

1 A Overview

2 We begin by outlining the structure of this supplementary material. §B expands our evaluation by
 3 benchmarking comparing with more state-of-the-art methods across a diverse set of datasets, thereby
 4 demonstrating the robustness and generality of our method. §C presents two key extensions: the
 5 integration of FlashAttention and the adaptation of our method to the LLaVA-1.5-13B backbone,
 6 underscoring its versatility and scalability. Finally, §D provides further implementation details.

7 B Comparison with more methods on more datasets

8 We compare our pruning strategy with six recent training-free token-pruning methods: ToMe [1],
 9 FastV [2], SparseVLM [3], PruMerge+ [4], VisionZip [5] and FasterVLM [6]. All approaches
 10 are evaluated at Avg. tokens of 128 and 64. Tab. 1 reports per-dataset accuracies on GQA [7],
 11 SQA [8], TextVQA [9], POPE [10], MME [11], MMB and MMB^{CN} [12] alongside the overall
 12 relative performance compared to the full-token baseline. At a 128 token budget, we achieve the
 13 highest accuracy on the majority of benchmarks, corresponding to 98.7% (+1.3%) of full-token
 14 performance. Under the more aggressive 64 token budget, our approach remains dominant, while
 15 preserving 97.1% (+3.1%) accuracy.

16 We further extend our analysis by comparing against two pruning methods, VTW [13] and TopV [14]
 17 on SQA [8], MMB [12], POPE [10] and OCRBench [15], each operating at approximately 49–50%
 18 of full-token FLOPs. As shown in Tab. 2, our strategy achieves 60.8 on MMB (+1.1), and matches or
 19 surpasses existing methods on SQA, POPE and OCRBench.

20 These results confirm that our method robustly maintains model performance under substantial token
 21 reduction and consistently outperforms all considered training-free baselines.

22 All reported metrics are taken directly from the original papers [14, 6].

Table 1: Comparison of our methods with other training-free token pruning methods. “Avg.tokens” refers to the average number of tokens that will be retained. Ratio represents the average percentage of performance maintained at the corresponding reduction ratio.

Method	Present at	Avg. tokens	GQA	SQA	TextVQA	POPE	MME	MMB	MMB ^{CN}	Ratio (%)
LLaVA-1.5-7B	NeurIPS’24	576	61.9	69.5	58.2	85.9	1511	64.7	58.3	100.0
ToMe [1]	ICLR’23	128	52.4	59.6	49.1	62.8	1088	53.3	48.8	80.9%
FastV [2]	ECCV’24	128	49.6	60.2	50.6	59.6	1209	56.1	51.4	82.6%
SparseVLM [3]	ICML’25	128	56.0	67.1	54.9	80.5	1376	60.0	51.1	92.4%
PruMerge+ [4]	arXiv’24	128	57.8	67.6	54.3	81.5	1421	61.3	54.7	94.5%
VisionZip [5]	CVPR’25	128	57.6	68.9	56.8	83.2	1432	62.0	56.7	96.4%
FasterVLM [6]	arXiv’24	128	58.2	69.1	57.0	84.6	1461	62.7	57.3	97.4%
Ours	–	128	59.9	69.6	57.7	85.2	1458	64.3	58.3	98.7%
ToMe [1]	ICLR’23	64	48.6	50.0	45.3	52.5	922	43.7	38.9	69.2%
FastV [2]	ECCV’24	64	46.1	51.1	47.8	48.0	1020	48.0	42.7	71.6%
SparseVLM [3]	ICML’25	64	52.7	62.2	51.8	75.1	1221	56.2	46.1	85.4%
PruMerge+ [4]	arXiv’24	64	54.9	68.6	53.0	77.4	1198	59.3	51.0	89.6%
VisionZip [5]	CVPR’25	64	55.1	69.0	55.5	77.0	1366	60.1	55.4	93.1%
FasterVLM [6]	arXiv’24	64	55.4	69.1	55.8	80.4	1370	61.3	55.1	94.0%
Ours	–	64	57.7	69.6	57.1	82.5	1445	63.6	57.1	97.1%

Table 2: Performance comparison with VTW and TopV on SQA, MMBench, POPE and OCRBench.

Method	Present at	FLOPs	SQA	MMB(cn,en)	POPE	OCRBench
LLaVA-v1.5-7B	NeurIPS’24	100%	69.5	61.5	85.9	31.3
VTW [13]	AAAI’25	50%	69.6	59.6	85.9	5.1
TopV [14]	CVPR’25	49%	69.6	59.7	84.2	31.0
Ours	–	49%	69.7	60.8	85.9	31.2

Table 3: Comparison of our methods (FlashAttention version) with PyramidDrop.

