MWOP: MODALITY-AWARE WIDTH-WISE OPERATION PRUNING FOR EFFICIENT MLLMS

Anonymous authors
Paper under double-blind review

ABSTRACT

Multimodal large language models (MLLMs) incur substantial inference costs when processing long sequences of visual and textual tokens. Existing operation compression methods exploit modality differences, but fine-grained execution selection within attention heads and shared feed-forward network (FFN) channels remains underexplored. Our analysis reveals that redundancy varies across modality-interaction paths within the same attention head and across modality-specific executions of the same FFN channel. Building on these findings, we propose *Modality-aware Width-wise Operation Pruning (MWOP)*. MWOP combines head-wise path-level pruning of visual-to-visual (V2V), text-to-visual (T2V), and text-to-text (T2T) attention paths with modality-conditioned pruning of FFN channel executions. A common first-order Taylor criterion estimates operation importance to guide sequential pruning of V2V, T2V, and T2T. FFN importance is then re-evaluated under the pruned attention configuration, followed by modality-conditioned channel selection and LoRA-based recovery training. MWOP preserves the token sequence while reducing attention and FFN computation. When combined with token compression, it reduces both sequence length and computation on the remaining tokens. Extensive experiments show that MWOP achieves a $1.6\times$ prefill speedup over the dense model with negligible performance loss on LLaVA-OneVision-7B. Combining MWOP with two representative token compression methods increases their prefill speedups from $2.0\times$ and $1.9\times$ to $2.9\times$ and $2.7\times$, while retaining 96.5% and 97.6% performance, respectively. Results on Qwen2.5-VL-7B further support MWOP's applicability across architectures.

1 INTRODUCTION

Multimodal large language models (MLLMs) process hundreds or thousands of visual tokens alongside textual instructions through a shared Transformer decoder (Li et al., 2024a; Bai et al., 2025). Propagating these long sequences through self-attention and feed-forward networks (FFNs) across decoder layers incurs substantial inference overhead. Redundancy exists in both the token sequence and the computation applied to it, creating two complementary opportunities for compression (Wu et al., 2026b; Liu et al., 2026). *Token compression* prunes or merges visual tokens to reduce the sequence length processed by subsequent layers (Zhang et al., 2025; Yang et al., 2025). *Operation compression* instead reduces the computation performed on retained tokens by selectively skipping or sparsifying attention and FFN operations (Liu et al., 2025; Li et al., 2025; Zhang et al., 2024a).

Operation compression reduces LLM computation along the depth and width dimensions (Wu et al., 2026b), as illustrated in Fig. 1 (a–c). Depth compression methods such as ShortV (Yuan et al., 2025) skip attention and FFN updates for visual tokens in selected LLM layers. Width compression methods such as RedundancyLens (Li et al., 2025) reduce visual-side computation through local self-attention and dynamic FFN channel selection. Building on these modality-aware designs, we focus on *finer-grained width pruning* within LLM layers. Specifically, we ask *which modality-interaction paths within each attention head and which modality-conditioned executions of each shared FFN channel should be retained*.

Under causal attention with visual tokens preceding textual tokens, there are three modality-interaction paths: visual-to-visual (V2V), text-to-visual (T2V), and text-to-text (T2T). These paths coexist within each attention head, but their redundancy can differ. Our analysis shows that V2V

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

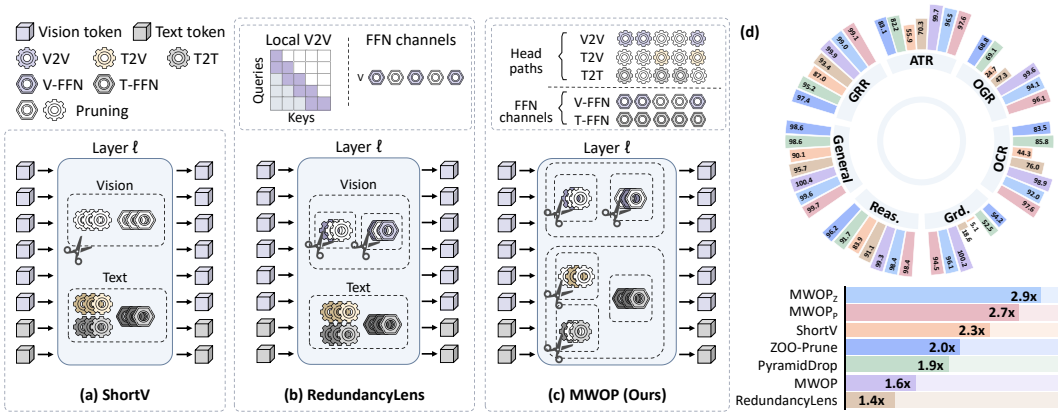


Figure 1: **Comparison of operation compression methods.** (a) ShortV skips visual token updates in selected layers. (b) RedundancyLens sparsifies visual-side attention and FFN computation. (c) MWOP performs finer-grained width compression by independently pruning V2V, T2V, and T2T attention paths and modality-conditioned FFN channel executions while preserving the token sequence. (d) MWOP achieves a 1.6× prefill speedup with negligible performance loss on LLaVA-OneVision-7B. Combining it with token compression further increases the speedup to 2.9×, with 96.5% performance retention.

and T2V exhibit greater redundancy than T2T, and that a head with low importance along one path may remain important along another (Section 4). Whole-head pruning therefore couples distinct path-specific retention decisions and can remove useful interactions along with redundant ones. This motivates *head-wise path-level attention pruning*, which selectively removes redundant paths within each head while preserving its useful interactions.

A similar granularity issue arises in FFNs, where visual and textual tokens share the same channel parameters. Our analysis shows that performance is more sensitive to pruning channel executions at textual positions than at visual positions. At matched channel-execution pruning ratios, selecting channels separately for the two modalities preserves more performance than applying a global shared mask, with the advantage growing as compression increases (Section 5). These findings suggest that parameter sharing does not imply identical redundancy patterns across modalities. This motivates *modality-conditioned FFN channel pruning*, which treats a shared channel's execution on each modality as a separate compression unit.

Building on these insights, we propose *Modality-aware Width-wise Operation Pruning (MWOP)*, which combines head-wise path-level attention pruning with modality-conditioned FFN channel pruning. MWOP uses a common first-order Taylor criterion (Molchanov et al., 2019) to assess operation importance in both modules. It sequentially compresses V2V, T2V, and T2T, corresponding to visual aggregation, cross-modal transfer, and textual propagation, respectively. It then re-evaluates FFN redundancy under the resulting attention configuration and selects channel executions separately for each modality. This staged procedure allows later decisions to account for earlier compression. Subsequent LoRA-based recovery training mitigates the performance loss from joint pruning. MWOP preserves the token sequence while reducing attention and FFN computation. It therefore complements token compression, which shortens the sequence processed by the LLM. Combining the two reduces both sequence length and computation on the remaining tokens, enabling further prefill acceleration.

Experiments on LLaVA-OneVision-7B show that MWOP achieves a 1.6× prefill speedup over the dense model with 99.7% average performance retention across 12 benchmarks. Combining MWOP with the token compression methods ZOO-Prune and PyramidDrop increases their prefill speedups from 2.0× to 2.9× and from 1.9× to 2.7×, with 96.5% and 97.6% performance retention, respectively. Our contributions are threefold: (1) We show that redundancy differs across paths within the same attention head and across modalities within the same FFN channel, motivating finer-grained pruning decisions. (2) We propose MWOP, a width-pruning framework that integrates head-wise path-level attention pruning and modality-conditioned FFN channel pruning through Taylor-guided staged selection and recovery training. (3) We validate MWOP on LLaVA-OneVision-7B and Qwen2.5-VL-7B, demonstrating its ability to reduce computation with negligible performance loss and achieve further acceleration when combined with token compression.

2 Related Work

Visual *token compression* methods can be categorized by the stage at which token reduction occurs: (1) Encoder-level methods prune or merge patch tokens during visual encoding (Bolya et al., 2022; Wu et al., 2026c). (2) Projector-level methods reduce visual token redundancy between the visual encoder and the LLM. PruMerge (Shang et al., 2025) and VisionZip (Yang et al., 2025) shorten visual token sequences by merging highly similar tokens. (3) LLM-level methods compress visual tokens within the LLM. FastV (Chen et al., 2024a) and PyramidDrop (Xing et al., 2024) prune visual tokens using importance scores derived from text-to-visual attention.

Operation compression skips or sparsifies attention and FFN operations while preserving the token sequence. Methods are broadly categorized into depth and width compression based on the targeted computational units. Depth compression skips visual token processing in selected LLM layers. ShortV (Yuan et al., 2025) freezes visual token attention and FFN updates in selected layers, while ViCA (Liu et al., 2026) removes these updates throughout the LLM and retains T2V interactions only at selected layers. HiDrop (Wu et al., 2026a) combines late injection and early exit with progressive token pruning. Width compression targets computation within layers. RedundancyLens (Li et al., 2025) uses local V2V attention and dynamic FFN channel selection for visual tokens. In contrast, MWOP performs finer-grained width compression through head-wise path-level attention pruning and modality-conditioned selection of FFN channel executions, while remaining compatible with token compression.

3 MWOP: Formulation and Overview

3.1 Modality-Specific Paths in Self-Attention

Excluding system tokens from the formulation, we consider visual and user-text tokens in a unified sequence. At layer l, let $V^{(l)}$ and $T^{(l)}$ denote their respective hidden states. For head h, the output $O^{(l,h)}$ before output projection takes the blockwise form

$$O^{(l,h)} = \begin{bmatrix} A_{VV}^{(l,h)} & A_{VT}^{(l,h)} \\ A_{TV}^{(l,h)} & A_{TT}^{(l,h)} \end{bmatrix} \begin{bmatrix} V^{(l)} W_V^{(l,h)} \\ T^{(l)} W_V^{(l,h)} \end{bmatrix}$$

where $W_V^{(l,h)}$ is the value projection shared across modalities, and the first and second subscripts of each attention block indicate query and key/value modalities, respectively. Under causal attention with visual tokens preceding user-text tokens, $A_{VT}^{(l,h)} = 0$, leaving three paths: V2V, T2V, and T2T. These paths support visual aggregation, cross-modal transfer, and textual propagation, respectively. Together with the layer-wise importance patterns in Section 4 and Appendix A.2, this functional progression motivates the staged pruning order V2V $\rightarrow$ T2V $\rightarrow$ T2T. For path-level masking analysis, we independently gate each path's additive contribution. Although T2V and T2T share softmax normalization, we zero either contribution without renormalizing the retained attention weights. This enables evaluation of each path's impact, treating each head–path pair as a pruning unit.

3.2 Modality-Conditioned Computation in the FFN

At layer l, let $x^{(l)}$ denote a token state, $\Phi^{(l)}(x^{(l)}) \in \mathbb{R}^{d_{\text{ffn}}^{(l)}}$ its intermediate FFN activation, and $W_{\text{out}}^{(l)}$ the output projection. Here, $d_{\text{ffn}}^{(l)}$ is the intermediate width, and $\Phi^{(l)}$ encompasses the input projections and channel-wise nonlinearities of standard or gated FFNs. For modality $\mu \in \{V, T\}$, let $\mathcal{I}_\mu^{(l)} \subseteq \{1, \ldots, d_{\text{ffn}}^{(l)}\}$ index the retained channels. Omitting bias terms for brevity, the pruned FFN is evaluated as

$$\widehat{\text{FFN}}_\mu^{(l)}(x^{(l)}) = \Phi_{\mathcal{I}_\mu^{(l)}}^{(l)}(x^{(l)})\, W_{\text{out}}^{(l)}[\mathcal{I}_\mu^{(l)}, :]$$

where $\Phi_{\mathcal{I}_\mu^{(l)}}^{(l)}$ computes only the retained channels using the corresponding parameter slices, and $[\mathcal{I}_\mu^{(l)}, :]$ selects the matching output-projection rows. Since FFNs process attention-updated token states, we re-evaluate channel importance after attention pruning. Parameters remain shared across modalities, but $\mathcal{I}_V^{(l)}$ and $\mathcal{I}_T^{(l)}$ may differ. We therefore treat the *visual-side* and *text-side* executions of each shared FFN channel as separate pruning units.

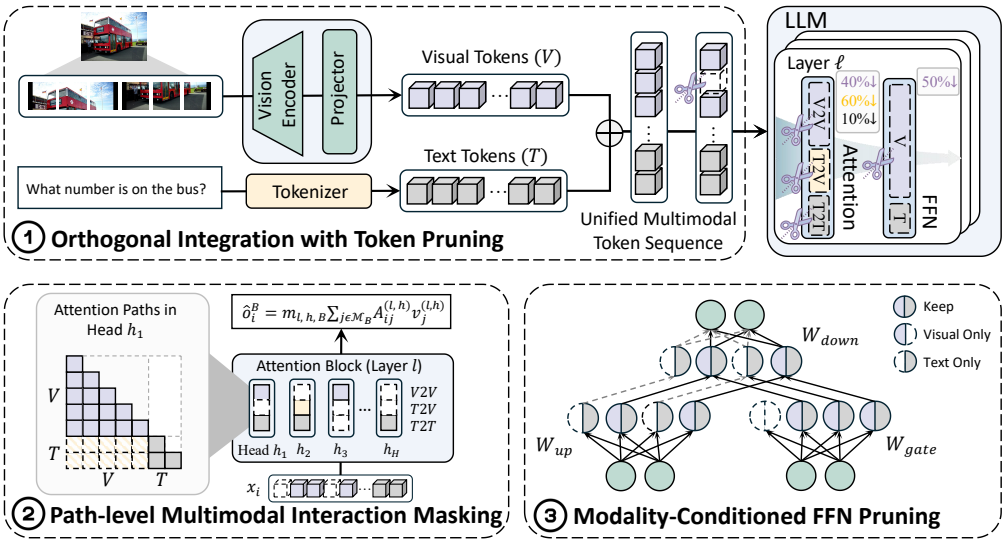


Figure 2: **Overview of MWOP.** (1) MWOP performs width pruning while preserving the token sequence and remaining compatible with token pruning. (2) V2V, T2V, and T2T paths are masked independently within each attention head. (3) Shared FFN channels are selectively executed for visual tokens, textual tokens, or both.

3.3 Overview of MWOP

As illustrated in Figure 2, MWOP performs head-wise path-level attention pruning in the order V2V $\rightarrow$ T2V $\rightarrow$ T2T, guided by first-order Taylor importance. With the attention masks fixed, it recomputes FFN channel importance using the same Taylor criterion and selects channels separately for visual and textual tokens. LoRA-based post-training recovery then mitigates the performance loss from pruning. MWOP preserves the token sequence and is therefore orthogonal to token pruning. Appendices A.1 and B.1 detail Taylor importance estimation and present comparisons with other importance criteria. Algorithm 1 in the appendix summarizes staged pruning and configuration selection. Following this staged procedure, we first analyze attention-path redundancy in Section 4 and then modality-conditioned FFN redundancy under the selected attention mask in Section 5.

4 Path-Wise Redundancy in Self-Attention

4.1 Redundancy Is Path-Conditional and Task-Dependent

We conduct training-free analyses on LLaVA-OneVision-7B using representative General and Grounding benchmarks (Section 6.1). Within each task family, we rank heads separately for each path by aggregated first-order Taylor importance. Each bottom-50% set contains 392 of 784 heads, yet only 126 heads on General and 141 on Grounding are shared across all three sets (Figure 3(a)). Low importance along one path therefore does not imply low importance along others, motivating separate pruning decisions within each head. We then evaluate single-path masking on 1,000

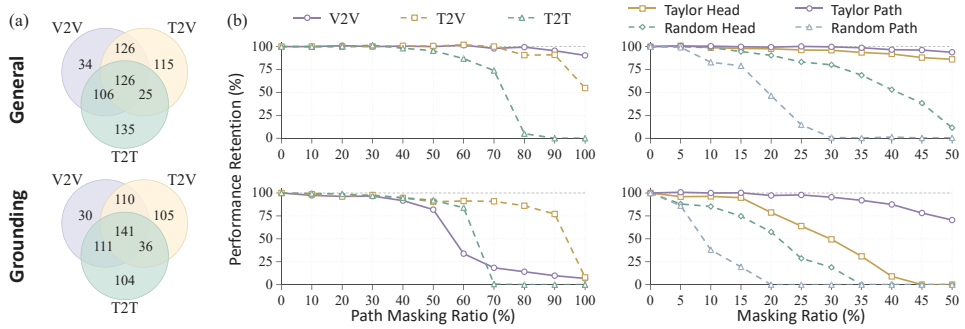


Figure 3: **Path-wise attention redundancy.** Rows show General (top) and Grounding (bottom). (a) Overlap of the bottom-50% Taylor-ranked head sets across paths. (b) Single-path masking with the other paths kept dense (left) and whole-head versus head-wise path-level masking using Taylor or random selection at matched numbers of masked path blocks (right). All analyses are training-free, with performance retention measured relative to the original dense LLaVA-OneVision-7B model.

