018.

[74] Bolei Zhou, Hang Zhao, Xavier Puig, et al. Scene parsing through ade20k dataset. In *CVPR*, 2017.

[75] Daquan Zhou, Zhiding Yu, Enze Xie, Chaowei Xiao, Anima Anandkumar, Jiashi Feng, and Jose M. Alvarez. Understanding the robustness in vision transformers. In *ICML*, 2022.

[76] Lei Zhu, Xinjiang Wang, Zhanghan Ke, Wayne Zhang, and Rynson Lau. Biformer: Vision transformer with bi-level routing attention. In *CVPR*, 2023.

Supplementary Material

In this supplementary material, we first provide the architecture details of our SSViT model, following by the experimental settings on different vision downstream tasks. And then we briefly discuss about the limitations and broader impacts of our study.

Model	Blocks	Channels	Heads	Ratios	Params(M)	FLOPs(G)
SSViT-T	[2, 2, 9, 2]	[64, 128, 256, 512]	[2, 4, 8, 16]	3	15	2.4
SSViT-S	[3, 5, 18, 4]	[64, 128, 256, 512]	[2, 4, 8, 16]	3	27	4.4
SSViT-B	[4, 9, 25, 9]	[80, 160, 320, 512]	[5, 5, 10, 16]	3	57	9.6
SSViT-L	[4, 9, 25, 9]	[112, 224, 448, 640]	[7, 7, 14, 20]	3	100	18.2
SS-Swin-T	[2, 2, 6, 2]	[96, 192, 384, 768]	[3, 6, 12, 24]	4	29	4.8
SS-Swin-S	[2, 2, 18, 2]	[96, 192, 384, 768]	[3, 6, 12, 24]	4	50	8.9
SS-Swin-B	[2, 2, 18, 2]	[128, 256, 512, 1024]	[4, 8, 16, 32]	4	88	15.7
SS-FasterViT-0	[2, 3, 6, 5]	[64, 128, 256, 512]	[–, –, 8, 16]	4	25	3.4
SS-FasterViT-1	[1, 3, 8, 5]	[80, 160, 320, 640]	[–, –, 8, 16]	4	41	5.4
SS-FasterViT-2	[3, 3, 8, 5]	[96, 192, 384, 768]	[–, –, 12, 24]	4	58	8.8

Table 8: Detailed Architectures of our models.

A Architecture Details

The architecture details are illustrated in Table 8. In SSViT, for the convolution stem, we adopt four 3×3 convolutions to embed the input image into tokens. batch normalization and GELU are used after each convolution. 3×3 convolutions with stride 2 are used between stages to reduce the feature resolution. 3×3 depth-wise convolutions are adopted in CPE. While 5×5 depth-wise convolutions are adopted for LCE.

For SS-Swin, we strictly adhere to the design principles of the Swin-Transformer [40] without using additional structures such as CPE or Conv Stem.

For SS-FasterViT, we adhere to the design principles of FasterViT [21]. We use convolutional structures in the first two stages of the model and apply S^3A in the latter two stages.

B Experimental Settings

ImageNet Image Classification.   We adopt the training strategy proposed in DeiT [59], but with the only supervision is classification loss. Specifically, our models are trained from scratch for 300 epochs with the input resolution of 224×224. The AdamW is used with a cosine decay learning rate scheduler and 5 epochs of linear warm-up. The initial learning rate, weight decay, and batch-size are set to 0.001, 0.05, and 1024, respectively. We apply the same data augmentation and regularization used in DeiT [59] (RandAugment [8] (randm9-mstd0.5-inc1) , Mixup [73] (prob = 0.8), CutMix [72] (prob = 1.0), Random Erasing (prob = 0.25), Exponential Moving Average (EMA) [49]). The maximum rates of increasing stochastic depth [28] are set to 0.1/0.15/0.4/0.5 for SSViT-T/S/B/L.