Method	Present at	Avg. tokens	MME	MMB	SQA	GQA	TextVQA	Ratio	FLOPs
LLaVA-1.5-7B	NeurIPS'24	576	1862	64.7	69.5	61.9	58.2	100%	100%
PDrop [16]	CVPR'25	192	1797	63.3	69.2	57.3	56.5	96.8%	43.9%
Ours	-	192	1832	64.9	69.6	60.4	57.7	99.0%	42.9%
Ours-Flash	-	192	1807	64.5	69.4	60.0	57.4	98.4%	42.4%
PDrop [16]	CVPR'25	128	1761	61.6	68.4	57.1	56.6	95.6%	35.1%
Ours	-	128	1785	64.3	69.7	59.9	57.4	98.1%	33.7%
Ours-Flash	-	128	1774	63.1	69.3	58.9	56.9	96.6%	33.2%
PDrop [16]	CVPR'25	64	1561	58.8	69.0	47.5	50.6	87.6%	25.5%
Ours	-	64	1745	63.6	69.6	57.7	57.1	96.7%	23.2%
Ours-Flash	-	64	1682	62.6	69.4	56.9	55.7	94.9%	22.9%

23 C Extended Experiment

24 **Extend to FlashAttention.** To be compatible with FlashAttention, similar to PDrop [16], we
 25 computed a more lightweight matrix. After the instruction is tokenized and entered into the model, it
 26 goes through multiple layers of self-attention, and the vector representation of the final token has
 27 fully integrated the information of the entire instruction. Therefore, we use the lightweight matrix
 28 obtained by calculating the dot product between the last text token of the instruction and the image
 29 token for token pruning. As shown in Tab. 3, our method is well compatible with FlashAttention,
 30 which proves the validity and universality of our method.

Table 4: Comparison of our methods with other training-free token pruning methods on LLaVA-1.5-13B. "Avg.tokens" refers to the average number of tokens that will be retained. Ratio represents the average percentage of performance maintained at the corresponding reduction ratio.

Method	Present at	# Token	VQA ^{V2}	GQA	TextVQA	POPE	MME	Ratio (%)
LLaVA-1.5-13B	-	576	80.0	63.3	61.2	86.0	1531	100%
FastV [2]	ECCV'24	288	79.5	62.6	60.9	85.2	1545	99.6%
SparseVLM [3]	ICML'25	288	78.5	59.9	59.5	71.3	1497	94.1%
FasterVLM [6]	arXiv'24	288	79.0	61.0	60.0	86.0	1530	98.6%
Ours	-	288	79.8	63.0	60.9	86.1	1530	99.8%
FastV [2]	ECCV'24	144	77.2	59.9	60.0	79.4	1494	95.8%
SparseVLM [3]	ICML'25	144	76.1	58.0	57.9	68.6	1499	91.8%
FasterVLM [6]	arXiv'24	144	77.4	58.7	59.0	83.1	1467	95.7%
Ours	-	144	79.0	61.5	60.2	86.7	1506	98.7%
FastV [2]	ECCV'24	58	70.3	54.9	55.6	67.3	1360	86.5%
SparseVLM [3]	ICML'25	58	68.3	54.4	52.6	62.6	1285	82.8%
FasterVLM [6]	arXiv'24	58	73.1	56.0	57.4	74.7	1371	90.0%
Ours	-	58	77.2	58.5	59.0	83.6	1478	95.8%

31 **Extend to LLaVA-1.5-13B.** As shown in Tab. 4, at 288 tokens our method achieves an overall
 32 accuracy of 99.8%. At 144 tokens, we achieve 79.0 on VQA^{V2} (+1.6) and 86.7 on POPE (+3.6),
 33 corresponding to 98.7% (+3.0%) overall accuracy. Even when using only 58 tokens our approach
 34 maintains 77.2 on VQA^{V2} (+4.1), 83.6 on POPE (+8.9), and preserves 95.8% of performance,
 35 representing a 5.8% improvement over FasterVLM. By demonstrating similar improvements on
 36 LLaVA-1.5-13B and those observed on LLaVA-1.5-7B, our method proves its strong generalizability
 37 across different model scales. All reported metrics are taken directly from the original papers [6].

38 D More Details

39 **Cross-Modal Weighted Pruning for First Layer** We assign each visual token an importance score
 40 based on the textual-visual attention weight. However, at this early stage the visual and textual tokens
 41 have not yet sufficiently interacted, so text-only pruning may discarding critical visual information.
 42 Previous methods [16, 3] avoid this issue by omitting first layer pruning entirely, yet the underlying

problem persists. To address this, we perform a calibrated cross-modal pruning in first layer by combining both visual and textual cues. Concretely, denote $\alpha_{t,v}$ as the attention weight from the text tokens to visual token v , and $\alpha_{[CLS],v}$ as the attention weight from the visual [CLS] token to v [6]. We use information $I(\mathbf{V}, \mathbf{T})$ with a weight and define the importance score of each visual token as

$$s_v = \lambda I(\mathbf{V}, \mathbf{T}) \alpha_{t,v} + \alpha_{[CLS],v}. \quad (1)$$

where λ is a scaling factor. This formulation ensures that when text–vision alignment is strong (high $I(\mathbf{V}, \mathbf{T})$), tokens with high textual attention are prioritized, whereas when alignment is weak, visual [CLS] saliency guides pruning and preserves essential visual information.

