ToaSt:
Token Channel Selection and Structured Pruning for Efficient ViT

***Hyunchan Moon*[1], *Cheonjun Park*[2†], *Steven L. Waslander*[3]**

[1]LG Electronics, [2]Hankuk University of Foreign Studies, [3]University of Toronto

† correspondence author : jun@hufs.ac.kr

Background

Why ViT compression?

- ViT achieves SOTA across classification, detection, segmentation — but far heavier than CNNs at equal accuracy
- Advantage appears only at large model/data scale → hard to deploy on mobile/edge

Where does the cost come from?

- Attention: quadratic in **sequence length N, O(N²)**
- **Most FLOPs scale with hidden dim D**, yet prior methods reduce N or prune weights — not D
- (FFN ≈ 61% of FLOPs, and grows to ~72% at ViT-22B as width scales)

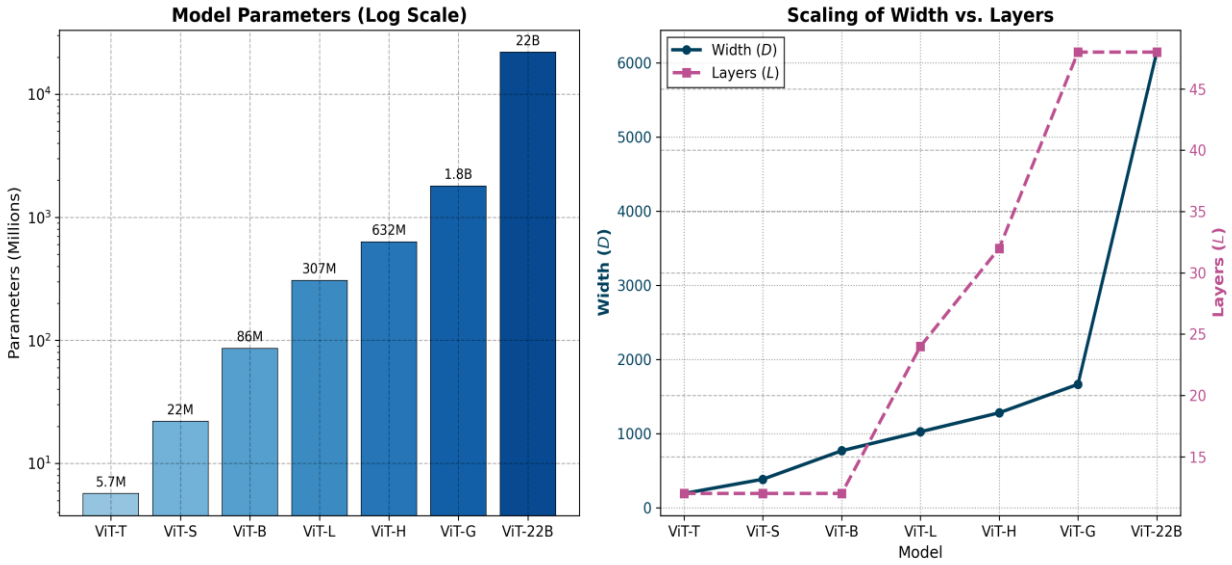

Fig1. ViT scaling — width (D) grows while depth saturates at larger scales

Operation	Complexity	FLOPs (%)	Primary Factor
Q, K, V Projections	$\mathcal{O}(3ND^2)$	~23.0%	Hidden dim D
Attention Score $(QK^\top)$	$\mathcal{O}(N^2D)$	~3.9%	Sequence length N
Attention Output (AV)	$\mathcal{O}(N^2D)$	~3.9%	Sequence length N
Output Projection	$\mathcal{O}(ND^2)$	~7.7%	Hidden dim D
FC1	$\mathcal{O}(4ND^2)$	~30.7%	Hidden dim D
FC2	$\mathcal{O}(4ND^2)$	~30.7%	Hidden dim D

Table 1. ViT FLOPs breakdown — most cost scales with hidden dim D

Motivation

Two main approaches

- **Structured weight pruning**
  - Removes heads/channels/blocks → needs **costly full-model retraining** to recover accuracy
- **Token compression (reduce N)**
  - Cuts attention's O(N²), but reduces FFN only linearly with N → dominant **O(D²) FFN cost remains**
  - Token-level decisions **propagate across layers** → complex optimization

