Making Vision Transformers Efficient from A Token Sparsification View

Shuning Chang[1*] Pichao Wang[2†] Ming Lin[2‡] Fan Wang[2] David Junhao Zhang[1]
Rong Jin[2] Mike Zheng Shou[1§]
[1]Show Lab, National University of Singapore [2]Alibaba Group
{changshuning, junhao.zhang}@u.nus.edu, {fan.w, jinrong.jr}@alibaba-inc.com, minglamz@amazon.com,
{pichaowang, mike.zheng.shou}@gmail.com

TUE-PM

https://github.com/changsn/STViT-R

https://arxiv.org/abs/2303.08685

Overview

Efficient ViT
- Linear self-attention (Swin)
- Token Sparsification (Ours)

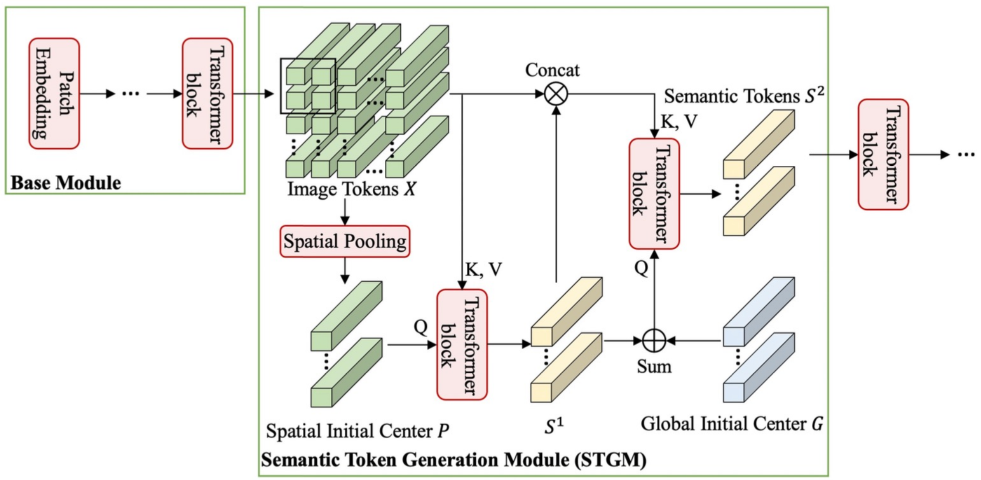
Issues in token sparsification

(i) Dramatic accuracy drops;

(ii) Application difficulty in the local vision transformer;

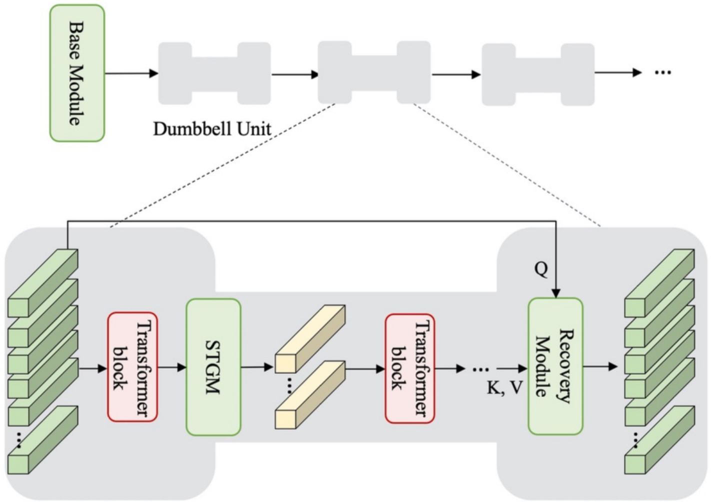
(iii) Non-general-purpose networks for downstream tasks.

STViT



A few tokens with high-level semantic representations can achieve both high performance and efficiency.

STViT-R



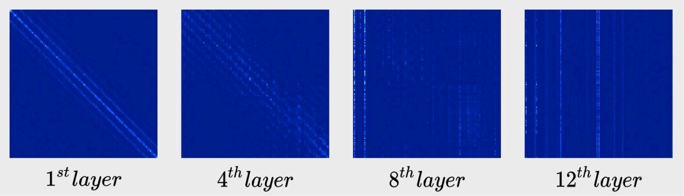
Restore full resolution feature map to achieve downstream tasks.