FLOPs Calculation. Following SparseVLM [3] and PDrop [16], let n_i denote the number of visual tokens preserved at the i -th layer. The total computational complexity, measured in floating-point operations (FLOPs), can be written as

$$\text{FLOPs} = \sum_{i=1}^L (4n_i d^2 + 2n_i^2 d + 3n_i d m) \quad (2)$$

where d is the hidden-state dimensionality, m is the intermediate size of the feed-forward network.

References

- [1] Daniel Bolya, Cheng-Yang Fu, Xiaoliang Dai, Peizhao Zhang, Christoph Feichtenhofer, and Judy Hoffman. Token merging: Your vit but faster. *arXiv preprint arXiv:2210.09461*, 2022.
- [2] Liang Chen, Haozhe Zhao, Tianyu Liu, Shuai Bai, Junyang Lin, Chang Zhou, and Baobao Chang. An image is worth 1/2 tokens after layer 2: Plug-and-play inference acceleration for large vision-language models. In *European Conference on Computer Vision*, pages 19–35. Springer, 2024.
- [3] Yuan Zhang, Chun-Kai Fan, Junpeng Ma, Wenzhao Zheng, Tao Huang, Kuan Cheng, Denis Gudovskiy, Tomoyuki Okuno, Yohei Nakata, Kurt Keutzer, et al. Sparsevlm: Visual token sparsification for efficient vision-language model inference. In *International Conference on Machine Learning*, 2025.
- [4] Yuzhang Shang, Mu Cai, Bingxin Xu, Yong Jae Lee, and Yan Yan. Llava-prumerge: Adaptive token reduction for efficient large multimodal models. *arXiv preprint arXiv:2403.15388*, 2024.
- [5] Senqiao Yang, Yukang Chen, Zhuotao Tian, Chengyao Wang, Jingyao Li, Bei Yu, and Jiaya Jia. Visionzip: Longer is better but not necessary in vision language models. *arXiv preprint arXiv:2412.04467*, 2024.
- [6] Qizhe Zhang, Aosong Cheng, Ming Lu, Zhiyong Zhuo, MinQi Wang, Jiajun Cao, Shaobo Guo, Qi She, and Shanghang Zhang. [cls] attention is all you need for training-free visual token pruning: Make vlm inference faster. *arXiv preprint arXiv:2412.01818*, 2024.
- [7] Drew A Hudson and Christopher D Manning. Gqa: A new dataset for real-world visual reasoning and compositional question answering. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 6700–6709, 2019.
- [8] Tanik Saikh, Tirthankar Ghosal, Amish Mittal, Asif Ekbal, and Pushpak Bhattacharyya. Scienceqa: A novel resource for question answering on scholarly articles. *International Journal on Digital Libraries*, 23(3):289–301, 2022.
- [9] Amanpreet Singh, Vivek Natarajan, Meet Shah, Yu Jiang, Xinlei Chen, Dhruv Batra, Devi Parikh, and Marcus Rohrbach. Towards vqa models that can read. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 8317–8326, 2019.
- [10] Yifan Li, Yifan Du, Kun Zhou, Jinpeng Wang, Wayne Xin Zhao, and Ji-Rong Wen. Evaluating object hallucination in large vision-language models. *arXiv preprint arXiv:2305.10355*, 2023.
- [11] Chaoyou Fu, Peixian Chen, Yunhang Shen, Yulei Qin, Mengdan Zhang, Xu Lin, Jinrui Yang, Xiawu Zheng, Ke Li, Xing Sun, Yunsheng Wu, and Rongrong Ji. Mme: A comprehensive evaluation benchmark for multimodal large language models, 2024.

- 87 [12] Yuan Liu, Haodong Duan, Yuanhan Zhang, Bo Li, Songyang Zhang, Wangbo Zhao, Yike Yuan,
88 Jiaqi Wang, Conghui He, Ziwei Liu, Kai Chen, and Dahua Lin. Mmbench: Is your multi-modal
89 model an all-around player?, 2024.
- 90 [13] Zhihang Lin, Mingbao Lin, Luxi Lin, and Rongrong Ji. Boosting multimodal large language
91 models with visual tokens withdrawal for rapid inference. In *Proceedings of the AAAI Confer-*
92 *ence on Artificial Intelligence*, volume 39, pages 5334–5342, 2025.
- 93 [14] Cheng Yang, Yang Sui, Jinqi Xiao, Lingyi Huang, Yu Gong, Chendi Li, Jinghua Yan, Yu Bai,
94 Ponnuswamy Sadayappan, Xia Hu, et al. Topv: Compatible token pruning with inference
95 time optimization for fast and low-memory multimodal vision language model. *arXiv preprint*
96 *arXiv:2503.18278*, 2025.
- 97 [15] Yuliang Liu, Zhang Li, Mingxin Huang, Biao Yang, Wenwen Yu, Chunyuan Li, Xu-Cheng Yin,
98 Cheng-Lin Liu, Lianwen Jin, and Xiang Bai. Ocrbench: on the hidden mystery of ocr in large
99 multimodal models. *Science China Information Sciences*, 67(12):220102, 2024.
- 100 [16] Xiang Chen et al. Pyramiddrop: Accelerating your large vision-language models via pyramid
101 visual redundancy reduction. *arXiv preprint arXiv:2410.17247*, 2024.