✓ **ToaSt addresses both by decoupling the strategy per component**

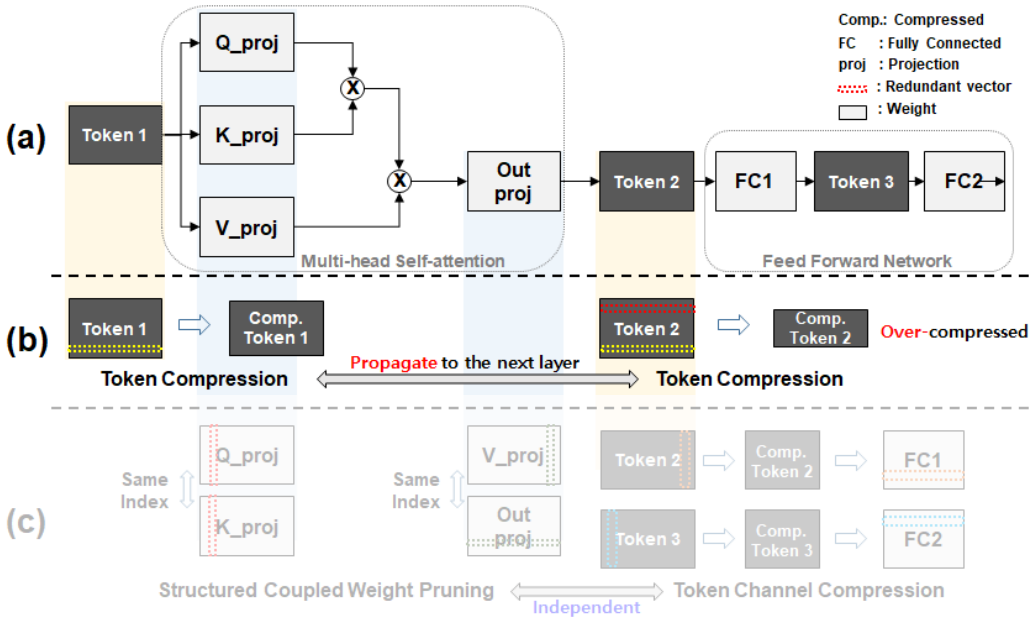

Fig2. (a) standard ViT block, (b) token compression propagates across layers

Motivation

Two main approaches

- **Structured weight pruning**
  - Removes heads/channels/blocks → needs **costly full-model retraining** to recover accuracy
- **Token compression (reduce N)**
  - Cuts attention's O(N²), but reduces FFN only linearly with N → dominant **O(D²) FFN cost remains**
  - Token-level decisions **propagate across layers** → complex optimization