COCO Object Detection and Instance Segmentation.   We apply RetinaNet [37], Mask-RCNN [24], and Cascaded Mask R-CNN [2] as the detection frameworks based on the MMDetection [5]. The models are trained unde "1 ×" (12 training epochs) and "3 × +MS" (36 training epochs with multi-scale training) settings. For the "1 ×" setting, images are resized to the shorter side of 800 pixels while the longer side is within 1333 pixels. For the "3 × +MS", multi-scale training strategy is applied to randomly resize the shorter side between 480 to 800 pixels. We use the AdamW with the initial learning rate of 1e-4. For RetinaNet, we set the weight decay to 1e-4. While for Mask-RCNN and Cascaded Mask R-CNN, we set it to 5e-2.

ADE20K Semantic Segmentation.   Based on MMSegmentation [7], we implement UperNet [67] and SemanticFPN [33] to validate the SSViT. For UperNet, we follow the previous setting of Swin-

Transformer [40] and train the model for 160k iterations with the input size of 512×512. For SemanticFPN, we also use the input resolution of 512×512 but train the models for 80k iterations.

Robustness Evaluation. Follow the previous works [75, 42], we evaluate the SSViT's robustness on the ImageNet-A [26] and ImageNet-R [25]. We also validated the model on ImageNet-V2 [52] to check for overfitting. All models are pretrained on ImageNet-1k.

C Limitations and Future Work.

While the Sparse Scan Self-Attention mechanism (S^3A) and the Sparse Scan Vision Transformer (SSViT) have demonstrated significant computational efficiency and strong performance across various tasks, there are still several limitations that need to be addressed. One notable limitation is the computational constraints that prevented us from experimenting with larger models and datasets such as ImageNet-21k. Exploring the potential of SSViT on such large-scale datasets could provide further insights into its scalability and robustness. In the future, we will strive to validate the performance of SSViT on large datasets and with larger models.

D Broader Impact Statement.

The development of the Sparse Scan Self-Attention mechanism (S^3A) and the Sparse Scan Vision Transformer (SSViT) has the potential to impact the field of computer vision by offering a more efficient alternative to traditional Transformers. By mimicking the human eye's sparse scanning mechanism, SSViT reduces computational load and improves the efficiency of vision models, which could lead to broader applications in resource-constrained environments.

The proposed SSViT is a general vision backbone that can be applied on different vision tasks, e.g., image classification, object detection instance segmentation, and semantic segmentation. It has no direct negative social impact. Possible malicious uses of SSViT as a general-purpose backbone are beyond the scope of our study to discuss.

E Code

We provide the code of our Sparse Scan Self-Attention.

```
import torch.nn as nn
import torch
from einops import rearrange
from natten.functional import natten2dqkrpb, natten2dav

class S3A(nn.Module):

    def __init__(self, embed_dim, num_heads, window_size, anchor_size,
                 stride):
        super().__init__()
        self.embed_dim = embed_dim
        self.num_heads = num_heads
        self.window_size = window_size
        self.anchor_size = anchor_size
        self.stride = stride
        self.head_dim = embed_dim // num_heads
        self.scaling = self.head_dim ** -0.5
        self.qkv = nn.Conv2d(embed_dim, embed_dim*3, 1, bias=True)
        self.out_proj = nn.Conv2d(embed_dim, embed_dim, 1, bias=True)

    def forward(self, x: torch.Tensor):
        '''
        x: (b c h w)
        '''
        bsz, _, h, w = x.size()
```

```python
26            qkv = self.qkv(x) # (b 3*c h w)
27
28            q, k, v = rearrange(qkv, 'b (m n d) h w -> m b n h w d', m=3,
                  n=self.num_heads)
29
30            k = k * self.scaling
31
32            window_size = self.window_size
33            anchor_size = self.anchor_size
34
35            attn = natten2dqkrpb(q, k, None, window_size, 1)
36            attn = attn.softmax(dim=-1)
37            v = natten2dav(attn, v, window_size, 1)
38
39            stride = self.stride
40
41            attn = natten2dqkrpb(q, k, None, anchor_size, stride)
42            attn = attn.softmax(dim=-1)
43            v = natten2dav(attn, v, anchor_size, stride)
44
45            res = rearrange(v, 'b n h w d -> b (n d) h w')
46            return self.out_proj(res)
```