Table 1: **Attention-path pruning with post-training recovery.** Setting indicates the path whose masking ratio is varied at each stage. Scores are normalized to the dense reference. ATR reports overall retention. GRR covers General and Reasoning, while OGR covers OCR and Grounding. Superscript R denotes uniformly random block selection for the marked path, while unmarked paths use Taylor selection. Taylor and random counterparts share the same masking ratios and recovery protocol. **Bold** and underlined values indicate the **best** and second-best results within each stage, respectively. Gray shading marks the selected configurations. Per-benchmark results are provided in Appendix C.1.

Setting	Mask Ratio (%)			Aggregate (%)			Category (%)			
	V2V	T2V	T2T	ATR	GRR	OGR	Gen.	Reas.	OCR	Grd.
Dense	0	0	0	100.0	100.0	100.0	100.0	100.0	100.0	100.0
V2V	30	0	0	**100.1**	100.1	**100.1**	100.3	99.9	**100.1**	**100.1**
	40	0	0	100.0	**100.3**	99.7	**100.6**	**100.1**	99.4	99.9
	40^{R}	0	0	96.0	98.7	93.4	99.8	97.6	95.3	91.5
	50	0	0	99.7	**100.3**	99.2	100.5	100.0	98.9	99.5
	60	0	0	98.8	99.8	97.8	100.2	99.5	97.1	98.5
T2V	40	50	0	**100.4**	100.0	**100.8**	100.1	99.9	**100.4**	**101.2**
	40	60	0	100.3	**100.1**	100.6	**100.3**	**100.0**	99.9	**101.2**
	40	60^{R}	0	95.0	95.6	94.5	97.7	93.4	96.0	92.9
	40^{R}	60^{R}	0	91.3	92.6	90.0	97.6	87.5	96.0	83.9
	40	70	0	99.5	99.6	99.4	99.5	99.7	99.0	99.8
	40	80	0	98.5	98.7	98.3	98.5	98.9	97.4	99.2
T2T	40	60	10	**100.0**	**99.9**	100.1	**100.3**	99.5	99.9	100.3
	40	60	10^{R}	99.7	99.0	100.3	99.3	98.7	100.0	100.5
	40^{R}	60^{R}	10^{R}	89.4	92.7	86.1	96.0	89.3	90.5	81.8
	40	60	20	99.9	**99.9**	99.9	99.9	**99.8**	**100.1**	99.6
	40	60	30	99.7	99.1	100.2	99.9	98.4	99.4	101.1
	40	60	40	99.4	98.5	100.4	99.8	97.1	99.3	**101.4**

examples from these benchmarks, keeping the other paths dense (Figure 3(b), left). General tolerates substantial V2V and T2V masking, whereas Grounding degrades sharply beyond 40% V2V masking but retains over 90% of baseline performance at 70% T2V masking. T2T masking causes performance collapse near 80% on General and 70% on Grounding. These differences motivate path-specific budgets evaluated across task families, with more conservative V2V pruning to preserve Grounding performance. Analyses for all four task families are provided in Appendix A.3.

4.2 Granularity and Importance-Guided Selection Both Matter

At masking ratio q, whole-head masking removes a fraction q of heads, while head-wise path-level masking independently removes the same fraction from each path, matching masked path-block counts up to rounding (Figure 3(b), right). At 30% masking, Taylor-guided path-level masking retains 95.3% performance versus 49.3% for whole-head masking on Grounding, and 99.6% versus 96.1% on General. Taylor also outperforms random selection at both granularities at this ratio, supporting the combined benefits of finer pruning granularity and importance-guided selection.

4.3 Recovering a General-Purpose Path Mask

The training-free analysis motivates asymmetric budgets, particularly more conservative V2V than T2V masking for Grounding. We evaluate candidates sequentially for V2V, T2V, and T2T after LoRA-based recovery, fixing earlier masks and keeping later paths dense. Hereafter, dense references are post-trained on the same data unless otherwise specified. We adopt $(40\%, 60\%, 10\%)$, achieving 100.0% ATR and 99.5%–100.3% retention across task families (Table 1). Raising T2T masking to 40% lowers Reasoning retention to 97.1%, despite 99.4% ATR. Randomizing all masks at these ratios yields 89.4% ATR under identical recovery, showing that Taylor's benefit persists.

5 Modality-Aware Redundancy in the FFN

5.1 FFN Redundancy Is Modality-Conditioned

Under the $(\text{V2V}, \text{T2V}, \text{T2T}) = (40\%, 60\%, 10\%)$ attention mask from Section 4, we recompute first-order Taylor importance separately for visual- and text-side FFN channel executions, termed *visual neurons* and *text neurons*, respectively. At masking ratio r, Global masks the same bottom-r channels for both modalities, whereas Vision+Text selects separate sets at matched per-modality ratios. Vision+Text retains more performance on General and Grounding, with a widening advantage at higher ratios (Figure 4(b), left). Thus, parameter sharing does not imply identical redun-

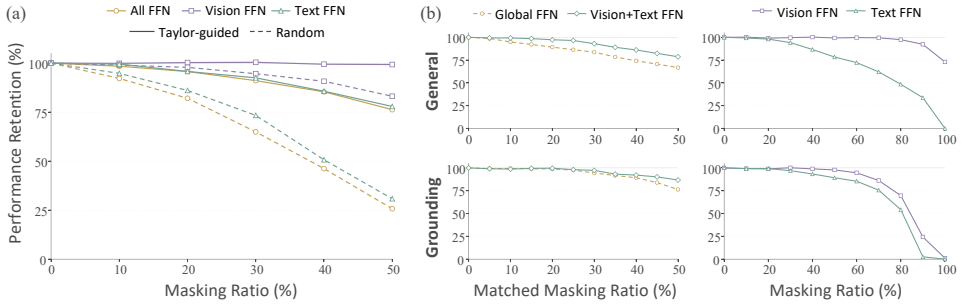


Figure 4: **Modality-conditioned FFN redundancy.** (a) Taylor (solid) vs. random (dashed) selection for All, Vision, and Text masking, with retention averaged equally across four task families. (b) Shared (Global) vs. modality-specific (Vision+Text) channel masking at matched per-modality ratios (left) and sensitivity to visual-only vs. text-only masking (right). All analyses are training-free, with performance measured relative to the attention-pruned model $\mathrm{MWOP}_{\mathrm{Attn}}$ with dense FFNs.

dancy across modalities. Training-free single-modality masking reveals asymmetric sensitivity (Figure 4(b), right). Visual-side masking preserves General performance over a broad range, whereas text-side masking degrades earlier on both task families. Grounding is less tolerant than General of aggressive visual-side masking. These findings motivate modality-conditioned channel selection and asymmetric FFN pruning budgets. Analyses for all four task families appear in Appendix B.3.

5.2 Granularity and Importance-Guided Selection Both Matter

We compare Taylor and random selection at matched ratios within each FFN masking setting (Figure 4(a)). Taylor outperforms random selection at every tested nonzero ratio. At 50% masking, the gains are 50.5%, 16.2%, and 47.1% for All, Vision, and Text masking, respectively. Visual-only masking remains nearly lossless with Taylor, compared with 83.1% retention under random selection. Visual-only masking also retains more performance than text-only masking under random selection, indicating that modality asymmetry persists without Taylor ranking. Together with the shared-mask comparison in Section 5.1, these results support both modality-conditioned pruning granularity and importance-guided selection.

5.3 Recovering the Joint Attention–FFN Operating Point

Guided by the training-free analysis, we evaluate FFN masking ratios after LoRA-based post-training recovery, keeping the attention mask fixed. With text channels dense, we select 50% visual masking, retaining 99.7% ATR and 98.9%–100.4% across task families (Table 2). Increasing visual masking to 60% reduces OCR and Grounding retention, while 70% lowers General retention to 94.4%. A separate diagnostic comparison fixes visual masking at 40% and varies text masking. Adding 10% text masking reduces ATR from 100.0% to 99.5%, with larger ratios further degrading General and Reasoning. Random selection at 50% visual masking yields 94.9% ATR under identical recovery, showing that importance-guided selection remains beneficial after post-training.

Table 2: **Modality-conditioned FFN pruning with post-training recovery.** Visual and Text denote FFN masking ratios. Scores are normalized to the dense reference. Superscript R denotes uniformly random channel selection for the marked modality, while unmarked masks use Taylor selection. Taylor and random counterparts share the same masking ratios and recovery protocol. Per-benchmark results are provided in Appendix C.1.

Setting	Mask Ratio (%)		Aggregate (%)			Category (%)			
	Visual	Text	ATR	GRR	OGR	Gen.	Reas.	OCR	Grd.
$\mathrm{MWOP}_{\mathrm{Attn}}$	0	0	100.0	99.9	100.1	100.3	99.5	99.9	100.3
Visual Neuron	40	0	**100.0**	**99.9**	**100.0**	100.1	**99.7**	**99.7**	**100.3**
	40^{R}	0	96.4	97.3	95.6	99.0	95.6	94.3	96.9
	50	0	99.7	**99.9**	99.6	**100.4**	99.3	98.9	100.2
	50^{R}	0	94.9	96.3	93.6	98.5	94.2	92.7	94.4
	60	0	99.3	99.7	98.9	100.0	99.4	98.2	99.6
	70	0	97.0	96.3	97.8	94.4	98.2	97.2	98.4
Text Neuron	40	10	**99.5**	**99.4**	**99.6**	**99.8**	**99.0**	**99.0**	**100.2**
	40	10^{R}	98.1	97.3	98.9	97.6	97.1	97.6	100.1
	40^{R}	10^{R}	94.2	94.5	93.9	96.4	92.7	93.6	94.1
	40	20	98.4	97.4	99.4	98.3	96.5	98.8	99.9
	40	30	97.0	95.0	99.1	95.9	94.0	98.2	99.9

Table 3: **Comparison with operation compression methods on LLaVA-OneVision-7B at approximately 60% retained LLM-side FLOPs.** Dep., Wid., and Tok. denote depth, width, and token compression; some compared methods also compress tokens. ATR, GRR, and OGR measure performance retention relative to the dense model. Unshaded rows apply compression after post-training; red and blue rows use post-training recovery with compression applied, with blue denoting ours. † marks methods originally proposed without post-compression recovery but evaluated with it here. MWOP achieves the best performance with the fewest FLOPs among compared methods, retaining 99.7% overall (ATR) and 99.6% fine-grained performance (OGR).

Method	Sparsity			Ret.	General			Reasoning			OCR			Grounding			ATR	GRR	OGR
	Dep.	Wid.	Tok.	(%)	GQA	VQA^{v2}	VQA^{ok}	SQA	AI2D	MMStar	VQA^{T}	VQA^{D}	OCRB	Ref	Ref+	RefG	(%)	(%)	(%)
LLaVA-OneVision-7B	-	-	-	100.0	63.1	82.2	59.9	96.0	81.9	61.4	75.3	87.4	63.5	77.9	70.5	76.9	100.0	100.0	100.0
ShortV	✓			65.6	62.5	81.0	58.9	94.7	79.8	60.1	63.6	70.7	49.5	46.3	42.1	42.8	84.0	98.3	69.7
RedundancyLens		✓		58.4	62.5	81.3	58.9	92.9	80.2	58.1	71.1	81.9	59.9	62.6	54.8	60.8	92.1	97.6	86.6
HalfV	✓		✓	60.4	62.6	81.4	59.7	95.6	81.4	59.9	73.1	85.1	55.0	52.7	46.4	47.8	89.2	99.1	79.4
DOP$_V$	✓	✓	✓	58.3	62.6	80.7	58.7	95.3	81.4	61.2	67.0	73.2	54.1	54.8	46.9	52.2	88.0	98.9	77.1
ShortV†	✓			65.6	63.2	81.8	60.0	95.9	80.9	60.5	72.1	82.8	59.6	76.3	68.7	74.7	97.8	99.5	96.1
RedundancyLens†		✓		58.4	63.3	81.6	59.8	93.8	81.3	59.2	73.5	84.8	62.6	75.5	67.7	74.7	98.0	98.8	97.2
YOPO	✓	✓		70.0	61.2	77.9	57.1	85.2	73.2	50.1	40.0	29.0	24.4	34.2	27.2	28.9	66.0	91.1	40.8
HalfV†	✓		✓	60.4	63.1	81.5	59.7	95.5	**81.7**	**60.8**	72.3	82.1	54.0	50.4	42.3	43.6	87.8	99.5	76.0
Skip-Vision		✓	✓	59.3	63.1	81.5	**60.7**	95.3	81.3	60.7	**74.0**	84.5	58.7	66.6	58.0	63.3	94.6	99.6	89.6
DOP$_V$†	✓	✓	✓	58.3	63.1	81.5	59.9	94.9	81.3	59.9	72.1	82.6	59.1	74.4	66.9	72.8	96.9	99.1	94.7
MWOP (ours)		✓		**56.8**	**63.4**	**82.0**	60.5	**96.2**	81.3	60.5	73.8	**85.0**	**64.5**	**77.9**	**70.7**	**77.1**	**99.7**	**99.9**	**99.6**

6 EXPERIMENTS

6.1 EXPERIMENTAL SETTINGS

Models. We conduct the main experiments on LLaVA-OneVision-7B (Li et al., 2024a) and extend the evaluation to Qwen2.5-VL-7B (Bai et al., 2025) to assess the effectiveness of MWOP across different MLLM architectures. Architectural details are provided in Appendix D.1.

Benchmarks. We evaluate MWOP on 12 benchmarks spanning 4 capability categories. General includes GQA (Hudson & Manning, 2019), VQAv2 (Goyal et al., 2017), and OK-VQA (Marino et al., 2019); Reasoning includes ScienceQA-IMG (Lu et al., 2022), AI2D (Kembhavi et al., 2016), and MMStar (Chen et al., 2024b); OCR includes TextVQA (Singh et al., 2019), DocVQA (Mathew et al., 2021), and OCRBench (Liu et al., 2024); Grounding includes RefCOCO, RefCOCO+, and RefCOCOg (Kazemzadeh et al., 2014; Yu et al., 2016; Mao et al., 2016). See Appendix F for details.

Efficiency Evaluation. Following HiDrop (Wu et al., 2026a), we report theoretical LLM FLOPs and measured prefill latency. Let V_ℓ and T_ℓ count visual and subsequent text tokens at layer ℓ, excluding the system prefix, and $N_\ell = V_\ell + T_\ell$. Counting two FLOPs per multiply–accumulate,

$$\text{FLOPs} = \sum_{\ell=1}^{L} \Big[4N_\ell d(d+d_{\text{KV}}) + 4d\big(\rho_{\text{V2V},\ell} C(V_\ell) + \rho_{\text{T2V},\ell} T_\ell V_\ell + \rho_{\text{T2T},\ell} C(T_\ell)\big) + 6dm\big(\rho_{V,\ell} V_\ell + \rho_{T,\ell} T_\ell\big) \Big]$$

where d, d_{KV}, and m are the hidden, key/value projection, and original FFN intermediate dimensions. $C(x) = x(x+1)/2$ counts causal token pairs; ρ denotes retained attention-path and modality-specific FFN channel ratios. FLOPs use average evaluation token counts (V, T) of $(3215, 19)$ for LLaVA-OneVision and $(485, 38)$ for Qwen2.5-VL. Prefill latency is measured on an NVIDIA A100 40GB using a unified stack optimized for token split–merge, modality-specific FFNs, and attention-path skipping. Appendices D.4 and D.5 detail the derivation and inference implementation.

Implementation Details. We perform post-training recovery of the compressed model via one epoch of LoRA fine-tuning (rank 128, scaling factor 256, dropout 0.05) on a multimodal instruction mixture of 282K samples, detailed in Appendix D.6. Training runs on 8 NVIDIA A100 40GB GPUs with a global batch size of 256, a learning rate of 1×10^{-5}, 3% warmup, and cosine decay. For a fair comparison, all methods evaluated with post-training recovery share the same recovery dataset. Method and training configurations are detailed in Appendices D.2 and D.7, respectively.

6.2 MAIN RESULT

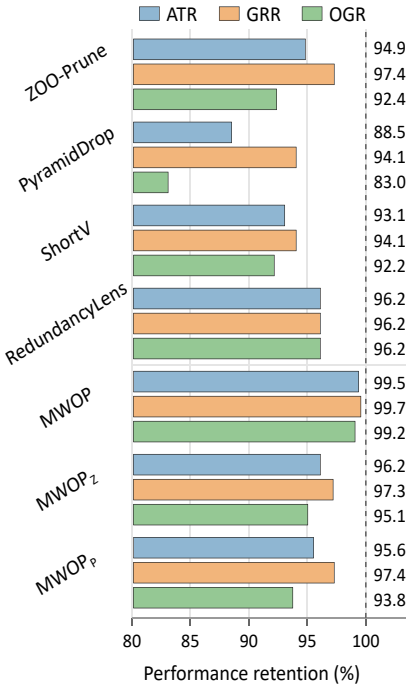
Comparison with Operation Compression Methods. Table 3 compares MWOP with operation compression methods at approximately 60% retained LLM-side FLOPs. MWOP achieves the highest ATR, GRR, and OGR of 99.7%, 99.9%, and 99.6%, respectively, while retaining the fewest FLOPs (56.8%). Its advantage is particularly pronounced in fine-grained tasks: MWOP improves OGR by 2.4 points over RedundancyLens with recovery, the strongest baseline, with corresponding ATR and GRR gains of 1.7 and 1.1 points. These results show that MWOP maintains near-dense performance across general visual understanding, reasoning, OCR, and grounding.