✓ **ToaSt addresses both by decoupling the strategy per component**

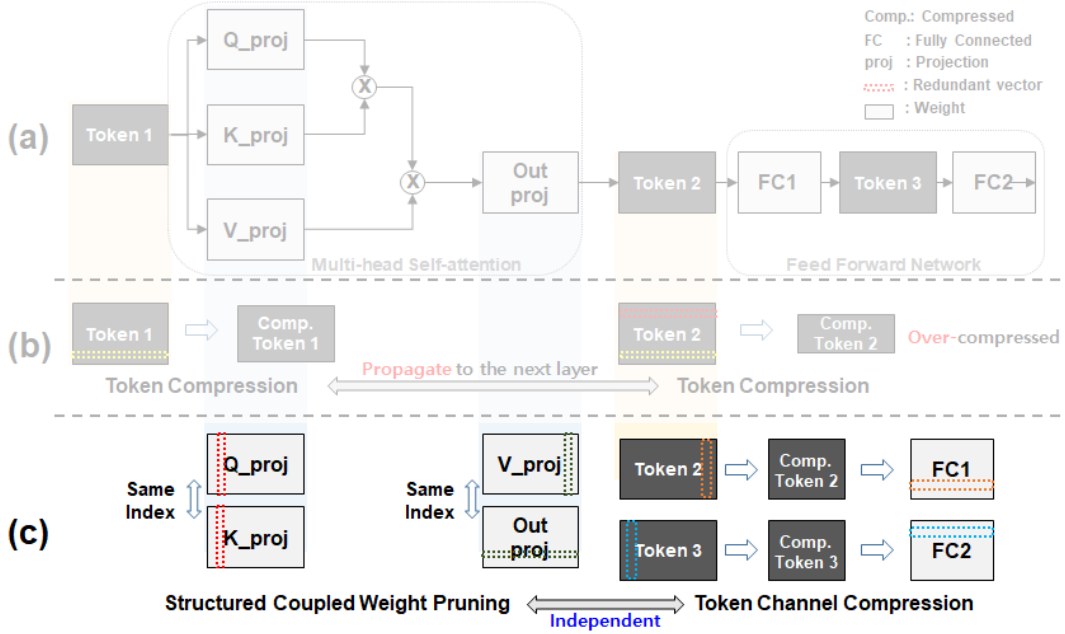

Fig2. (c) ToaSt compresses each layer independently

Proposed Method: ToaSt

Layer-Independent Compression — decoupling the strategy per component

- **MHSA** → **Structured Coupled Weight Pruning**
- **FFN** → **Token Channel Selection**, training-free
- Each layer compressed independently → **no inter-layer propagation**

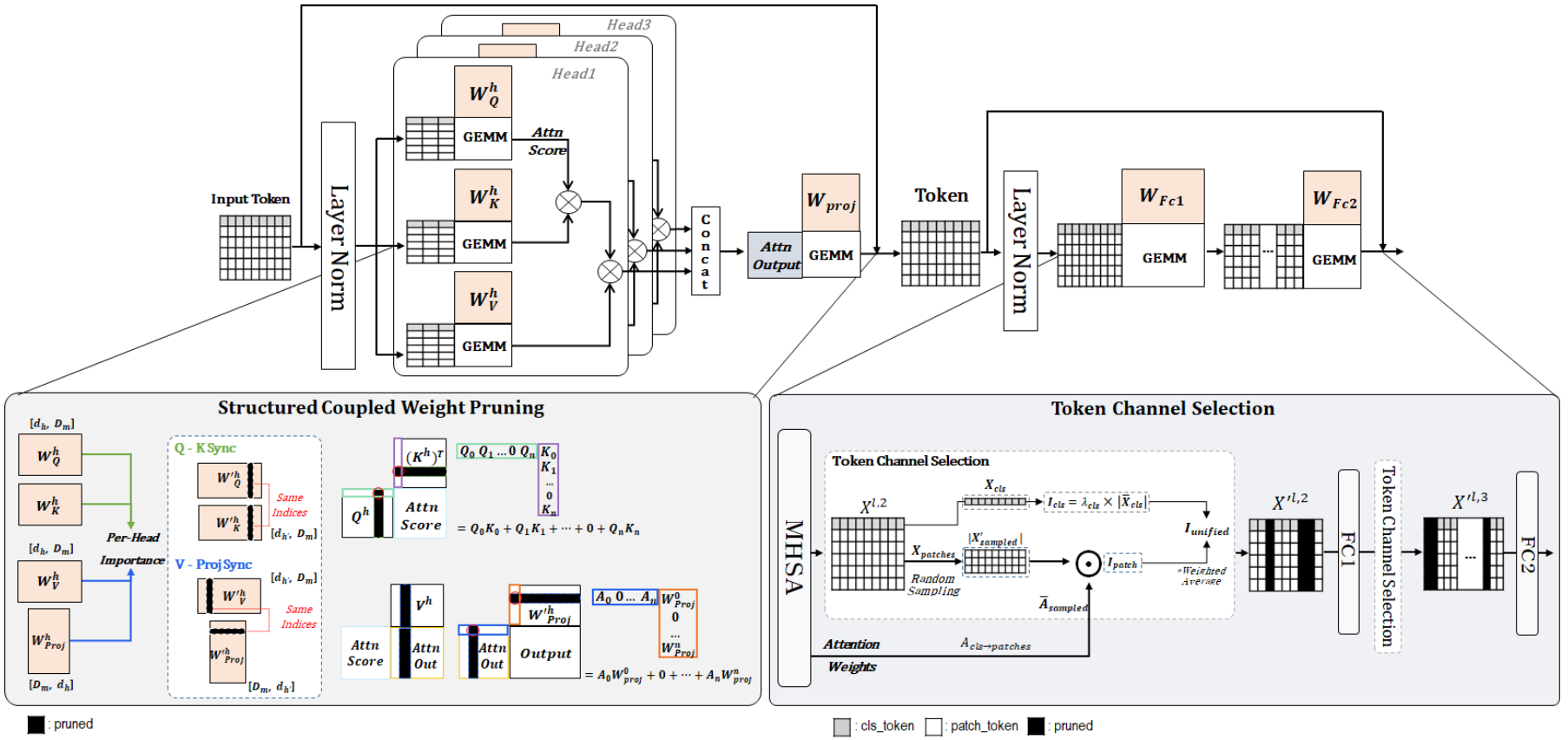

Fig3. ToaSt overview : coupled weight pruning (MHSA) + token channel selection (FFN), each layer compressed independently