Observation

(i) Unlike local CNNs, ViT discretizes feature map as tokens.

(ii) Discrete tokens are more beneficial for optimization [1].

(iii) There are only several vertical lines in the deep layers in the attention maps in different transformer layers.



Employing a few discrete tokens with high-level semantic information can potentially achieve both high performance and efficiency.

[1] Pichao Wang, Xue Wang, Hao Luo, Jingkai Zhou, Zhipeng Zhou, Fan Wang, Hao Li, and Rong Jin. Scaled relu matters for training vision transformers.

Method

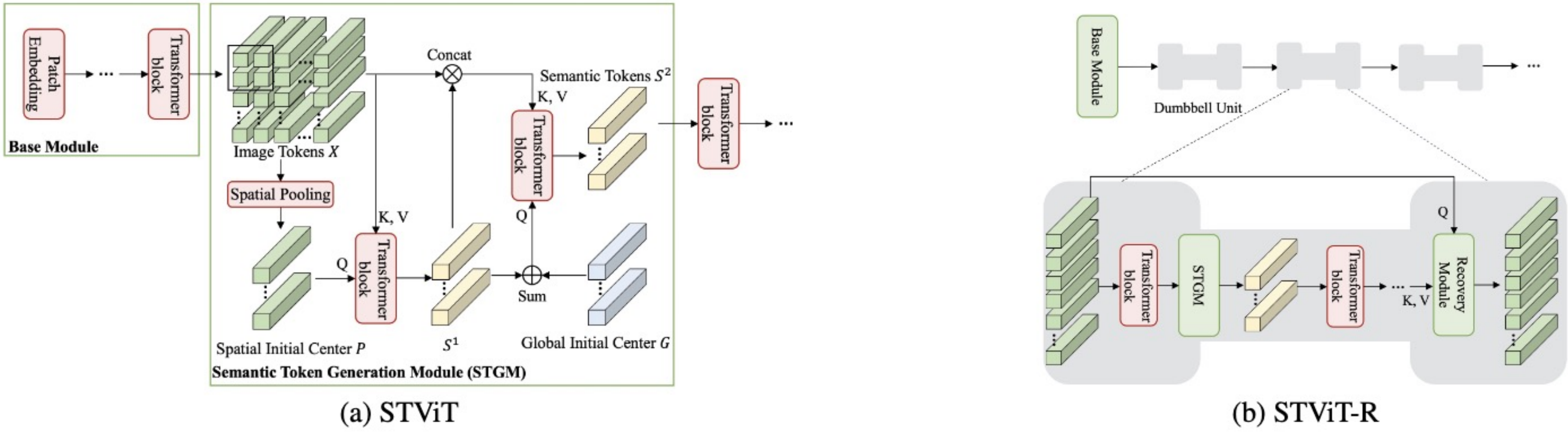


Figure 2. The architectures of our STViT and STViT-R.

Semantic token generation module (STGM)

Self-attention can conduct cluster center recovery (Sup. A.7)

- Spatial semantic tokens
- Global semantic tokens

STViT for downstream tasks

- Dumbbell units
- Recovery module

STViT in local vision transformers

Results

- Image classificationx

Model	Metrics	Base	No. of semantic tokens 16	36	64	100
STViT-DeiT-T	Top-1 Acc(%)	72.2	72.2(+0.0%)	72.7(+0.5)	73.0(+0.8)	73.2(+1.0)
	FLOPs(G)	1.26	0.53(-58%)	0.60(-52%)	0.71(-44%)	0.86(-32%)
	Throughput(img/s)	2752	5511(+101%)	4769(+74%)	4214(+53%)	3551(+29%)
STViT-DeiT-S	Top-1 Acc(%)	79.8	79.8(+0.0)	80.1(+0.3)	80.5(+0.7)	80.6(+0.8)
	FLOPs(G)	4.58	1.91(-58%)	2.20(-52%)	2.62(-43%)	3.16(-31%)
	Throughput(img/s)	1408	2891(+105%)	2542(+80%)	2229(+58%)	1837(+30%)
STViT-DeiT-B	Top-1 Acc(%)	81.8	81.8(+0.0)	82.2(+0.4)	82.6(+0.8)	82.7(+0.9)
	FLOPs(G)	17.58	7.31(-58%)	8.44(-52%)	10.04(-43%)	12.13(-31%)
	Throughput(img/s)	626	1308(+110%)	1150(+85%)	1087(+61%)	826(+33%)

Table 1. Applying STViT to DeiT-T, DeiT-S, and DeiT-B. The top-1 accuracy, complexity in FLOPs, and throughput are reported for different numbers of semantic tokens.