Table 4: **Comparison of token and operation compression methods on LLaVA-OneVision-7B at approximately 30% retained LLM-side FLOPs.** MWOP_P and MWOP_Z combine MWOP with PyramidDrop and ZOO-Prune. † denotes originally training-free methods evaluated with post-training recovery, while ‡ denotes originally training-based methods evaluated without post-training recovery. By composing operation and token compression, MWOP_Z and MWOP_P outperform all baselines overall while using fewer FLOPs.

Method	Sparsity			Ret.	General			Reasoning			OCR			Grounding			ATR	GRR	OGR
	Dep.	Wid.	Tok.	(%)	GQA	VQA^{v2}	VQA^{ok}	SQA	AI2D	MMStar	VQA^{T}	VQA^{D}	OCRB	Ref	Ref+	RefG	(%)	(%)	(%)
LLaVA-OneVision-7B	-	-	-	100.0	63.1	82.2	59.9	96.0	81.9	61.4	75.3	87.4	63.5	77.9	70.5	76.9	100.0	100.0	100.0
FastV			✓	32.8	59.9	77.6	58.6	89.7	77.0	54.6	61.1	60.4	44.6	47.5	39.5	46.5	80.1	93.9	66.3
PyramidDrop‡			✓	33.4	60.5	80.1	59.8	90.2	76.2	52.1	70.2	75.2	40.5	16.0	13.0	13.7	72.1	94.2	50.0
VisionZip			✓	32.7	61.8	80.3	59.8	91.5	76.8	55.8	69.7	55.2	50.1	8.5	7.4	9.4	70.3	95.9	44.7
DART			✓	32.8	62.2	80.9	59.3	92.0	78.0	57.1	71.7	75.6	54.9	22.3	19.4	20.8	77.6	96.7	58.6
VisPruner			✓	32.7	62.6	79.6	58.9	92.8	79.0	57.5	68.9	61.8	52.7	35.1	28.8	36.5	80.0	96.8	63.1
PruMerge+			✓	32.7	61.5	79.5	57.1	89.5	77.0	54.4	67.3	52.3	49.7	37.4	31.8	34.7	77.6	94.2	60.9
ZOO-Prune			✓	32.7	63.1	80.1	58.9	93.9	79.1	57.8	69.0	64.4	54.2	42.7	36.8	42.7	83.1	97.4	68.8
ShortV	✓			37.5	57.5	74.8	52.9	83.2	71.0	48.1	45.3	26.3	27.2	3.8	3.5	4.2	55.9	87.0	24.7
RedundancyLens		✓		32.0	60.5	78.6	57.3	88.7	76.2	53.9	60.6	58.2	51.4	15.3	12.6	14.2	70.3	93.4	47.3
HalfV	✓		✓	33.2	48.3	60.4	44.5	82.1	69.3	41.3	35.0	22.3	11.9	1.9	1.4	1.8	46.6	77.0	16.2
DOP_V	✓	✓	✓	32.0	63.3	79.6	58.8	94.2	80.2	60.0	65.5	75.6	51.3	46.3	39.2	45.7	84.8	98.1	71.5
FastV†			✓	32.8	61.2	78.6	59.4	90.6	78.9	55.1	63.7	64.6	45.5	59.0	49.8	56.6	85.2	95.4	75.0
PyramidDrop			✓	33.4	61.9	80.7	59.7	90.8	76.5	53.5	70.1	76.4	48.8	44.0	35.9	38.7	82.2	95.2	69.1
VisionZip†			✓	32.7	62.6	81.0	60.3	92.3	78.9	56.4	71.7	63.9	53.7	40.1	32.8	35.9	81.7	97.1	66.2
DART†			✓	32.8	62.7	81.1	60.0	93.0	79.3	58.0	72.5	79.8	58.0	31.9	27.5	29.6	82.0	97.7	66.2
VisPruner†			✓	32.7	62.9	81.1	59.6	94.0	80.3	58.2	72.2	69.0	58.1	57.0	49.3	54.8	89.1	98.1	80.1
PruMerge+†			✓	32.7	62.6	81.0	**60.5**	91.2	78.5	56.3	70.4	60.5	53.2	54.8	46.2	51.5	85.9	96.9	74.9
ZOO-Prune†			✓	32.7	**63.4**	81.4	60.1	94.8	80.3	58.3	72.0	72.2	58.6	64.0	55.6	60.7	91.8	98.6	85.1
ShortV†	✓			37.5	62.3	80.0	57.3	88.5	77.9	55.2	63.8	63.7	51.4	55.1	47.1	50.2	84.2	94.8	73.5
RedundancyLens†		✓		32.0	62.8	80.5	58.5	90.7	78.7	55.7	69.2	74.3	58.9	64.9	55.5	62.1	90.7	96.1	85.4
YOPO	✓	✓		34.7	58.9	74.1	53.5	84.0	71.9	48.6	22.6	17.2	14.2	29.3	21.4	23.3	58.1	87.8	28.4
HalfV†	✓		✓	33.2	55.9	72.0	55.3	85.6	71.9	47.1	45.9	34.8	26.7	22.8	16.2	18.5	61.8	87.0	36.5
Skip-Vision		✓	✓	32.0	63.0	80.9	59.3	94.6	81.1	60.1	66.1	74.9	54.7	62.9	55.2	59.9	90.8	98.8	82.7
DOP_V†	✓	✓	✓	32.0	63.3	81.4	60.0	94.7	80.6	**60.5**	71.2	82.7	57.9	65.7	57.4	64.1	93.7	**99.1**	88.2
MWOP_Z (ours)		✓	✓	27.9	62.9	81.6	59.8	**95.8**	**81.4**	58.9	71.9	73.4	61.4	**75.2**	**67.6**	**73.9**	96.5	99.0	94.1
MWOP_P (ours)		✓	✓	30.8	62.9	**81.7**	60.0	95.6	80.3	59.9	**73.6**	**83.8**	**63.0**	73.8	66.3	73.0	**97.6**	**99.1**	**96.1**

Table 5: **Efficiency and cross-model generalization. Left**: Theoretical FLOPs and practical prefill latency with breakdowns on two models. The component-removal configurations serve as references for potential speedup. Combining MWOP with token compression further reduces FLOPs and prefill latency on both architectures. **Right**: Cross-model generalization on Qwen2.5-VL-7B at approximately 40% retained LLM-side FLOPs, with standalone MWOP at 67.4% included as a reference. Composition improves overall and fine-grained retention over token compression alone. Per-benchmark results are reported in Appendix Table 13.

Method	Theoretical (TFLOPs)			Practical (Prefill Latency, ms)		
	Attn	FFN	Total	Attn	FFN	Total
LLaVA-OneVision-7B						
Dense	7.42	36.89	44.31	45.8	188.3	237.0
w/o Vis. Token	0.03	0.22	0.25	2.5 (18.1×)	9.7 (19.4×)	13.2 (18.0×)
w/o V2V&T2V	5.32	36.89	42.21	32.7 (1.4×)	188.3 (1.0×)	221.5 (1.1×)
w/o FFN_V	7.42	0.22	7.63	45.8 (1.0×)	10.8 (17.4×)	56.9 (4.2×)
ShortV	3.35	14.62	17.97	23.2 (2.0×)	80.5 (2.3×)	104.5 (2.3×)
RedundancyLens	6.60	22.22	28.82	42.9 (1.1×)	130.4 (1.4×)	174.8 (1.4×)
MWOP (ours)	6.57	18.60	25.17	42.5 (1.1×)	107.4 (1.8×)	152.1 (1.6×)
ZOO-Prune	3.20	18.55	21.75	21.3 (2.2×)	96.3 (2.0×)	119.0 (2.0×)
MWOP_Z (ours)	2.99	**9.38**	**12.37**	**20.4** (**2.2**×)	**59.5** (**3.2**×)	**81.2** (**2.9**×)
PyramidDrop	3.42	18.55	21.96	23.1 (2.0×)	100.1 (1.9×)	124.9 (1.9×)
MWOP_P (ours)	3.16	10.49	13.65	21.9 (2.1×)	64.8 (2.9×)	88.2 (2.7×)
Qwen2.5-VL-7B						
Dense	0.91	5.97	6.88	8.2	40.7	49.8
w/o Vis. Token	0.06	0.43	0.50	2.5 (3.2×)	10.0 (4.1×)	13.5 (3.7×)
w/o V2V&T2V	0.86	5.97	6.83	7.7 (1.1×)	38.0 (1.1×)	46.7 (1.1×)
w/o FFN_V	0.91	0.43	1.35	8.0 (1.0×)	10.4 (3.9×)	19.4 (2.6×)
ShortV	0.46	2.61	3.07	5.8 (1.4×)	22.3 (1.8×)	28.7 (1.7×)
RedundancyLens	0.91	3.75	4.66	8.8 (0.9×)	32.3 (1.3×)	41.8 (1.2×)
MWOP (ours)	0.88	3.76	4.64	7.8 (1.1×)	29.9 (1.4×)	38.8 (1.3×)
ZOO-Prune	0.48	3.19	3.67	5.2 (1.6×)	30.2 (1.3×)	36.5 (1.4×)
MWOP_Z (ours)	0.47	**2.09**	**2.56**	**4.4** (**1.9**×)	**19.7** (**2.1**×)	**25.1** (**2.0**×)
PyramidDrop	0.48	3.19	3.68	5.1 (1.6×)	25.0 (1.6×)	31.3 (1.6×)
MWOP_P (ours)	0.47	2.44	2.91	4.6 (1.8×)	23.5 (1.7×)	29.6 (1.7×)



Composition with Token Compression. Token and operation compression target complementary dimensions of LLM-side computation: token count and computation per retained token. Combining MWOP with PyramidDrop and ZOO-Prune yields MWOP_P and MWOP_Z. Table 4 compares these variants with token and operation compression methods at approximately 30% retained LLM-side FLOPs. MWOP_P and MWOP_Z rank first and second in ATR (97.6%, 96.5%), GRR (99.1%, 99.0%), and OGR (96.1%, 94.1%), while using fewer FLOPs (30.8%, 27.9%) than all baselines. Under the same recovery protocol, it improves ATR, GRR, and OGR over PyramidDrop by 15.4, 3.9, and 27.0 points, while MWOP_Z improves over ZOO-Prune by 4.7, 0.4, and 9.0 points. MWOP_P also surpasses DOP_V with recovery, the strongest baseline, by 3.9, 0.0, and 7.9 points, respectively. These results show that combining operation and token compression preserves more performance than token compression alone, particularly on fine-grained tasks, at a comparable budget.

Table 6: **Multi-image and video evaluation on LLaVA-OneVision-7B.** All methods are evaluated directly after post-training recovery on single-image data. Sub. denotes subtitles.

Settings	Multi-Image				Video				
	Avg (%)	Q-Bench2	Mantis-Eval	BLINK	Avg (%)	MVBench	Video-MME w/o Sub.	Video-MME w/ Sub.	NExT-QA
LLaVA-OneVision-7B	100.0	72.6	54.8	48.8	100.0	57.9	58.7	62.0	78.7
ZOO-Prune†	**100.8**	**75.0**	54.4	48.7	98.9	56.3	58.0	**61.4**	79.2
PyramidDrop	90.6	70.9	46.5	43.6	96.9	54.9	56.7	60.0	78.2
ShortV†	85.3	60.4	46.1	43.2	92.1	52.5	52.4	57.1	75.9
RedundancyLens†	85.1	62.6	43.8	43.5	91.2	52.6	51.2	55.9	76.1
MWOP	100.4	72.7	**55.3**	**48.9**	**99.2**	**56.8**	**58.1**	61.2	**79.6**
MWOP_P	97.9	73.2	53.5	46.5	93.6	53.0	53.9	57.9	77.0
MWOP_Z	96.9	72.8	52.5	46.2	93.9	52.9	54.6	58.0	76.9

Efficiency and Cross-Model Generalization. Table 5 (left) reports efficiency evaluation on two MLLMs. Removing V2V and T2V gives only a 1.1× prefill speedup on both models, while removing visual-side FFNs yields 4.2× on LLaVA-OneVision and 2.6× on Qwen2.5-VL, indicating greater acceleration potential in visual-side FFNs at the evaluated input lengths. On LLaVA-OneVision, MWOP reduces FLOPs by 43.2% with a 1.6× prefill speedup, mainly from 1.8× faster FFNs. Composing MWOP with PyramidDrop and ZOO-Prune raises speedups from 1.9×/2.0× to 2.7×/2.9×, respectively, with FLOP reductions of 69.2%/72.1%. On Qwen2.5-VL, MWOP reduces FLOPs by 32.6% with a 1.3× prefill speedup. Composition raises ZOO-Prune's speedup from 1.4× to 2.0× and PyramidDrop's from 1.6× to 1.7×. Shorter visual sequences may limit gains through lower arithmetic intensity and greater relative memory-access and fixed overheads (Appendix C.3). Table 5 (right) evaluates cross-model generalization on Qwen2.5-VL. As a reference, standalone MWOP retains 67.4% of LLM-side FLOPs and achieves 99.5% ATR, 99.7% GRR, and 99.2% OGR. At approximately 40% retained FLOPs, MWOP_Z and MWOP_P outperform their token-only counterparts by 1.3 and 7.1 percentage points in ATR, and 2.7 and 10.8 points in OGR, respectively. These results support MWOP's effectiveness across architectures and its complementary benefits to token compression, particularly on fine-grained tasks.

7 Discussion

Task-Specialized Path Masks. Starting from the shared configuration (40% V2V, 60% T2V, 10% T2T), we adjust path compression ratios using task-specific Taylor profiles (Figure 5). After post-training recovery, increasing V2V compression to 70% for General retains 100.1% performance, versus 100.3% with shared masks. For Reasoning, retaining all T2T paths raises retention from 99.5% to 100.0%. Increasing T2T compression to 20% for OCR and 50% for Grounding yields 100.1% and 100.3% retention, versus 99.9% and 100.3%, respectively. Task-specialized masks thus allow stronger compression on selected paths while preserving target-category performance.

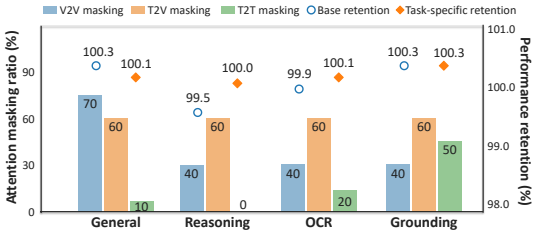


Figure 5: **Task-specialized attention-path compression.** Bars show path compression ratios, and markers show post-recovery performance retention.

Multi-Image and Video Understanding. Table 6 reports multi-image and video results after recovery solely on single-image data. Without additional multi-image or video fine-tuning, MWOP retains 100.4% and 99.2% of dense performance, respectively, achieving the highest video retention among the compared compression methods. These results demonstrate that MWOP preserves the dense model's capabilities across input formats. Incorporating multi-image or video data into recovery training may further improve performance.

8 Conclusion

We identify fine-grained redundancy across modality-interaction paths within attention heads and modality-specific executions of shared FFN channels. Building on these findings, MWOP combines head-wise path-level attention pruning and modality-conditioned FFN channel pruning through Taylor-guided selection and post-training recovery. Experiments on two MLLM architectures show that MWOP reduces computation while preserving near-dense performance, including on fine-grained OCR and grounding tasks. Its composition with token compression further accelerates prefill while maintaining high performance retention, demonstrating the complementary benefits of reducing token count and computation per retained token. Future work will explore learnable gating to jointly optimize pruning decisions and compression budgets.

AI USE STATEMENT

We used generative AI tools for translation, language editing, and figure preparation based on author-provided code and data. The authors manually reviewed and revised all AI-assisted content for accuracy and consistency with the study. We take full responsibility for the final manuscript, including its text, figures, scientific claims, and other materials prepared with the assistance of generative AI.

REFERENCES

Saeed Ranjbar Alvar, Gursimran Singh, Mohammad Akbari, and Yong Zhang. Divprune: Diversity-based visual token pruning for large multimodal models. In *2025 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 9392–9401. IEEE, 2025.

Shuai Bai, Keqin Chen, Xuejing Liu, Jialin Wang, Wenbin Ge, Sibo Song, Kai Dang, Peng Wang, Shijie Wang, Jun Tang, Humen Zhong, Yuanzhi Zhu, Mingkun Yang, Zhaohai Li, Jianqiang Wan, Pengfei Wang, Wei Ding, Zheren Fu, Yiheng Xu, Jiabo Ye, Xi Zhang, Tianbao Xie, Zesen Cheng, Hang Zhang, Zhibo Yang, Haiyang Xu, and Junyang Lin. Qwen2.5-vl technical report, 2025. URL https://arxiv.org/abs/2502.13923.