Proposed Method: Structured Coupled Weight Pruning

Goal: reduce internal head dim dk (not global D) → preserves residual interface

- **Prune Q-K and V-Proj with synchronized indices**
  - Importance via Geometric Median (closest to center = most redundant)
- **Head-wise uniform → regular, GPU-friendly matmul**

✓ **Synchronization is critical —** non-aligned pruning collapses accuracy

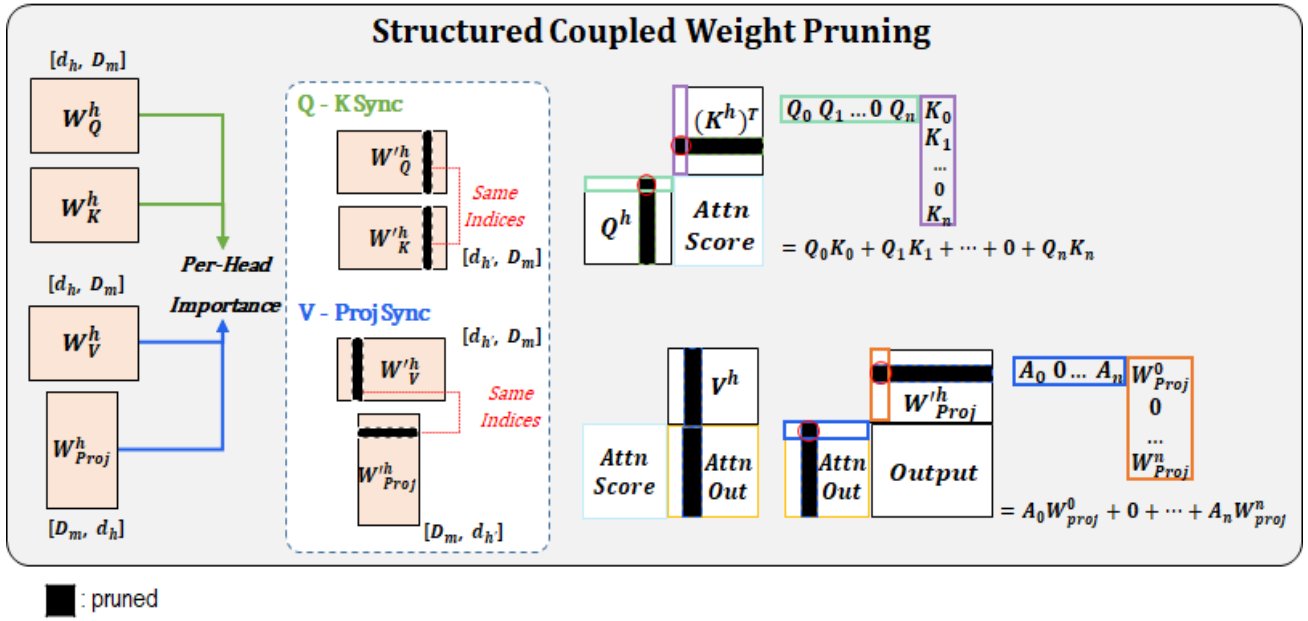

Fig4. Structured Coupled Weight Pruning — Q-K / V-Proj pruned with synchronized indices