Model	Metrics	Base	Move STGM	No. of semantic tokens 4	9	16
STViT-Swin-T	Top-1 Acc(%)	81.3	81.0(-0.3%)	80.8(-0.5)	81.5(+0.2)	81.8(+0.5%)
	FLOPs(G)	4.5	3.14(-30%)	2.99(-34%)	3.43(-24%)	4.06(-10%)
	Throughput(img/s)	878	1124(+29%)	1128(+29%)	1061(+22%)	1008(+15%)
STViT-Swin-S	Top-1 Acc(%)	83.0	82.8(-0.2%)	82.4(-0.6%)	83.0(-0.0)	83.1(+0.1%)
	FLOPs(G)	8.7	5.95(-32%)	5.95(-32%)	6.53(-25%)	7.36(-15%)
	Throughput(img/s)	551	739(+35%)	732(+34%)	691(+26%)	657(+20%)
STViT-Swin-B	Top-1 Acc(%)	83.5	83.2(-0.3%)	83.0(-0.5)	83.4(-0.1)	83.7(+0.2%)
	FLOPs(G)	15.4	10.48(-32%)	10.48(-32%)	11.51(-25%)	12.97(-16%)
	Throughput(img/s)	415	558(+35%)	551(+33%)	521(+26%)	489(+19%)

Table 2. Applying STViT to Swin-T, Swin-S, and Swin-B. The top-1 accuracy, complexity in FLOPs, and throughput are reported for different numbers of semantic tokens in each window. *Base* indicates the corresponding original Swin model. *Move STGM* indicates changing the default position of STGM.

Model	Top-1 Acc	FLOPs(G)	△
DeiT-S			
DynamicViT [31]	79.3	2.9(-37%)	-0.5
IA-RED^2 [30]	79.1	3.2(-30%)	-0.7
PS-ViT [34]	79.4	2.6(-43%)	-0.4
TokenLearner [32]	76.1	1.9(-44%)	-1.8
DGE [33]	79.7	3.1 (-49%)	-0.6
A-ViT [47]	78.6	3.6 (-39%)	-0.3
Evo-ViT [45]	79.4	3.0(-35%)	-0.4
EViT [24]	78.5	2.3(-50%)	-1.3
STViT(Ours)	79.8	1.91(**-58%**)	**-0.0**
DeiT-B			
IA-RED^2 [30]	80.3	11.8(-33%)	-1.5
DynamicViT [31]	81.3	11.2(-36%)	-0.5
PS-ViT [34]	81.5	9.8(-44%)	-0.3
TokenLearner [32]	83.7	28.7(-48%)	-1.1
Evo-ViT [45]	81.3	10.2(-33%)	-0.5
EViT [24]	80.0	8.7(-51%)	-1.8
STViT(Ours)	81.8	7.31(**-58%**)	**-0.0**

- Downstream tasks

	AP^b	AP^b_{50}	AP^b_{75}	AP^b_s	AP^m	AP^m_{50}	AP^m_{75}	AP^m_s	FLOPs(G)
Swin-S	51.8	70.4	56.3	35.2	44.7	67.9	48.5	28.8	194
STViT-R-Swin-S	51.8	70.6	56.1	36.7	44.7	67.8	48.6	29.0	134(-31%)
Swin-B	51.9	70.9	56.5	35.4	45.0	68.4	48.7	28.9	343
STViT-R-Swin-B	52.2	70.8	56.8	36.5	45.2	68.3	49.1	29.5	233(-32%)

Table 5. Results on COCO object detection and instance segmentation under Cascade Mask R-CNN with 3× schedule. The FLOPs are measured for backbones.

Method	Backbone	mIoU	FLOPs(G)
UperNet	Swin-S	49.3	49
UperNet	STViT-R-Swin-S	48.3	34(-31%)
UperNet	Swin-B	49.7	87
UperNet	STViT-R-Swin-B	48.9	60(-31%)

Table 14. Results of semantic segmentation on the ADE20K val set. A multi-scale inference with resolution $[0.5, 0.75, 1.0, 1.25, 1.5, 1.75]\times$ is applied. FLOPs and latency are measured in backbones with resolution 512×512.

Thank you!