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.

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*, pp. 19–35. Springer, 2024a.

Lin Chen, Jinsong Li, Xiaoyi Dong, Pan Zhang, Yuhang Zang, Zehui Chen, Haodong Duan, Jiaqi Wang, Yu Qiao, Dahua Lin, et al. Are we on the right way for evaluating large vision-language models? *Advances in Neural Information Processing Systems*, 37:27056–27087, 2024b.

Wei-Lin Chiang, Zhuohan Li, Zi Lin, Ying Sheng, Zhanghao Wu, Hao Zhang, Lianmin Zheng, Siyuan Zhuang, Yonghao Zhuang, Joseph E. Gonzalez, Ion Stoica, and Eric P. Xing. Vicuna: An open-source chatbot impressing gpt-4 with 90%* chatgpt quality, March 2023. URL https://lmsys.org/blog/2023-03-30-vicuna/.

Sixun Dong, Juhua Hu, Mian Zhang, Ming Yin, Yanjie Fu, and Qi Qian. Mmtok: Multimodal coverage maximization for efficient inference of vlms. In *International Conference on Learning Representations*, volume 2026, pp. 48075–48099, 2026.

Mark Endo, Xiaohan Wang, and Serena Yeung-Levy. Feather the throttle: Revisiting visual token pruning for vision-language model acceleration. In *2025 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 22826–22835. IEEE, 2025.

Chaoyou Fu, Yuhan Dai, Yongdong Luo, Lei Li, Shuhuai Ren, Renrui Zhang, Zihan Wang, Chenyu Zhou, Yunhang Shen, Mengdan Zhang, et al. Video-mme: The first-ever comprehensive evaluation benchmark of multi-modal llms in video analysis. In *2025 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 24108–24118. IEEE, 2025.

Xingyu Fu, Yushi Hu, Bangzheng Li, Yu Feng, Haoyu Wang, Xudong Lin, Dan Roth, Noah A Smith, Wei-Chiu Ma, and Ranjay Krishna. Blink: Multimodal large language models can see but not perceive. In *European Conference on Computer Vision*, pp. 148–166. Springer, 2024.

Yash Goyal, Tejas Khot, Douglas Summers-Stay, Dhruv Batra, and Devi Parikh. Making the v in vqa matter: Elevating the role of image understanding in visual question answering. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 6904–6913, 2017.

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*, pp. 6700–6709, 2019.

Dongfu Jiang, Xuan He, Huaye Zeng, Cong Wei, Max Ku, Qian Liu, and Wenhu Chen. Mantis: Interleaved multi-image instruction tuning. *arXiv preprint arXiv:2405.01483*, 2024.

Kushal Kafle, Brian Price, Scott Cohen, and Christopher Kanan. Dvqa: Understanding data visualizations via question answering. In *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 5648–5656. IEEE, 2018.

Sahar Kazemzadeh, Vicente Ordonez, Mark Matten, and Tamara Berg. Referitgame: Referring to objects in photographs of natural scenes. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pp. 787–798, 2014.

Aniruddha Kembhavi, Mike Salvato, Eric Kolve, Minjoon Seo, Hannaneh Hajishirzi, and Ali Farhadi. A diagram is worth a dozen images. In *European conference on computer vision*, pp. 235–251. Springer, 2016.

Aniruddha Kembhavi, Minjoon Seo, Dustin Schwenk, Jonghyun Choi, Ali Farhadi, and Hannaneh Hajishirzi. Are you smarter than a sixth grader? textbook question answering for multimodal machine comprehension. In *2017 IEEE conference on computer vision and pattern recognition (CVPR)*, pp. 5376–5384. IEEE, 2017.

Youngeun Kim, Youjia Zhang, Huiling Liu, Aecheon Jung, Sunwoo Lee, and Sungeun Hong. Zooprune: training-free token pruning via zeroth-order gradient estimation in vision-language models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 39572–39582, 2026.

Ranjay Krishna, Yuke Zhu, Oliver Groth, Justin Johnson, Kenji Hata, Joshua Kravitz, Stephanie Chen, Yannis Kalantidis, Li-Jia Li, David A Shamma, et al. Visual genome: Connecting language and vision using crowdsourced dense image annotations. *International journal of computer vision*, 123(1):32–73, 2017.

Bo Li, Yuanhan Zhang, Dong Guo, Renrui Zhang, Feng Li, Hao Zhang, Kaichen Zhang, Peiyuan Zhang, Yanwei Li, Ziwei Liu, et al. Llava-onevision: Easy visual task transfer. *arXiv preprint arXiv:2408.03326*, 2024a.

Hongliang Li, Jiaxin Zhang, Wenhui Liao, Dezhi Peng, Kai Ding, and Lianwen Jin. Redundancylens: Revealing and exploiting visual token processing redundancy for efficient decoder-only mllms. In *Findings of the Association for Computational Linguistics: ACL 2025*, pp. 24056–24067, 2025.

Kunchang Li, Yali Wang, Yinan He, Yizhuo Li, Yi Wang, Yi Liu, Zun Wang, Jilan Xu, Guo Chen, Ping Luo, et al. Mvbench: A comprehensive multi-modal video understanding benchmark. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 22195–22206, 2024b.

Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. Microsoft coco: Common objects in context. In *European conference on computer vision*, pp. 740–755. Springer, 2014.

Aoming Liu, Reuben Tan, Boqing Gong, and Bryan A Plummer. Fine-grained token allocation via operation pruning for efficient mllms. *arXiv preprint arXiv:2507.02909*, 2025.

Wenjie Liu, Hao Wu, Xin Qiu, Xudong Wang, Yingqi Fan, Yihan Zhang, Anhao Zhao, Yunpu Ma, and Xiaoyu Shen. Vica: Efficient multimodal llms with vision-only cross-attention. *arXiv preprint arXiv:2602.07574*, 2026.

Yuliang Liu, Zhang Li, Mingxin Huang, Biao Yang, Wenwen Yu, Chunyuan Li, Xu-Cheng Yin, Cheng-Lin Liu, Lianwen Jin, and Xiang Bai. Ocrbench: on the hidden mystery of ocr in large multimodal models. *Science China Information Sciences*, 67(12):220102, 2024.

Pan Lu, Swaroop Mishra, Tanglin Xia, Liang Qiu, Kai-Wei Chang, Song-Chun Zhu, Oyvind Tafjord, Peter Clark, and Ashwin Kalyan. Learn to explain: Multimodal reasoning via thought chains for science question answering. *Advances in neural information processing systems*, 35:2507–2521, 2022.

Junhua Mao, Jonathan Huang, Alexander Toshev, Oana Camburu, Alan L Yuille, and Kevin Murphy. Generation and comprehension of unambiguous object descriptions. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 11–20, 2016.

Kenneth Marino, Mohammad Rastegari, Ali Farhadi, and Roozbeh Mottaghi. Ok-vqa: A visual question answering benchmark requiring external knowledge. In *Proceedings of the IEEE/cvf conference on computer vision and pattern recognition*, pp. 3195–3204, 2019.

Ahmed Masry, Jia Qing Tan, Shafiq Joty, Enamul Hoque, et al. Chartqa: A benchmark for question answering about charts with visual and logical reasoning. In *Findings of the association for computational linguistics: ACL 2022*, pp. 2263–2279, 2022.

Minesh Mathew, Dimosthenis Karatzas, and CV Jawahar. Docvqa: A dataset for vqa on document images. In *Proceedings of the IEEE/CVF winter conference on applications of computer vision*, pp. 2200–2209, 2021.

Anand Mishra, Shashank Shekhar, Ajeet Kumar Singh, and Anirban Chakraborty. Ocr-vqa: Visual question answering by reading text in images. In *2019 international conference on document analysis and recognition (ICDAR)*, pp. 947–952. IEEE, 2019.

Pavlo Molchanov, Arun Mallya, Stephen Tyree, Iuri Frosio, and Jan Kautz. Importance estimation for neural network pruning. In *2019 IEEE/CVF conference on computer vision and pattern recognition (CVPR)*, pp. 11256–11264. IEEE, 2019.

Yuzhang Shang, Mu Cai, Bingxin Xu, Yong Jae Lee, and Yan Yan. Llava-prumerge: Adaptive token reduction for efficient large multimodal models. In *2025 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 22857–22867. IEEE, 2025.

Jiaqi Shi, Xulong Zhang, Yuechan Li, Xiaoyang Qu, and Jianzong Wang. From inheritance to saturation: Disentangling the evolution of visual redundancy for architecture-aware mllm inference acceleration. In *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 17328–17357, 2026.

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*, pp. 8317–8326, 2019.

Mingjie Sun, Zhuang Liu, Anna Bair, and Zico Kolter. A simple and effective pruning approach for large language models. In *International Conference on Learning Representations*, volume 2024, pp. 4942–4964, 2024.

Zichen Wen, Yifeng Gao, Shaobo Wang, Junyuan Zhang, Qintong Zhang, Weijia Li, Conghui He, and Linfeng Zhang. Stop looking for "important tokens" in multimodal language models: Duplication matters more. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pp. 9972–9991, 2025.

Hao Wu, Yingqi Fan, Dai Jinyang, Junlong Tong, Yunpu Ma, and Xiaoyu Shen. Hidrop: Hierarchical vision token reduction in mllms via late injection, concave pyramid pruning, and early exit. In *International Conference on Learning Representations*, volume 2026, pp. 97296–97316, 2026a.

Hao Wu, Junlong Tong, Xudong Wang, Yang Tan, Changyu Zeng, Anastasia Antsiferova, and Xiaoyu Shen. From data to model: A survey of the compression lifecycle in mllms. *TechRxiv*, 2026b.

Hao Wu, Xudong Wang, Jialiang Zhang, Junlong Tong, Xinghao Chen, Junyan Lin, Yunpu Ma, and Xiaoyu Shen. Utptrack: Towards simple and unified token pruning for visual tracking. *arXiv preprint arXiv:2602.23734*, 2026c.

Junbin Xiao, Xindi Shang, Angela Yao, and Tat-Seng Chua. Next-qa: Next phase of question-answering to explaining temporal actions. In *2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 9772–9781. IEEE, 2021.

Long Xing, Qidong Huang, Xiaoyi Dong, Jiajie Lu, Pan Zhang, Yuhang Zang, Yuhang Cao, Conghui He, Jiaqi Wang, Feng Wu, et al. Pyramiddrop: Accelerating your large vision-language models via pyramid visual redundancy reduction. *arXiv preprint arXiv:2410.17247*, 2024.

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. In *2025 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 19792–19802. IEEE, 2025.

Licheng Yu, Patrick Poirson, Shan Yang, Alexander C Berg, and Tamara L Berg. Modeling context in referring expressions. In *European conference on computer vision*, pp. 69–85. Springer, 2016.

Qianhao Yuan, Qingyu Zhang, Yanjiang Liu, Jiawei Chen, Yaojie Lu, Hongyu Lin, Jia Zheng, Xianpei Han, and Le Sun. Shortv: Efficient multimodal large language models by freezing visual tokens in ineffective layers. In *2025 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 329–339. IEEE, 2025.

Weili Zeng, Ziyuan Huang, Kaixiang Ji, and Yichao Yan. Skip-vision: Efficient and scalable acceleration of vision-language models via adaptive token skipping. In *2025 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 21384–21397. IEEE, 2025.

Qizhe Zhang, Aosong Cheng, Ming Lu, Renrui Zhang, Zhiyong Zhuo, Jiajun Cao, Shaobo Guo, Qi She, and Shanghang Zhang. Beyond text-visual attention: Exploiting visual cues for effective token pruning in vlms. In *2025 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 20857–20867. IEEE, 2025.

Yanzhe Zhang, Ruiyi Zhang, Jiuxiang Gu, Yufan Zhou, Nedim Lipka, Diyi Yang, and Tong Sun. Llavar: Enhanced visual instruction tuning for text-rich image understanding. *arXiv preprint arXiv:2306.17107*, 2023.

Zeliang Zhang, Phu Pham, Wentian Zhao, Kun Wan, Yu-Jhe Li, Jianing Zhou, Daniel Miranda, Ajinkya Kale, and Chenliang Xu. Treat visual tokens as text? but your MLLM only needs fewer efforts to see. *arXiv preprint arXiv:2410.06169*, 2024a.

Zicheng Zhang, Haoning Wu, Erli Zhang, Guangtao Zhai, and Weisi Lin. Q-Bench^{++}: A benchmark for multi-modal foundation models on low-level vision from single images to pairs. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 46(12):10404–10418, 2024b.

This appendix supplements the main text with further analyses of attention and FFN redundancy, detailed results across tasks and backbones, and experimental settings for pruning, recovery, and efficiency evaluation. It is organized as follows:

- Section A: attention-path importance, task-dependent redundancy, and calibration-task effects.
- Section B: modality-conditioned FFN importance, rescoring after attention pruning, and modality-specific masking sensitivity.
- Section C: detailed pruning results across benchmarks and backbones, and input-length effects on computational efficiency.
- Section D: model and pruning configurations, sequential search and recovery, training settings, and efficiency evaluation protocols.
- Section E: descriptions of the comparison methods.
- Section F: descriptions of the evaluation benchmarks.

A More Attention Analysis

We extend the analysis in Section 4 with a detailed investigation of attention-path importance and redundancy across General, Reasoning, OCR, and Grounding. We first compare importance estimation criteria (Section A.1) and examine the layer-wise distribution of path importance (Section A.2). We then analyze path-conditional redundancy across task categories (Section A.3), assess the cross-task similarity of importance rankings (Section A.4), and evaluate how the calibration task affects pruning performance (Section A.5). Together, these analyses provide empirical support for path-wise pruning and inform the choice of importance criteria and calibration data.

A.1 Attention Importance Estimation

Each attention head is decomposed into three pruning units corresponding to V2V, T2V, and T2T. We distinguish attention-weight statistics, output-contribution statistics, and loss-based importance, since these quantities characterize different aspects of a path.

Attention Mass. Attention mass measures the normalized attention weight assigned to a path. For a query token i and a path with key positions $\mathcal{K}_b$, its path-specific mass is $\sum_{j\in\mathcal{K}_b} A_{ij}$, where A is the attention matrix of the corresponding head. This statistic describes the distribution of attention across key modalities, without accounting for value vectors or sensitivity of the prediction loss.

Attention Contribution. Attention contribution characterizes the output produced by a path rather than its attention weights alone. For a path b whose key/value modality is ν, this contribution is $O_b = A_b U_\nu$, where A_b is the corresponding attention block and U_ν contains the projected value vectors. We use the sample-averaged score $\|O_b\|_{\mathrm{F}}/|\mathcal{Q}_b|$, where $\|\cdot\|_{\mathrm{F}}$ denotes the Frobenius norm and $|\mathcal{Q}_b|$ is the number of query tokens. Contributions are measured before the output projection.

First-Order Taylor. For a pruning unit u, the Taylor score approximates the magnitude of the loss change induced by gating its output to zero (Molchanov et al., 2019):

$$S(u) = \frac{1}{|\mathcal{D}_{\mathrm{cal}}|} \sum_{x\in\mathcal{D}_{\mathrm{cal}}} \left| \sum_k o_{u,k}(x) \frac{\partial \mathcal{L}_x}{\partial o_{u,k}(x)} \right|,$$

where $\mathcal{D}_{\mathrm{cal}}$ is the calibration set, $\mathcal{L}_x$ is the sample loss, and $o_{u,k}(x)$ is the k-th element of the unit's output. Within each sample, activation-gradient products are summed before taking the absolute value. The resulting magnitudes are then averaged across samples.

For attention, each unit is a path $b \in \{\mathrm{VV}, \mathrm{TV}, \mathrm{TT}\}$ of a head at a given layer. We compute the additive contribution $O_b = A_b U_\nu$ before the attention output projection. We use the following per-query-normalized Taylor score as a *path contribution sensitivity proxy*:

$$S^{\mathrm{attn}}_{\ell,h,b} = \frac{1}{|\mathcal{D}_{\mathrm{cal}}|} \sum_{x\in\mathcal{D}_{\mathrm{cal}}} \frac{\left|\left\langle O_{\ell,h,b}(x), \nabla_{O_{\ell,h}(x)} \mathcal{L}_x \right\rangle\right|}{|\mathcal{Q}_b(x)|},$$

where $\mathcal{Q}_b(x)$ denotes the query set, $O_{\ell,h}(x)$ is the full head output before projection, and the inner product sums over queries and head dimensions. We use teacher-forced answer cross-entropy for $\mathcal{L}_x$. With attention weights held fixed, the score measures the loss sensitivity to gating $O_{\ell,h,b}$ and excludes the softmax renormalization induced by inference masking. At inference, we apply path masks before softmax and normalize over the retained causal keys. The sparse kernel skips the masked QK and attention-value products to reduce FLOPs (Section D.5).