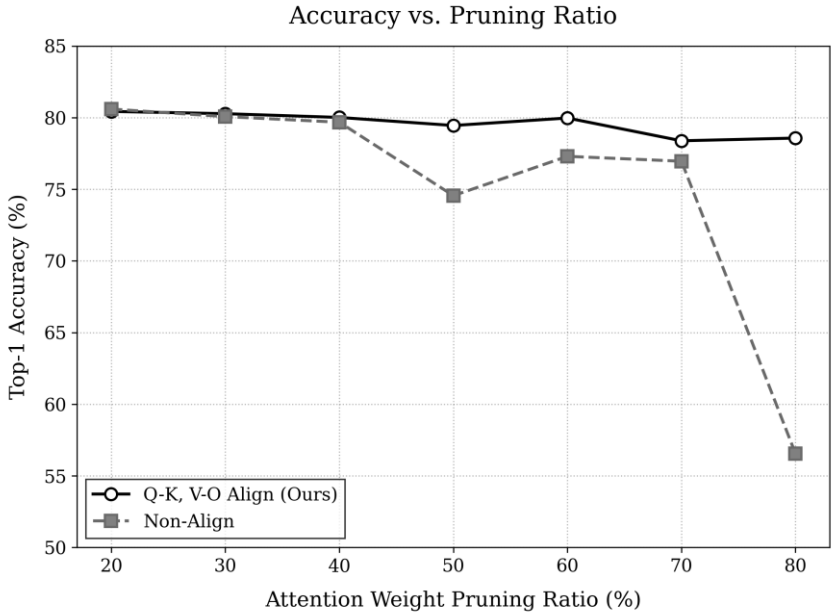

Fig5. Synchronized (Align) pruning preserves accuracy; non-aligned collapses at high ratios

Proposed Method: Token Channel Selection

Why possible: FFN redundancy analysis (pre-trained ViT)

- Deeper layers: high sparsity, collapsing rank, **$R^2 \approx 1.0$** (channels linearly dependent)

How (training-free, per layer):

- $R^2 \approx 1.0$ → sample only **2–20% tokens** to score importance → keep TopK
- **FC1 conservative / FC2 aggressive (up to 90%)**

Two advantages:

- **Cost**: **fully training-free** — no retraining/calibration
- **Accuracy**: **kept** channels have 3–5.5× **higher** SNR → pruned ones are noise-dominant → implicit noise filter, **accuracy *improves***