Comparison of importance criteria. Figure 6 compares the criteria across four representative benchmarks. Taylor generally preserves performance more effectively as masking increases, with clear advantages for V2V on TextVQA and RefCOCOg and for T2T across the evaluated benchmarks. Attention Contribution exhibits sharp T2T performance drops at moderate masking ratios, while Attention Mass remains more competitive but degrades earlier than Taylor on RefCOCOg. Differences are smaller for T2V, where Attention Contribution is competitive on TextVQA.

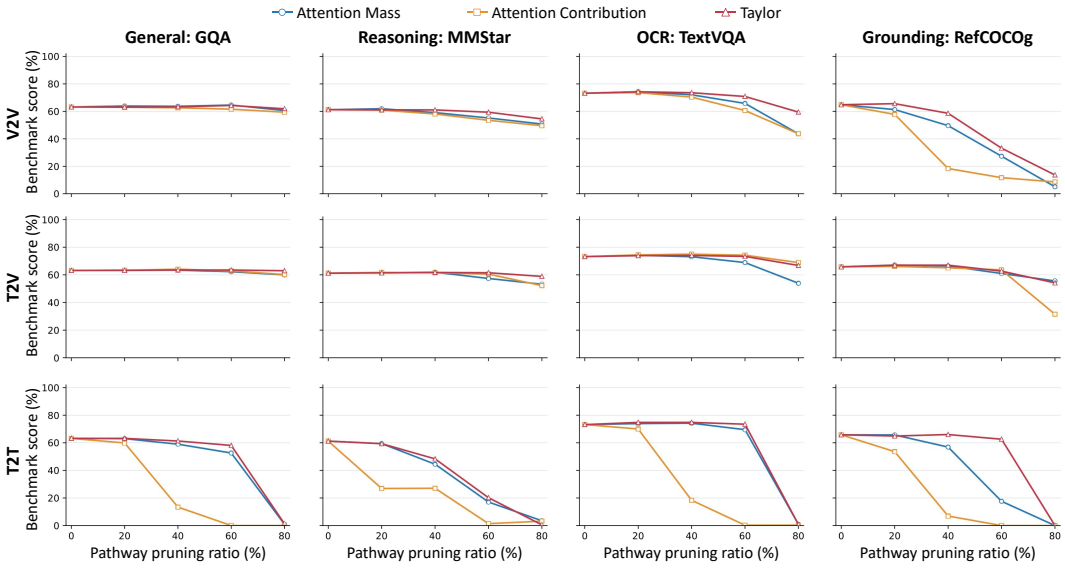


Figure 6: **Comparison of attention-path importance criteria on LLaVA-OneVision-7B.** Rows correspond to V2V, T2V, and T2T masking. Columns respectively show GQA, MMStar, TextVQA, and RefCOCOg, representing General, Reasoning, OCR, and Grounding. Curves use Attention Mass, Attention Contribution, or first-order Taylor importance for training-free path selection.

A.2 Layer-Wise Distribution of Path Importance

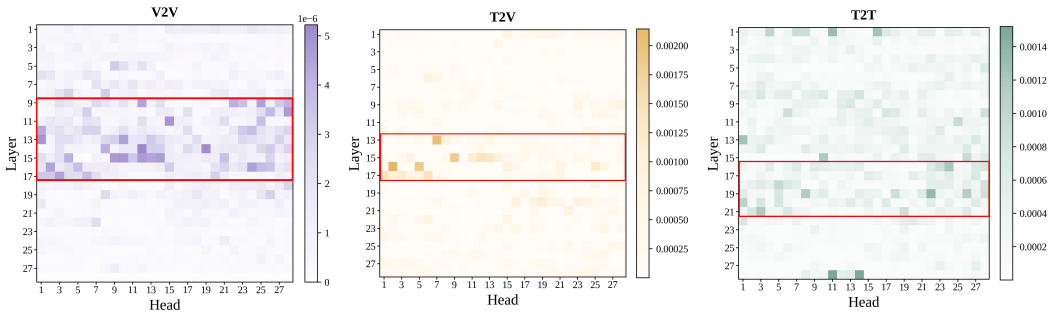


Figure 7: **Layer-head distribution of attention-path Taylor importance on LLaVA-OneVision-7B.** Panels show V2V, T2V, and T2T. Rows and columns index decoder layers and query heads, using one-based labels. Darker cells indicate larger importance scores, and red boxes highlight the depth ranges discussed in the text.

Figure 7 shows that the three paths have different depth profiles. High V2V importance is concentrated approximately in layers 9 to 17 and becomes much weaker beyond layer 19. T2V importance

is more localized, with prominent bands around layers 13 to 17 and a small number of high-scoring heads. T2T importance is distributed more broadly, with several prominent heads around layers 16 to 21 as well as isolated peaks in other decoder layers.

The overlapping high-importance regions are consistent with a gradual shift from visual processing to cross-modal integration and text-side computation. This pattern also motivates our sequential search order: first V2V, then T2V, and finally T2T. The nonuniform profiles also suggest that uniform layer-wise allocation can overlook important paths. Each panel uses a separate color scale, with intensities indicating relative importance within that path.

A.3 Path-Conditional Redundancy across Task Categories

Overlap of low-importance paths. Figure 8 extends the overlap analysis in Figure 3 to all four task categories. The three-way intersection accounts for only 32.1%, 49.5%, 28.8%, and 36.0% of each path's low-importance set for General, Reasoning, OCR, and Grounding. Thus, low importance on one path does not imply that the entire head is redundant.

Across all four categories, the V2V and T2V intersection is larger than the T2V and T2T intersection. This indicates that the low-importance patterns of the two visually related paths overlap more strongly, whereas text-side redundancy is less consistently aligned with cross-modal redundancy. These differences motivate selecting each path according to its own importance, rather than assigning a shared pruning decision to all three paths within a head.

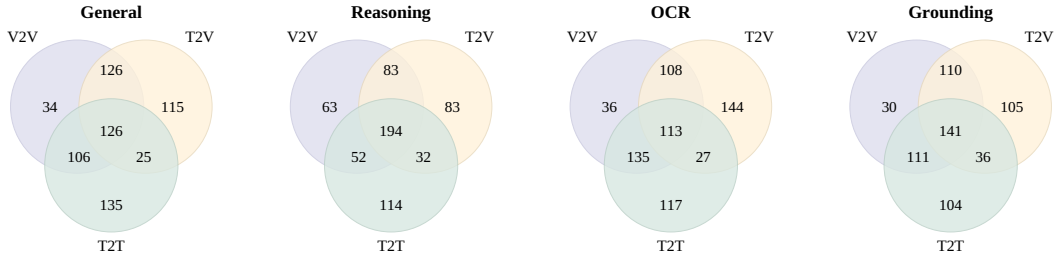


Figure 8: **Overlap of low-importance attention paths across four task categories.** Each circle contains the layer-head identities in the bottom 50% of Taylor scores for one path: 392 of 784 identities on LLaVA-OneVision-7B. Region labels give the numbers of identities in the corresponding exclusive or shared sets. The central region contains heads whose three paths all lie in their respective low-importance sets.

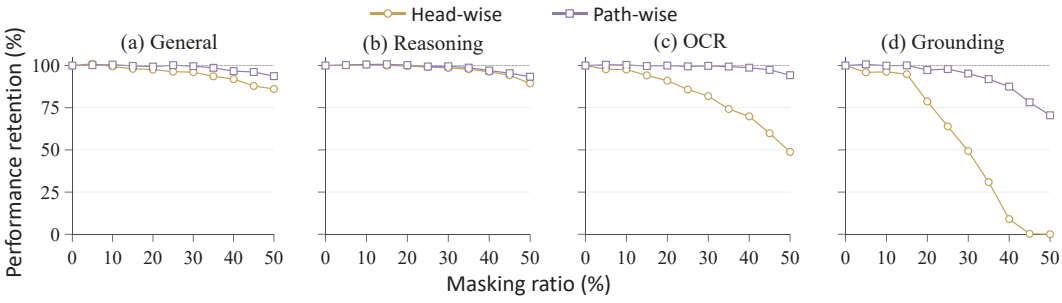


Figure 9: **Whole-head versus path-wise attention masking on LLaVA-OneVision-7B.** Both use Taylor importance. The x-axis gives the fraction of masked heads or the independently masked fraction of each path type, matching the total number of removed path blocks up to rounding. Panels show training-free performance retention on the four task categories relative to the dense reference. The dashed line denotes 100% retention.

Whole-head versus path-wise masking. Figure 9 compares Taylor-guided masking at the two granularities. At ratio q, head-wise masking removes a fraction q of heads and all three of their paths, whereas path-wise masking selects a fraction q independently within each path type. The advantage of independent path selection is especially pronounced on OCR and Grounding. At 30% masking, Grounding retains 95.3% performance under path-wise selection, compared with 49.3% under whole-head selection. General and Reasoning exhibit smaller gaps over the same range.

Sensitivity to individual paths. Figure 10 isolates each path by masking it while keeping the other two dense. The sensitivity profiles differ substantially across tasks. General tolerates extensive V2V masking, whereas Grounding deteriorates much earlier when the same path is removed. In contrast, Grounding retains over 90% performance at 70% T2V masking. Aggressive T2T masking produces a sharp collapse across categories, although the onset differs by task. These single-path interventions support using task-dependent pruning ratios rather than treating all three paths as equally redundant.

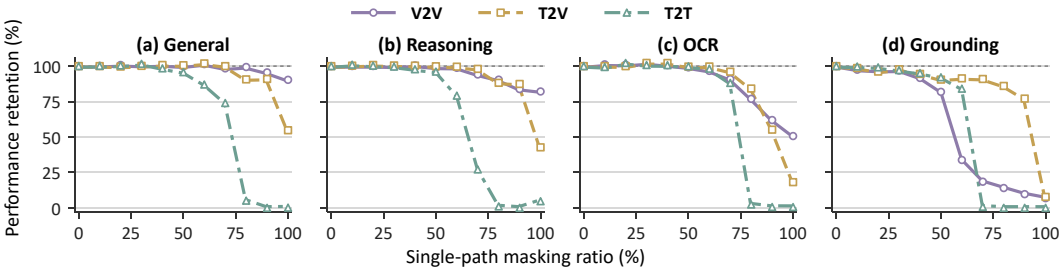


Figure 10: **Training-free sensitivity to single-path attention masking on LLaVA-OneVision-7B.** In each curve, Taylor-guided masking is applied only to the indicated path, with the other two paths kept dense. The horizontal axis gives the masking ratio of that path. The vertical axis gives performance retention relative to the dense reference. Panels cover the four task categories, and the dashed line denotes 100% retention.

A.4 Cross-Task Similarity of Attention Importance

Figure 11 reports Spearman rank correlations between task-specific Taylor importance profiles, computed over the 784 layer-head identities separately for each path. V2V has consistently high cross-task similarity, with off-diagonal values ranging from 0.88 to 0.98. The corresponding ranges are 0.70 to 0.93 for T2V and 0.55 to 0.91 for T2T. Reasoning is particularly distinct on T2T, with similarities of 0.55, 0.60, and 0.57 to General, OCR, and Grounding, respectively.

These results indicate that visual-side importance is more consistent across the evaluated task categories, whereas text-side importance is more task-dependent. We further compare calibration tasks to evaluate how these ranking similarities translate into transferable pruning masks.

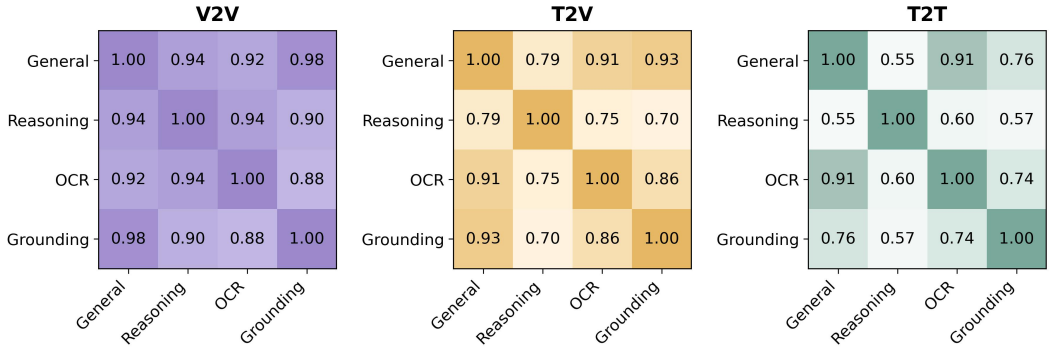


Figure 11: **Cross-task similarity of attention-path Taylor importance on LLaVA-OneVision-7B.** Each matrix compares the layer-head importance profiles of General, Reasoning, OCR, and Grounding for one path. Cell values report pairwise Spearman rank correlations, with larger values indicating more similar rankings.

A.5 Effect of the Calibration Task

We compare calibration tasks under fixed $(\text{V2V}, \text{T2V}, \text{T2T})$ masking ratios of $(40\%, 60\%, 10\%)$ and the same recovery protocol. Each single-category setting uses 256 calibration samples, while *All Tasks* combines four category-specific rankings estimated from 256 samples each. The aggregate rank is $r(u) = \min_c r_c(u)$, where $r_c(u)$ is the rank of unit u in category c and rank 1 denotes highest importance, favoring units important to at least one category.

Table 7: **Effect of the calibration task on attention-path pruning.** Performance retention (%) is measured relative to the unpruned, post-trained LLaVA-OneVision-7B.

Setting	Aggregate (%)			Category (%)			
	ATR	GRR	OGR	Gen.	Reas.	OCR	Grd.
Dense	100.0	100.0	100.0	100.0	100.0	100.0	100.0
All Tasks	**100.0**	**99.9**	100.1	**100.3**	**99.5**	99.9	100.3
General	99.5	99.4	99.6	100.2	98.6	98.9	100.2
Reasoning	96.6	98.6	94.5	97.9	99.3	90.3	98.8
OCR	99.9	99.1	**100.7**	99.8	98.5	**100.6**	**100.8**
Grounding	99.3	99.5	99.2	100.1	98.9	98.6	99.7

Table 7 shows that OCR-guided ranking achieves the highest single-category ATR (99.9%), with category-wise retention ranging from 98.5% to 100.8%. In contrast, Reasoning-guided ranking yields 96.6% ATR and only 90.3% OCR retention, suggesting that OCR-based calibration produces a more transferable importance ranking in this comparison. With its larger calibration budget, *All Tasks* achieves 100.0% ATR and a narrower category-wise retention range of 99.5% to 100.3%.

B More FFN Analysis

We extend the analysis in Section 5 with a detailed investigation of modality-conditioned FFN importance and redundancy. We first compare channel importance criteria (Section B.1) and examine ranking changes and the effect of rescoring after attention pruning (Section B.2). We then compare shared and modality-specific channel masks and analyze visual-side versus text-side sensitivity across General, Reasoning, OCR, and Grounding (Section B.3). Together, these analyses inform FFN importance estimation and the allocation of pruning budgets between visual and text tokens.

B.1 Importance of Modality-Conditioned FFN Channels

An FFN pruning unit is an intermediate channel on visual or text tokens. Let $h_{\ell,i,c}(x)$ be its gated activation immediately before the down projection. FFN weights are shared across modalities, but channel importance and the resulting masks are modality-specific. We compare three criteria.

Activation. For channel c at layer ℓ and modality μ, the mean absolute activation is

$$S^{\text{act}}_{\ell,c,\mu} = \frac{\sum_{x\in\mathcal{D}_{\text{cal}}} \sum_{i\in\mathcal{I}_\mu(x)} |h_{\ell,i,c}(x)|}{\sum_{x\in\mathcal{D}_{\text{cal}}} |\mathcal{I}_\mu(x)|},$$

where $\mathcal{I}_\mu(x)$ denotes the token positions of modality μ. This score measures channel activity.

Wanda. We adapt Wanda (Sun et al., 2024) to structured channels by weighting activation with the Euclidean norm of the corresponding down-projection column:

$$S^{\text{Wanda}}_{\ell,c,\mu} = \|W_{\text{down},\ell}[:,c]\|_2 S^{\text{act}}_{\ell,c,\mu}.$$

Taylor. For a modality-specific channel gate, the Taylor score is

$$S^{\text{gate}}_{\ell,c,\mu} = \frac{1}{|\mathcal{D}_{\text{cal}}|} \sum_{x\in\mathcal{D}_{\text{cal}}} \left| \sum_{i\in\mathcal{I}_\mu(x)} h_{\ell,i,c}(x) \frac{\partial \mathcal{L}_x}{\partial h_{\ell,i,c}(x)} \right|.$$

We sum activation-gradient products over tokens within each sample before taking the absolute value, then average across samples. This score is used in Figure 12. For joint pruning, FFN scores are recomputed on the initial model with the selected attention mask applied.

Comparison of importance criteria. For a controlled comparison, the three scores use the same calibration samples. For each criterion, we rank channels within each layer and task, average the ranks equally across the four tasks, and apply matched per-layer channel budgets.

Figure 12 shows that importance-guided masking generally outperforms random selection. The three criteria perform similarly for visual-side masking. Under aggressive text-side masking, Taylor has a clearer advantage on GQA, MMStar, and TextVQA, although it does not lead at every ratio on RefCOCOg. Thus, the benefit of loss-based ranking depends on both task and token modality.