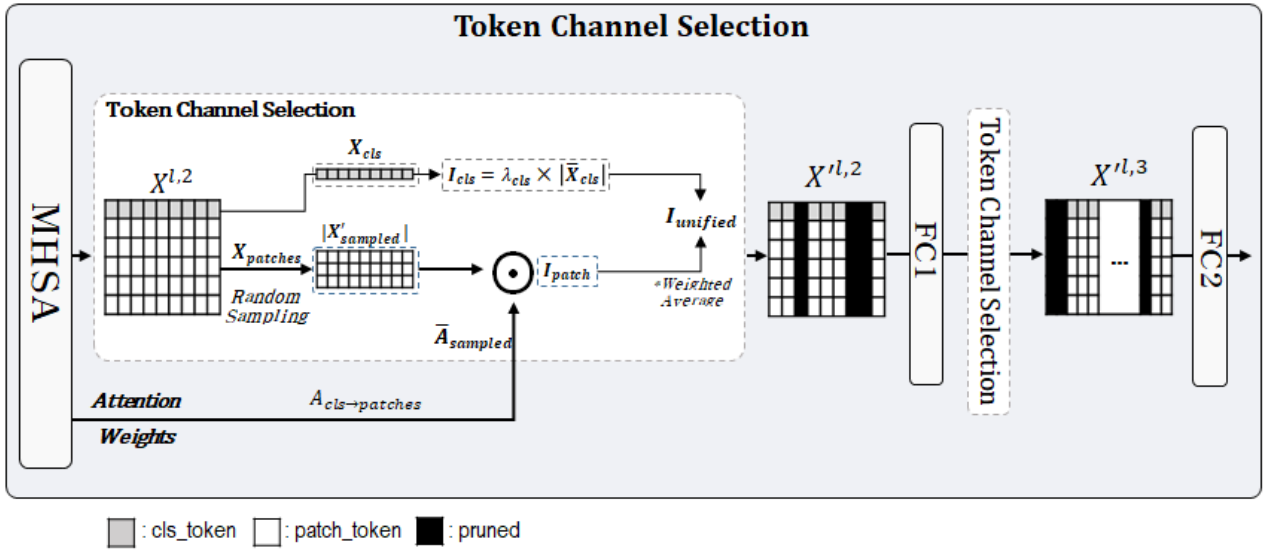

Fig6. Token Channel Selection — sample tokens, score channels, keep TopK

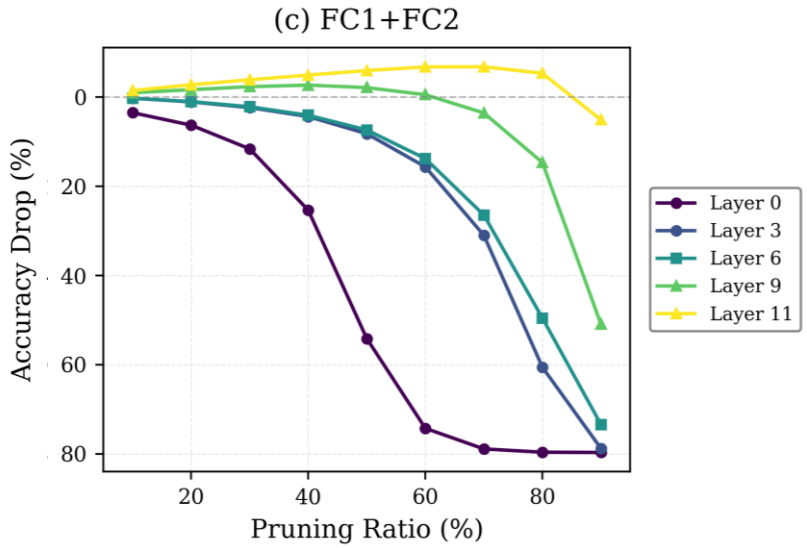

Fig7. Combined FC1+FC2 pruning — deeper layers improve accuracy even under aggressive pruning

Block	Kept SNR	Pruned SNR	Ratio
17	0.198	0.065	3.04×
18	0.229	0.055	4.20×
19	0.282	0.051	5.50×
20	0.409	0.077	5.29×
21	0.562	0.153	3.67×

Table2. Kept channels show 3–5.5× higher SNR than pruned (noise-dominant)

Experiment

Classification (ImageNet-1K)

- **Consistent gains across 9 models / 3 families (DeiT, ViT-MAE, Swin)**
- **ViT-MAE-Huge: 88.52% (+1.64%p), 39.4% FLOPs↓**
- **DeiT-Small: 83.40% (+3.58%p), 2.07× speedup**

Model	Method	Top-1 (%)	Top-5 (%)	GFLOPs	FLOPs ↓ (%)	Throughput (img/s)	Speedup
DeiT-Small	Baseline	79.82	94.95	4.6	–	2313.2	1.00×
	ToMe (Bolya et al., 2022)	79.35	94.65	2.7	41.3	2737.1	1.18×
	DiffRate (Chen et al., 2023)	79.56	94.80	2.9	37.0	2808.1	1.21×
	ToaSt (Ours)	**83.40**	**96.97**	**2.5**	**45.7**	**4783.3**	**2.07×**
ViT-MAE-Huge	Baseline	86.88	98.07	167.4	–	129.7	1.00×
	ToMe (Bolya et al., 2022) (r=5)	86.28	97.88	113.9	31.9	185.9	1.43×
	DiffRate (Chen et al., 2023)	86.65	97.88	103.4	38.2	202.9	1.56×
	ToaSt (Ours)	**88.52**	**98.29**	**101.4**	**39.4**	**206.2**	**1.59×**

Table3. ImageNet-1K — consistent gains with fewer FLOPs across families

Generalization & complementarity

- Downstream (training-free TCS): **COCO 52.2 vs 51.9 mAP**, ADE20K 0.00 mIoU drop
- **Orthogonal to token compression** → combine w/ ToMe for additive speedup (**DeiT-S 2.65×**)

Backbone	Method	GFLOPs	Box mAP	Mask mAP
Swin Small	Baseline	194	51.9	45.0
	ToaSt	144 (25.8%↓)	**52.2**	45.0
Swin Base	Baseline	343	51.9	45.0
	ToaSt	251 (26.8%↓)	**52.2**	45.1
	ToaSt	246 (28.3%↓)	51.8	44.9

Setting	GFLOPs	Quality	Δ
ADE20K (Baseline)	87.0	48.13 mIoU	–
ADE20K (4-layer TCS)	82.8	48.13 mIoU	0.00
ADE20K (6-layer TCS)	80.9	47.98 mIoU	−0.15
CIFAR-100 (Baseline)	15.4	89.37%	–
CIFAR-100 (ToaSt)	10.1 (34.4%↓)	89.44%	+0.07

Table4. Downstream — detection, segmentation, classification preserved

Takeaway

- Decoupled, mostly training-free, **compression that improves accuracy** — across 9 models & 3 tasks

Model	Method	Top-1 (%)	Top-5 (%)	GFLOPs	Throughput (img/s)
DeiT-Small	ToMe	79.35	94.65	2.7G	2737.1
	ToaSt	83.40	96.97	2.5G	4783.3
	ToaSt + ToMe (r=13)	79.98	95.64	1.83G	6138.1

Table5. Orthogonal to token compression — additive speedup with ToMe

Thank you!

Contact

mhcqwe92@gmail.com