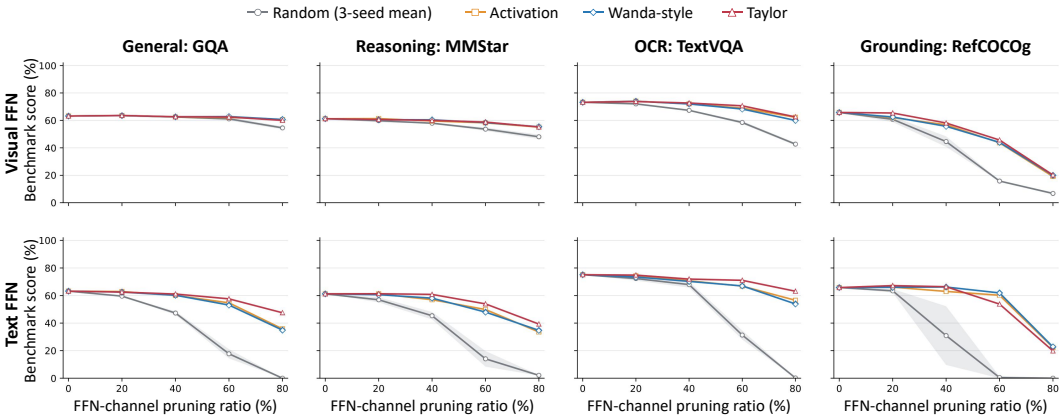


Figure 12: **Comparison of FFN-channel importance criteria on LLaVA-OneVision-7B.** Columns show GQA, MMStar, TextVQA, and RefCOCOg for General, Reasoning, OCR, and Grounding. The top and bottom rows respectively mask visual-side and text-side channels. Random curves average three seeds.

B.2 FFN Scoring after Attention Pruning

Figure 13 compares FFN Taylor rankings from the same initial checkpoint before and after applying the selected attention mask. Visual token rankings remain relatively stable across layers, whereas text token rankings change more through much of the middle and later decoder. Rankings computed over all tokens show an intermediate pattern. These differences motivate estimating FFN importance in the model state in which the selected FFN masks will subsequently be applied.

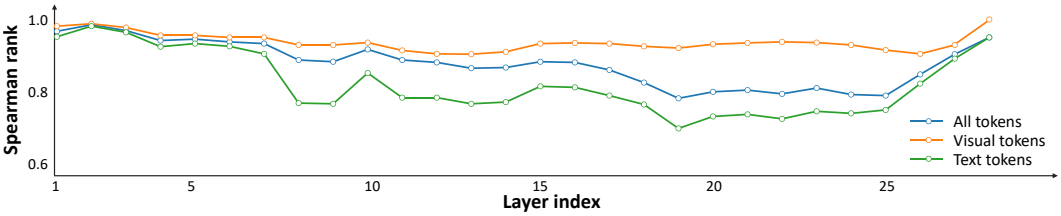


Figure 13: **Layer-wise agreement of FFN Taylor rankings across model states.** Spearman rank correlation compares channel rankings from the same initial checkpoint before and after applying the selected attention mask, with scores estimated from all tokens, visual tokens, or text tokens.

In Table 8, *Dense* is the post-trained LLaVA-OneVision-7B reference without attention or FFN masking, used to normalize retention metrics. *Old* computes FFN Taylor rankings on the initial unpruned checkpoint, while *New* uses the same weights with attention masking ratios of $(40\%, 60\%, 10\%)$ for $(\mathrm{V2V}, \mathrm{T2V}, \mathrm{T2T})$. Both settings apply the attention and visual-side FFN masks to the initial checkpoint and perform one epoch of recovery, with text-side FFNs kept dense.

Recomputing FFN Taylor importance rankings yields larger aggregate gains as the visual-side masking ratio increases. At 40% masking, pruning based on the recomputed rankings achieves 100.0% ATR, compared with 94.1% using the original rankings, with GRR and OGR gains of 7.2 and 4.6 percentage points, respectively. These results support recomputing FFN importance with the selected attention mask applied before choosing the modality-specific channel masks for recovery.

Table 8: **Effect of recomputing FFN rankings under attention masking.** Dense is the unmasked post-trained reference used to normalize ATR, GRR, and OGR. Bold marks the better Old/New result at each masking ratio. Arrows indicate changes from Old to New in percentage points.

Ratio	Setting	General			Reasoning			OCR			Grounding			ATR	GRR	OGR
		GQA	VQA^{v2}	VQA^{ok}	SQA	AI2D	MMStar	VQA^{T}	VQA^{D}	OCRB	Ref	Ref+	RefG	(%)	(%)	(%)
0%	Dense	63.1	82.2	59.9	96.0	81.9	61.4	75.3	87.4	63.5	77.9	70.5	76.9	100.0	100.0	100.0
10%	Old	63.2	81.8	**59.5**	95.4	81.0	60.1	73.6	85.1	**66.6**	**77.8**	**69.9**	**76.4**	99.4	99.2	**99.7**
	New	**64.4**	**83.6**	56.9	**96.9**	**82.3**	**63.6**	**74.8**	**85.3**	65.7	77.1	69.8	76.2	**100.1** (↑ 0.7)	**100.6** (↑ 1.5)	99.6 (↓ 0.1)
20%	Old	63.0	81.6	**58.2**	94.1	79.8	59.1	72.4	84.0	64.9	76.7	69.0	**75.5**	98.1	98.0	98.2
	New	**64.0**	**82.8**	55.7	**97.0**	**82.3**	**63.8**	**73.6**	**85.8**	**65.7**	**77.2**	**69.9**	75.2	**99.7** (↑ 1.6)	**100.1** (↑ 2.1)	**99.2** (↑ 1.1)
30%	Old	62.9	80.9	**56.4**	89.9	78.8	58.0	70.6	82.8	64.5	76.0	68.4	75.0	96.5	96.1	97.0
	New	**64.4**	**83.0**	56.3	**97.0**	**81.9**	**64.0**	**75.0**	**86.0**	**65.1**	**78.0**	**70.5**	**76.6**	**100.2** (↑ 3.7)	**100.4** (↑ 4.3)	**100.0** (↑ 3.0)
40%	Old	62.2	80.3	52.7	83.9	76.1	56.3	69.3	81.4	64.5	74.3	66.9	73.4	94.1	92.7	95.4
	New	**63.4**	**82.0**	**60.0**	**96.2**	**81.3**	**61.3**	**74.1**	**85.2**	**65.6**	**77.8**	**70.8**	**77.3**	**100.0** (↑ 5.9)	**99.9** (↑ 7.2)	**100.0** (↑ 4.6)

B.3 Modality-Conditioned Redundancy across Task Categories

Shared versus modality-specific channel masks. At masking ratio r, *Global FFN* uses one low-importance channel set for both modalities, whereas *Vision+Text FFN* selects a separate bottom-r set for visual and text tokens. Both settings suppress the same fraction of channel executions for each modality, so their comparison isolates the effect of allowing modality-specific channel identities. Figure 14 shows that the advantage of Vision+Text grows with the masking ratio on General, OCR, and Grounding, while the curves remain close on Reasoning. Thus, modality conditioning improves the trade-off between performance and compression for several task categories.

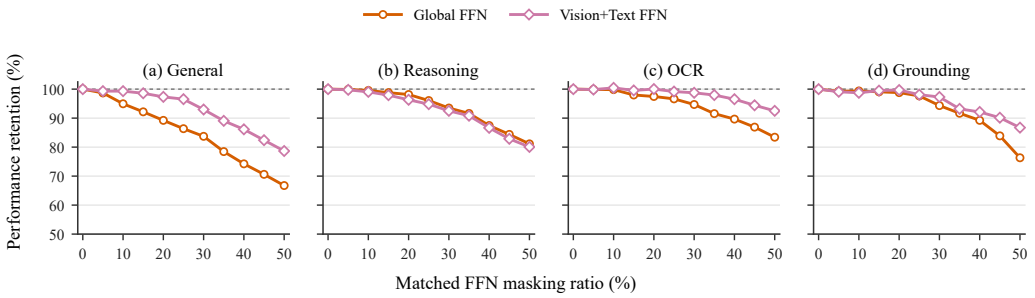


Figure 14: **Shared versus modality-specific FFN masks across four task categories.** All curves use Taylor-guided selection on LLaVA-OneVision-7B under the fixed $(40\%, 60\%, 10\%)$ attention-path mask and report training-free performance retention relative to the attention-only reference.

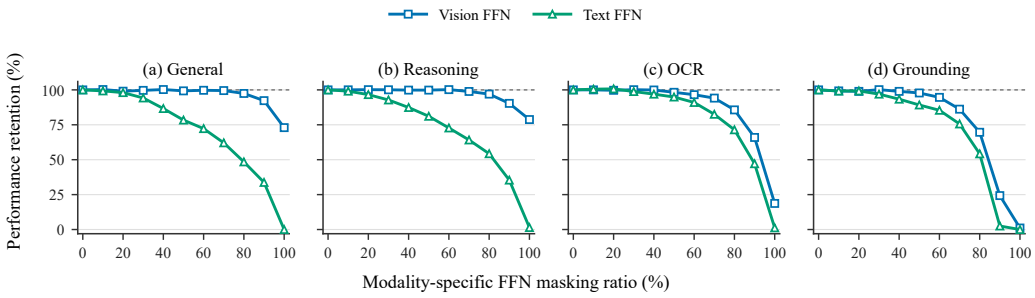


Figure 15: **Sensitivity to modality-specific FFN masking.** Vision FFN and Text FFN mask visual and text token channel executions respectively, while keeping the other modality dense. Masks are selected by Taylor importance on LLaVA-OneVision-7B with the fixed $(40\%, 60\%, 10\%)$ attention-path mask.

Visual-side versus text-side sensitivity. Figure 15 shows that visual-side FFN masking preserves General and Reasoning performance over a broad range, whereas text-side masking causes an earlier decline. At moderate text-side masking ratios, General and Reasoning degrade more than OCR and Grounding. Under very aggressive visual-side masking, however, OCR and Grounding also

deteriorate sharply. The relative sensitivity of the task categories varies with token modality and masking ratio. These results also support concentrating the FFN pruning budget on visual-side executions while conservatively retaining text-side computation.

C Additional Results

We present detailed pruning results and a complementary analysis of input-length effects. We first report per-benchmark attention-path and modality-conditioned FFN masking results on LLaVA-OneVision-7B (Section C.1), followed by corresponding results and comparisons with compression methods on Qwen2.5-VL-7B (Section C.2). We then discuss how input sequence length affects computational costs and the interpretation of speedup (Section C.3). Together, these results characterize performance retention and computational efficiency across tasks, backbones, and input profiles.

For single-image evaluation, category retention averages the three benchmark score ratios to their post-trained dense references (in percent), and ATR, GRR, and OGR average category retentions over all four categories, General and Reasoning, and OCR and Grounding, respectively.

C.1 Detailed Results on LLaVA-OneVision-7B

Attention-path masking. Table 9 details the benchmark-level changes underlying Table 1. Increasing V2V masking from 30% to 60% lowers DocVQA from 87.1 to 83.6, while GQA and VQAv2 remain nearly unchanged. Similarly, with V2V fixed at 40%, increasing T2V masking from 50% to 80% reduces DocVQA from 85.7 to 81.9 and TextVQA from 74.7 to 72.0. Document and scene-text understanding expose losses that can be obscured by stable general VQA scores.

The T2T sweep exhibits a different pattern. Increasing its masking ratio from 10% to 40% lowers MMStar from 60.5 to 58.1, while the three grounding scores remain stable or improve. Thus, near-dense aggregate retention need not imply uniform preservation across benchmarks. Selection also matters within a fixed budget: at 40% V2V masking, Taylor exceeds random selection by 6.5 points on DocVQA and 6.9 points on RefCOCO+, despite identical masking ratios and recovery settings.

Table 9: **Per-benchmark attention-path masking results on LLaVA-OneVision-7B after recovery.** Setting identifies the path varied at each stage, and the three mask columns specify the complete configuration.

Setting	Path Mask (%)			General			Reasoning			OCR			Grounding			ATR	GRR	OGR
	V2V	T2V	T2T	GQA	VQA^{v2}	VQA^{ok}	SQA	AI2D	MMStar	VQA^{T}	VQA^{D}	OCRB	Ref	Ref+	RefG	(%)	(%)	(%)
Dense Model	0	0	0	63.1	82.2	59.9	96.0	81.9	61.4	75.3	87.4	63.5	77.9	70.5	76.9	100.0	100.0	100.0
V2V	30	0	0	63.2	**82.2**	60.4	**96.4**	**81.6**	61.2	**75.2**	**87.1**	**64.0**	**78.5**	70.2	**77.0**	**100.1**	100.1	**100.1**
	40	0	0	63.2	**82.2**	**60.9**	96.2	**81.6**	**61.7**	74.5	86.7	63.7	78.0	**70.4**	76.8	100.0	**100.3**	99.7
	40^{R}	0	0	62.8	81.6	60.3	94.7	79.9	59.3	73.1	80.2	61.7	71.9	63.5	71.0	96.0	98.7	93.4
	50	0	0	63.2	**82.2**	60.8	96.2	81.5	61.6	74.5	85.6	63.4	77.6	70.0	76.7	99.7	**100.3**	99.2
	60	0	0	**63.4**	82.1	60.1	95.9	81.4	60.9	74.1	83.6	61.8	77.0	69.1	75.8	98.8	99.8	97.8
T2V	40	50	0	**63.3**	**82.1**	60.0	**96.6**	81.5	61.1	74.7	**85.7**	**66.1**	**78.5**	71.5	**78.0**	**100.4**	100.0	**100.8**
	40	60	0	**63.3**	82.0	**60.3**	96.4	**81.8**	61.3	74.6	85.3	65.5	78.4	**71.8**	77.8	100.3	**100.1**	100.6
	40	60^{R}	0	62.4	80.8	57.6	91.5	78.4	54.9	70.4	81.3	64.5	73.1	65.2	71.0	95.0	95.6	94.5
	40^{R}	60^{R}	0	61.7	80.1	58.5	68.5	74.0	**61.9**	**88.1**	77.1	52.7	67.1	58.2	63.9	91.3	92.6	90.0
	40	70	0	63.2	81.9	59.2	96.4	81.4	60.9	73.6	84.7	65.0	77.5	70.4	76.9	99.5	99.6	99.4
	40	80	0	62.8	81.4	58.0	95.7	81.5	59.9	72.0	81.9	65.3	77.3	69.7	76.6	98.5	98.7	98.3
T2T	40	60	10	**63.3**	**82.0**	**60.4**	**96.5**	81.5	60.5	74.0	85.3	**65.9**	78.1	71.0	77.0	**100.0**	**99.9**	100.1
	40	60	10^{R}	**63.3**	81.9	58.8	96.2	81.3	59.3	74.5	85.4	65.8	78.2	70.5	77.8	99.7	99.0	100.3
	40^{R}	60^{R}	10^{R}	61.4	80.0	56.0	86.6	75.4	52.6	69.5	73.6	60.3	66.1	56.6	61.7	89.4	92.7	86.1
	40	60	20	63.1	81.9	60.0	96.0	**81.7**	**61.2**	**74.9**	85.5	65.5	77.4	70.8	76.3	99.9	**99.9**	99.9
	40	60	30	63.1	81.9	60.0	95.8	81.0	59.2	74.1	**85.7**	64.7	**78.7**	71.4	77.7	99.7	99.1	100.2
	40	60	40	63.0	81.9	59.9	94.7	80.3	58.1	74.0	**85.7**	64.5	**78.7**	**71.9**	**77.9**	99.4	98.5	**100.4**

Modality-conditioned FFN masking. Table 10 decomposes the aggregate FFN results in Table 2. With text-side FFNs kept dense, raising visual-side masking from 60% to 70% reduces OK-VQA from 60.1 to 50.0, whereas GQA changes from 63.1 to 63.3 and VQAv2 from 81.9 to 81.7. The decline in General performance at this setting is therefore uneven across its constituent benchmarks, with a particularly sharp loss on the knowledge-oriented OK-VQA benchmark.

The text-side sweep reveals a complementary concentration of errors. With visual-side masking fixed at 40%, increasing text-side masking from 0% to 30% lowers ScienceQA from 96.2 to 86.7 and OK-VQA from 60.0 to 54.3. Over the same sweep, RefCOCO+ changes only from 70.8 to 70.4 and RefCOCOg from 77.3 to 76.6. These differences motivate checking individual benchmarks alongside overall retention when choosing modality-specific FFN budgets.

Table 10: **Per-benchmark FFN masking results on LLaVA-OneVision-7B after recovery.** Visual and Text specify modality-conditioned channel masking ratios under the fixed $(40\%, 60\%, 10\%)$ attention-path mask. $\text{MWOP}_{\text{Attn}}$ is the attention-only reference with dense FFNs.

Setting	Channel Mask (%) Visual	Text	General GQA	VQA^{v2}	VQA^{ok}	Reasoning SQA	AI2D	MMStar	OCR VQA^{T}	VQA^{D}	OCRB	Grounding Ref	Ref+	RefG	ATR (%)	GRR (%)	OGR (%)
$\text{MWOP}_{\text{Attn}}$	0	0	63.3	82.0	60.4	96.5	81.5	60.5	74.0	85.3	65.9	78.1	71.0	77.0	100.0	99.9	100.1
Visual Neuron	40	0	**63.4**	**82.0**	60.0	**96.2**	81.3	**61.3**	**74.1**	**85.2**	**65.6**	77.8	**70.8**	**77.3**	**100.0**	**99.9**	**100.0**
	40^{R}	0	63.1	81.0	58.9	92.3	79.4	57.5	70.9	80.4	61.5	76.0	67.7	74.7	96.4	97.3	95.6
	50	0	**63.4**	**82.0**	**60.5**	**96.2**	81.3	60.5	73.8	85.0	64.5	**77.9**	70.7	77.1	99.7	**99.9**	99.6
	50^{R}	0	62.9	80.5	58.6	91.2	78.7	56.2	70.3	77.4	61.2	74.4	66.0	72.5	94.9	96.3	93.6
	60	0	63.1	81.9	60.1	96.0	**81.4**	60.7	72.8	84.4	64.5	77.5	70.2	76.8	99.3	99.7	98.9
	70	0	63.3	81.7	50.0	94.9	80.8	59.7	72.3	82.5	64.2	76.8	69.3	75.7	97.0	96.3	97.8
Text Neuron	40	10	**63.0**	**81.8**	**60.0**	**95.4**	**81.4**	**60.4**	73.5	**85.4**	64.5	77.9	**71.1**	76.7	**99.5**	**99.4**	**99.6**
	40	10^{R}	62.9	81.4	56.4	93.7	80.0	59.0	**73.6**	85.1	62.0	**78.1**	70.8	76.8	98.1	97.3	98.9
	40^{R}	10^{R}	62.7	80.4	55.2	89.6	77.1	55.5	70.7	79.7	60.8	74.2	65.5	72.5	94.2	94.5	93.9
	40	20	62.7	81.4	57.8	90.8	80.2	59.6	73.0	85.0	**65.0**	77.5	70.7	**76.9**	98.4	97.4	99.4
	40	30	62.4	80.7	54.3	86.7	77.9	59.5	73.1	84.7	64.0	**78.1**	70.4	76.6	97.0	95.0	99.1

C.2 Detailed Results on Qwen2.5-VL-7B

Attention-path masking. Table 11 repeats the path-ratio sweep on Qwen2.5-VL-7B. The $(60\%, 70\%, 0\%)$ configuration retains 99.8% ATR, and adding 10% T2T masking gives 99.5%. Larger T2T ratios increasingly affect General and Reasoning: at 30%, GRR decreases to 98.1%, compared with 99.9% when T2T remains dense. Thus, this backbone also supports substantial compression of visually related paths while benefiting from a conservative text-side allocation.

Table 11: **Per-benchmark attention-path masking results on Qwen2.5-VL-7B after recovery.** All path masks use Taylor importance. V2V is swept first, followed by T2V with V2V fixed at 60%, and then T2T with the first two ratios fixed at 60% and 70%, respectively.

Setting	Path Mask (%) V2V	T2V	T2T	General GQA	VQA^{v2}	VQA^{ok}	Reasoning SQA	AI2D	MMStar	OCR VQA^{T}	VQA^{D}	OCRB	Grounding Ref	Ref+	RefG	ATR (%)	GRR (%)	OGR (%)
Dense Model	0	0	0	62.9	82.5	59.3	87.8	83.0	63.6	83.4	90.9	84.8	88.9	82.2	86.1	100.0	100.0	100.0
V2V	40	0	0	63.0	**82.5**	58.9	87.8	**83.1**	63.0	**83.7**	**91.1**	**85.0**	88.7	82.0	85.9	99.9	99.8	**100.0**
	50	0	0	**63.2**	**82.5**	58.9	87.7	82.9	**63.6**	83.6	90.7	84.7	**89.1**	82.2	**86.1**	**100.0**	**99.9**	**100.0**
	60	0	0	62.9	**82.5**	**59.3**	87.0	82.9	63.5	83.5	90.4	84.7	88.9	**82.4**	86.0	99.9	99.8	99.9
	70	0	0	62.9	82.3	59.0	**88.0**	82.6	62.9	83.0	89.5	84.9	88.4	81.5	85.7	99.5	99.7	99.4
T2V	60	50	0	63.2	**82.3**	59.0	87.4	83.2	**63.2**	**83.6**	**90.0**	83.9	88.8	82.0	85.7	99.7	99.8	99.6
	60	60	0	63.2	**82.3**	**59.1**	**87.7**	83.4	62.5	83.4	89.9	84.3	**89.0**	**82.1**	85.8	99.7	99.8	**99.7**
	60	70	0	**63.4**	82.2	58.7	87.5	**83.8**	63.0	83.0	89.3	**84.9**	88.9	82.0	**86.1**	**99.8**	**99.9**	99.6
	60	80	0	**63.4**	82.1	58.0	87.4	83.3	62.0	81.4	87.4	84.3	88.3	81.1	85.4	98.8	99.2	98.4
T2T	60	70	10	**63.4**	**82.1**	**58.1**	87.9	**83.7**	**62.7**	**82.7**	**89.0**	84.6	**88.7**	**82.0**	85.6	**99.5**	**99.6**	99.3
	60	70	20	63.1	82.0	56.2	**88.3**	83.5	61.4	**82.7**	88.9	84.8	88.6	**82.0**	**85.9**	99.0	98.7	**99.4**
	60	70	30	63.1	82.0	54.9	86.4	82.8	62.4	82.5	88.6	**85.7**	88.1	81.7	85.2	98.6	98.1	99.2

Modality-conditioned FFN masking. For the FFN experiments in Table 12, we fix the attention mask at $(60\%, 70\%, 0\%)$. This attention mask differs from the highlighted 10% T2T row in the attention-only sweep. Masking 40% of visual-side FFN executions while retaining all text-side executions gives 99.5% ATR, 99.7% GRR, and 99.2% OGR. Adding 20% text-side masking reduces ATR to 97.6%, with a larger decline in GRR than OGR.

The adopted configurations therefore share the same modality-aware structure across backbones, but not the same ratios: Qwen2.5-VL-7B uses more V2V and T2V masking and less visual-side FFN masking than LLaVA-OneVision-7B. This supports selecting the operating point for each backbone rather than transferring a single numerical mask configuration unchanged.

Table 12: **Per-benchmark FFN masking results on Qwen2.5-VL-7B after recovery.** Visual and Text give modality-conditioned channel masking ratios under the fixed $(60\%, 70\%, 0\%)$ attention-path mask.

Setting	Channel Mask (%) Visual	Text	General GQA	VQA^{v2}	VQA^{ok}	Reasoning SQA	AI2D	MMStar	OCR VQA^{T}	VQA^{D}	OCRB	Grounding Ref	Ref+	RefG	ATR (%)	GRR (%)	OGR (%)
$\text{MWOP}_{\text{Attn}}$	0	0	63.4	82.2	58.7	87.5	83.8	63.0	83.0	89.3	84.9	88.9	82.0	86.1	99.8	99.9	99.6
Visual Neuron	30	0	**63.6**	**82.3**	**58.7**	87.9	**83.5**	62.8	**82.6**	**89.1**	84.3	88.6	81.9	86.0	**99.6**	**99.9**	**99.3**
	40	0	63.4	**82.3**	58.5	87.5	83.0	**63.3**	81.9	88.7	**85.0**	88.6	81.8	85.9	99.5	99.7	99.2
	50	0	63.4	82.1	57.7	**88.3**	83.3	62.9	81.0	88.5	83.4	**89.0**	**82.2**	**86.1**	99.2	99.6	98.8
	60	0	63.3	82.0	57.3	88.0	82.8	62.5	80.0	88.4	83.9	88.8	82.1	85.9	98.9	99.2	98.6
Text Neuron	40	10	**63.2**	**81.8**	**55.4**	**88.0**	**83.2**	**62.4**	**81.8**	**88.6**	**85.0**	**88.7**	**81.8**	**85.6**	**98.9**	**98.6**	**99.1**
	40	20	**63.2**	81.0	50.3	87.0	82.8	**62.4**	81.0	88.5	84.0	88.2	81.3	**85.6**	97.6	96.7	98.5

Comparison with compression methods. Table 13 reports per-benchmark comparisons on Qwen2.5-VL-7B. MWOP retains 99.5% ATR with operation compression alone. Its token-

compressed variants, MWOP_Z and MWOP_P, retain 96.2% and 95.6% ATR, respectively. Compared with the corresponding token-only methods with post-training recovery in this table, MWOP_Z increases ATR from 94.9% to 96.2% and OGR from 92.4% to 95.1%. MWOP_P increases ATR from 88.5% to 95.6% and OGR from 83.0% to 93.8%. These results extend the benefit of combining operation and token compression to the second backbone.

Table 13: **Comparison of compression methods on Qwen2.5-VL-7B at approximately 40% retained LLM-side FLOPs.** Standalone MWOP at 67.4% retained FLOPs is included as a reference.

Method	Sparsity			General			Reasoning			OCR			Grounding			ATR	GRR	OGR
	Dep.	Wid.	Tok.	GQA	VQA^{v2}	VQA^{ok}	SQA	AI2D	MMStar	VQA^{T}	VQA^{D}	OCRB	Ref	Ref+	RefG	(%)	(%)	(%)
Qwen2.5-VL-7B	-	-	-	62.9	82.5	59.3	87.8	83.0	63.6	83.4	90.9	84.8	88.9	82.2	86.1	100.0	100.0	100.0
PyramidDrop†			✓	53.9	76.4	35.3	82.4	72.1	44.4	78.7	65.4	23.5	31.7	26.3	27.2	65.1	81.4	48.9
ZOO-Prune			✓	59.7	79.6	34.3	84.9	76.8	55.8	57.2	64.0	57.0	81.3	73.7	76.4	83.5	87.7	79.4
ShortV	✓			54.0	71.1	31.6	81.5	68.1	46.7	26.1	57.7	47.1	55.1	47.9	48.7	66.7	78.9	54.5
RedundancyLens		✓		57.0	78.1	34.4	83.2	75.9	55.8	72.8	80.2	69.5	78.0	70.1	72.2	86.0	86.2	85.7
PyramidDrop			✓	60.2	79.4	57.2	84.1	77.1	55.6	80.6	81.1	67.8	70.4	61.4	67.1	88.5	94.1	83.0
ZOO-Prune†			✓	62.7	80.8	56.8	85.5	81.0	61.0	75.7	81.1	71.7	86.2	79.0	83.3	94.9	97.4	92.4
ShortV†	✓			62.2	80.3	54.9	82.2	78.3	55.8	77.4	80.7	75.4	84.2	77.1	81.0	93.1	94.1	92.2
RedundancyLens†		✓		62.4	81.2	56.4	84.0	80.5	58.4	78.9	86.5	81.0	86.8	79.8	83.6	96.2	96.2	96.2
MWOP (ours)		✓		**63.4**	**82.3**	**58.5**	87.5	**83.0**	**63.3**	**81.9**	**88.7**	**85.0**	**88.6**	**81.8**	**85.9**	**99.5**	**99.7**	**99.2**
MWOP_Z (ours)		✓	✓	62.9	81.3	56.3	85.5	81.7	60.1	77.4	83.4	77.8	87.3	80.3	84.4	96.2	97.3	95.1
MWOP_P (ours)		✓	✓	62.5	81.3	56.6	**88.0**	80.5	59.6	81.2	85.7	75.6	84.0	76.1	81.9	95.6	97.4	93.8

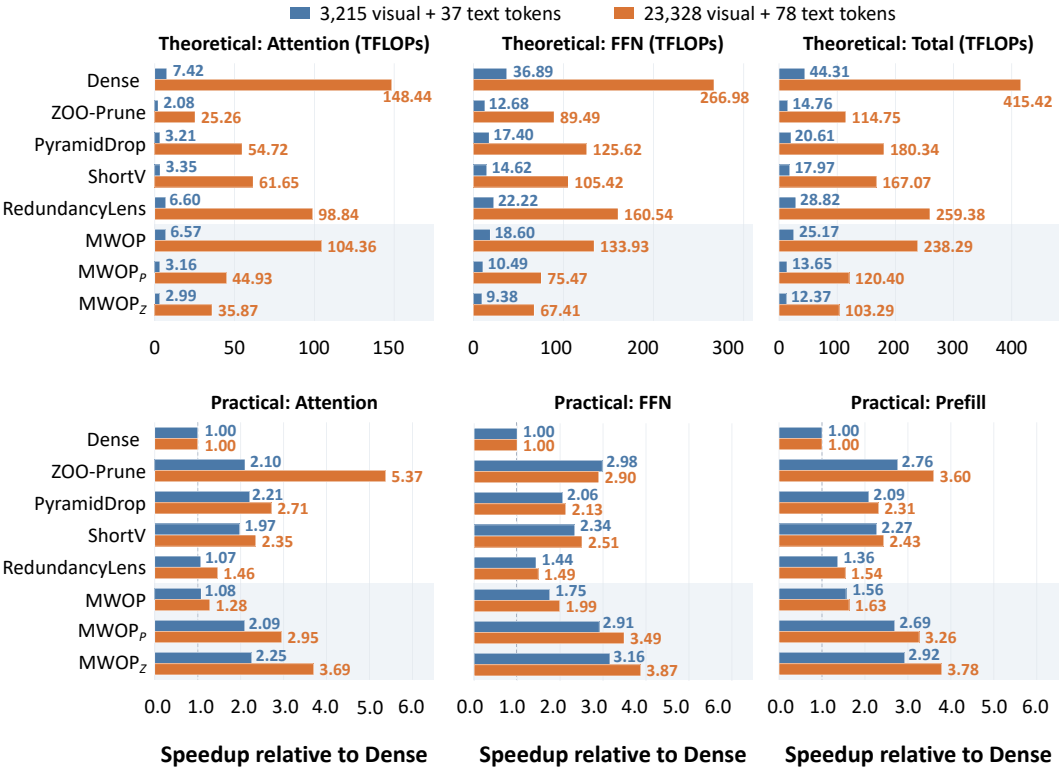


Figure 16: **Input-length dependence of computational cost and acceleration on LLaVA-OneVision-7B**. Blue bars use the mean single-image input from 12 benchmarks. Orange bars use the mean video input from MVBench, Video-MME, and NExT-QA. The **top** row reports theoretical attention, FFN, and total decoder FLOPs in TFLOPs. The **bottom** row reports measured attention, FFN, and prefill speedups. Lower FLOPs and higher speedups indicate greater efficiency under the corresponding input-length setting. Standalone ZOO-Prune and PyramidDrop use their default configurations, distinct from those in Table 5.

C.3 Effect of Input Sequence Length

We evaluate decoder prefill on an NVIDIA A100-SXM4-40GB GPU with batch size 1 and BF16 precision for both the single-image and video input profiles. Figures 16 and 17 compare computational cost and measured prefill acceleration at two representative input profiles for LLaVA-OneVision-7B and Qwen2.5-VL-7B, respectively. The blue bars use the mean visual and text token counts from

the 12 single-image benchmarks. The orange bars use the corresponding means from MVBench, Video-MME, and NExT-QA. These profiles represent shorter image inputs and longer video inputs. System tokens are excluded from the FLOPs inputs for both single-image profiles and the Qwen2.5-VL video profile (Section D.4). Latency measurements include system tokens following Section D.5. Each method and its dense reference use matching input profiles.

Longer visual sequences increase the attention share of the computation. For dense LLaVA-OneVision-7B, attention accounts for approximately 16.7% of total FLOPs at the image profile and 35.7% at the video profile. The corresponding shares for Qwen2.5-VL-7B are 13.2% and 19.5%. This shift is consistent with the quadratic growth of causal visual self-attention, compared with the linear token dependence of projections and FFNs (Section D.4). FFNs nevertheless remain the larger component at both profiles, leaving room for complementary savings by reducing both the number of tokens and the operations performed on each retained token.

The measured results show larger prefill speedups at the video profile. MWOP$_Z$ increases prefill speedup from $2.92\times$ to $3.78\times$ on LLaVA-OneVision-7B and from $1.98\times$ to $2.94\times$ on Qwen2.5-VL-7B. The gains for MWOP alone are more modest, increasing from $1.56\times$ to $1.63\times$ and from $1.28\times$ to $1.51\times$ on LLaVA-OneVision-7B and Qwen2.5-VL-7B, respectively.

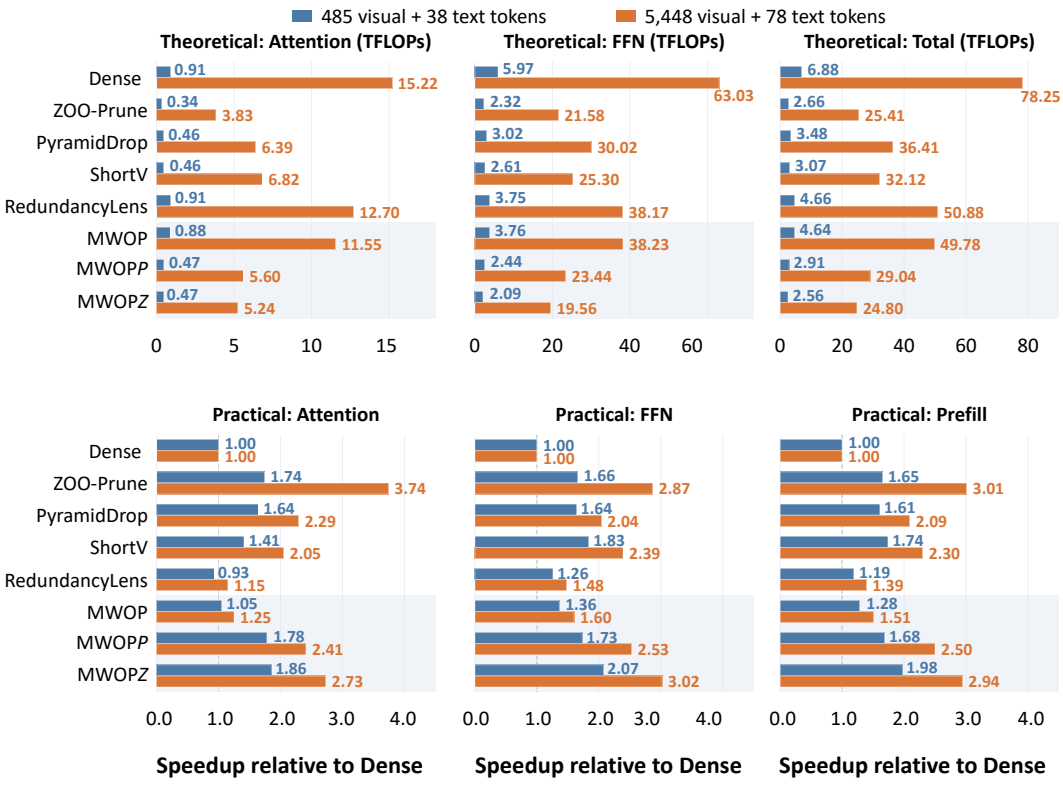


Figure 17: **Input-length dependence of computational cost and acceleration on Qwen2.5-VL-7B.** Blue bars use 485 visual and 38 text tokens for images, and orange bars use 5,448 visual and 78 text tokens for videos. The displayed text counts exclude system tokens, which are included in latency measurements. Standalone ZOO-Prune and PyramidDrop use their default configurations, distinct from those in Table 5.

D Implementation Details

We describe the implementation and experimental settings of MWOP. We first introduce the evaluated LMMs (Section D.1), specify the adopted operation and token pruning configurations (Section D.2), and present the sequential search and recovery procedure (Section D.3). We then detail the FLOPs calculation (Section D.4) and the prefill implementation and latency evaluation protocol (Section D.5). Finally, we summarize the post-training data composition (Section D.6) and train-

ing configuration (Section D.7). Together, these details specify the setup for pruning, recovery, and efficiency evaluation across the two backbones and their combinations with token pruning.

D.1 Details of LMMs

LLaVA-OneVision-7B connects a SigLIP vision encoder to Qwen2-7B-Instruct through a two-layer MLP projector that maps visual features into the language embedding space. Its AnyRes strategy represents high-resolution images using a global view and local crops.

Qwen2.5-VL-7B integrates a vision transformer supporting native dynamic resolution with Qwen2.5-7B. Its vision encoder uses window attention, while an MLP merger combines each spatial group of 2×2 patch features into a single visual token aligned with the language embedding space.

Both models process visual and textual tokens within an autoregressive language decoder. Table 14 summarizes their language-decoder configurations. The attention and FFN masks remain fixed throughout post-training and are applied during multimodal prefill at inference.

Table 14: Language-model backbone configurations of LLaVA-OneVision-7B and Qwen2.5-VL-7B.

Model	LLaVA-OneVision-7B	Qwen2.5-VL-7B
LLM Backbone	Qwen2-7B-Instruct	Qwen2.5-7B
Blocks	28	28
Query Heads	28	28
KV Heads	4	4
Head Dim	128	128
Hidden Dim	3584	3584
FFN Dim	18944	18944

D.2 Configuration Details of Our Method

Attention-path pruning. Figure 18 visualizes the layer-head masks for V2V, T2V, and T2T attention paths, with target pruning ratios of 40%, 60%, and 10%, respectively. Each mask specifies the retained modality-specific interactions across 28 decoder layers and 28 query heads. Pruning is applied to individual attention paths rather than entire attention heads.

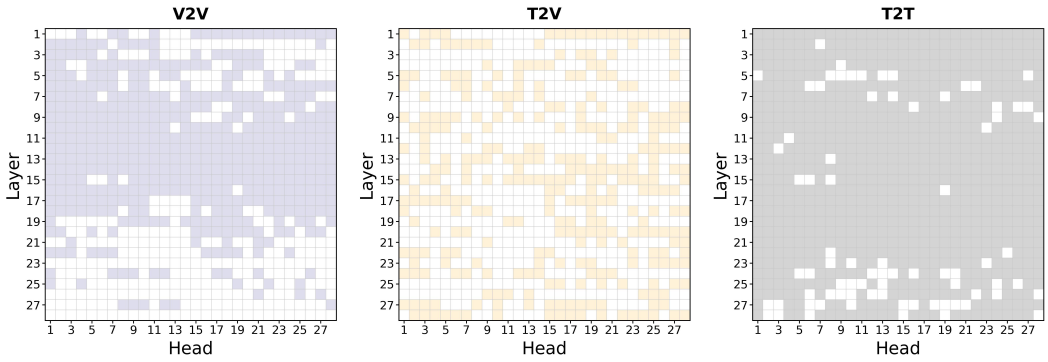


Figure 18: **Layer-head attention-path masks for LLaVA-OneVision-7B.** Rows and columns denote decoder layers and query heads, respectively, using one-based labels. Colored cells are retained, while white cells are masked. The masks remove 314/784 V2V paths (40.1%), 470/784 T2V paths (59.9%), and 78/784 T2T paths (9.9%). Each path is selected independently within a head.

Modality-conditioned FFN pruning. Table 15 reports the layer-wise FFN pruning configuration. We prune 50% of the visual token intermediate channels globally across all decoder layers while retaining all text token channels. Under the L27-first configuration, all visual token FFN channels in the final decoder layer (layer 27, zero-indexed) are masked first. The remaining pruning budget

is allocated across the other layers using the global Taylor ranking. The resulting allocation is nonuniform across layers, with substantially heavier pruning in the early and late layers than in the middle layers. Qwen2.5-VL-7B instead uses attention-path ratios of $(60\%, 70\%, 0\%)$ and FFN masking ratios of $(40\%, 0\%)$ for visual and text tokens, respectively. The layer-wise table below describes the LLaVA-OneVision configuration, with separate entries for visual and text tokens.

Table 15: **Layer-wise FFN masking for the adopted LLaVA-OneVision-7B configuration.** Layers are indexed from 0 to 27. Counts and percentages refer to masked modality-conditioned channel executions out of 18,944 channels per layer. The allocation masks 50% of visual-side channels globally across layers while retaining all text-side channels for the entire input sequence in every decoder layer.

Layer	Pruned FFN Channels		Layer	Pruned FFN Channels	
	Vision	Text		Vision	Text
0	16,473 (87.0%)	0 (0.0%)	14	980 (5.2%)	0 (0.0%)
1	14,690 (77.5%)	0 (0.0%)	15	2,533 (13.4%)	0 (0.0%)
2	14,722 (77.7%)	0 (0.0%)	16	7,231 (38.2%)	0 (0.0%)
3	12,337 (65.1%)	0 (0.0%)	17	9,414 (49.7%)	0 (0.0%)
4	10,083 (53.2%)	0 (0.0%)	18	12,317 (65.0%)	0 (0.0%)
5	9,088 (48.0%)	0 (0.0%)	19	15,706 (82.9%)	0 (0.0%)
6	126 (0.7%)	0 (0.0%)	20	15,423 (81.4%)	0 (0.0%)
7	9 (0.0%)	0 (0.0%)	21	16,450 (86.8%)	0 (0.0%)
8	18 (0.1%)	0 (0.0%)	22	16,345 (86.3%)	0 (0.0%)
9	637 (3.4%)	0 (0.0%)	23	17,375 (91.7%)	0 (0.0%)
10	2 (0.0%)	0 (0.0%)	24	18,010 (95.1%)	0 (0.0%)
11	1 (0.0%)	0 (0.0%)	25	17,873 (94.3%)	0 (0.0%)
12	11 (0.1%)	0 (0.0%)	26	18,346 (96.8%)	0 (0.0%)
13	72 (0.4%)	0 (0.0%)	27	18,944 (100.0%)	0 (0.0%)

Combination with token pruning. For MWOP combined with PDrop, pruning occurs at entry to layers 7, 14, and 21 (zero-based), after 7, 14, and 21 blocks have been executed. Cumulative visual token retention ratios are approximately 54.4%, 29.6%, and 16.1%, respectively, measured relative to the original visual token count. Both MWOP_P and MWOP_Z retain approximately 50% of the original visual tokens on average across all decoder layers. In both combinations, the original position IDs of the retained tokens are preserved throughout inference.

D.3 Algorithm Implementation

Algorithm 1 searches V2V, T2V, and T2T attention paths, followed by visual and text FFN channels. Attention importance is estimated once on the initial model $\mathcal{M}_0$. FFN importance for both modalities is estimated once on $\mathcal{M}_0$ with the attention masks selected through the T2T stage applied. At each stage, a training-free ratio sweep evaluates masked models on the selection set $\mathcal{D}_{\text{sel}}$, which contains 1,000 samples, to identify candidates for post-training recovery. The candidate set is expanded according to the observed performance changes. Every candidate is evaluated with its cumulative masks applied to $\mathcal{M}_0$ and independently recovered for one epoch on $\mathcal{D}_{\text{rec}}$. The masks remain fixed throughout recovery, using the settings in Section D.7.

Each recovered candidate is evaluated on all samples in the full benchmark suite $\mathcal{B}_{\text{full}}$. The candidate whose ATR is closest to 100% provides the anchor mask for the next stage. Stages pass the selected masks forward, while all candidate runs start from the same initial weights $\mathcal{M}_0$. We store all recovered candidates in $\mathcal{H}$ in search order and complete every stage through FFN_T. The recorded full-benchmark ATR scores are reused for final selection. Starting from the final-stage anchor, we revisit candidates in search order and replace the current selection only when a candidate has both higher full-benchmark ATR and lower theoretical FLOPs. Here, $F(\mathcal{M})$ denotes the theoretical decoder FLOPs of candidate $\mathcal{M}$, computed using the counting convention in Section D.4.

Algorithm 1 Sequential Redundancy Search with Post-Training Recovery

Input: Dense model $\mathcal{M}_0$, calibration data $\mathcal{D}_{\text{cal}}$, recovery data $\mathcal{D}_{\text{rec}}$, selection set $\mathcal{D}_{\text{sel}}$ (1,000 samples), full benchmark suite $\mathcal{B}_{\text{full}}$, ratio grid $\mathcal{R} = \{0, 0.1, \ldots, 1.0\}$.
Order: $\mathcal{U} = [\text{V2V}, \text{T2V}, \text{T2T}, \text{FFN}_V, \text{FFN}_T]$.

$\Omega \leftarrow \varnothing, \quad \mathcal{H} \leftarrow []$
$\mathcal{I} \leftarrow \textsc{TaylorImportance}(\mathcal{M}_0, \mathcal{D}_{\text{cal}}, \text{Attn})$
for $u \in \mathcal{U}$ **do**
  if $u = \text{FFN}_V$ **then**
    $\mathcal{I} \leftarrow \textsc{TaylorImportance}(\text{Mask}(\mathcal{M}_0, \Omega), \mathcal{D}_{\text{cal}}, \text{FFN})$ ▷ Both modalities, once
  end if
  for $\rho \in \mathcal{R}$ **do**
    $\Omega_\rho \leftarrow \textsc{AddMask}(\Omega, u, \mathcal{I}_u, \rho)$
    $A_{\text{TF}}(\rho) \leftarrow \textsc{EvaluateATR}(\text{Mask}(\mathcal{M}_0, \Omega_\rho), \mathcal{D}_{\text{sel}})$
  end for
  $\mathcal{C} \leftarrow \textsc{DropCandidates}(\mathcal{R}, A_{\text{TF}})$
  while $\mathcal{C}$ has untested candidates **do**
    Select an untested $\rho \in \mathcal{C}$
    $\mathcal{M}_\rho^{\text{REC}} \leftarrow \textsc{PostTrain}(\text{Mask}(\mathcal{M}_0, \Omega_\rho), \mathcal{D}_{\text{rec}})$ ▷ One epoch, fixed masks
    $A_{\text{full}}(\mathcal{M}_\rho^{\text{REC}}) \leftarrow \textsc{EvaluateATR}(\mathcal{M}_\rho^{\text{REC}}, \mathcal{B}_{\text{full}})$
    Expand $\mathcal{C} \subseteq \mathcal{R}$ using this stage's performance changes
  end while
  Append this stage's recovered candidates to $\mathcal{H}$ in search order
  $\rho^{\text{anc}} \leftarrow \arg\min_{\rho \in \mathcal{C}} |A_{\text{full}}(\mathcal{M}_\rho^{\text{REC}}) - 100|$
  $\Omega \leftarrow \Omega_{\rho^{\text{anc}}}, \quad \mathcal{M}_{\text{anc}} \leftarrow \mathcal{M}_{\rho^{\text{anc}}}^{\text{REC}}$
end for
$\mathcal{M}^* \leftarrow \mathcal{M}_{\text{anc}}$ ▷ Final-stage anchor
for $\mathcal{M} \in \mathcal{H}$ in search order **do**
  if $A_{\text{full}}(\mathcal{M}) > A_{\text{full}}(\mathcal{M}^*)$ and $F(\mathcal{M}) < F(\mathcal{M}^*)$ **then**
    $\mathcal{M}^* \leftarrow \mathcal{M}$
  end if
end for
Output: Compressed model $\mathcal{M}^*$ and $\textsc{GetAllMasks}(\mathcal{M}^*)$.

D.4 FLOPs Calculation Details

Input profiles. For theoretical FLOPs, we use representative input profiles with V_0 visual tokens and T text tokens, excluding system tokens. The single-image profiles are $(V_0, T) = (3215, 19)$ for LLaVA-OneVision-7B and $(485, 38)$ for Qwen2.5-VL-7B. The Qwen2.5-VL video profile uses $(V_0, T) = (5448, 78)$ under the same FLOPs counting convention.

Counting convention. We count two FLOPs per multiply-accumulate and report matrix-multiplication costs in TFLOPs (10^{12} FLOPs). At decoder layer ℓ, the counted sequence length is $N_\ell = V_\ell + T$, where V_ℓ denotes the remaining visual tokens. System tokens are excluded from both projection costs and attention-pair counts. MWOP preserves token counts, whereas token pruning reduces V_ℓ at the layer boundaries specified by each token-pruning method.

Attention computation. Both decoders have $L = 28$ layers, $H = 28$ query heads, $H_{\mathrm{KV}} = 4$ key/value heads, head dimension $d_h = 128$, hidden width $d = 3584$, and key/value width $d_{\mathrm{KV}} = 512$. MWOP retains all Q/K/V/O projection dimensions and the native grouped-query attention mapping. Within the counted input sequence, let $\mathcal{A}_{\ell,h}(i)$ denote the causal keys retained for query i in head h, including unpruned interactions with any non-system text preceding the visual span. The mean retained pair count per head and the attention FLOPs at layer ℓ are

$$a_\ell = \frac{1}{H} \sum_{h=1}^{H} \sum_{i=1}^{N_\ell} \left|\mathcal{A}_{\ell,h}(i)\right|, \qquad F_{\mathrm{attn},\ell} = 4N_\ell d(d + d_{\mathrm{KV}}) + 4da_\ell.$$

The first term accounts for the four projections. The second accounts for QK products and attention-value multiplication, each costing $2da_\ell$. Dense causal attention gives $a_\ell = N_\ell(N_\ell + 1)/2$, including the diagonal. For MWOP, the pair count follows the actual layer-head masks, which need not distribute the global pruning ratio uniformly across layers.

FFN computation and total FLOPs. Let $m = 18944$ be the full intermediate width and $k_{V,\ell}$ the retained visual channel count. With alignment b, the executed visual width and FFN cost are

$$\widetilde{m}_{V,\ell} = \min\left(m, b\left\lceil \frac{k_{V,\ell}}{b} \right\rceil\right), \qquad F_{\mathrm{FFN},\ell} = 6d\big(V_\ell \widetilde{m}_{V,\ell} + Tm\big).$$

The factor six counts the gate, up, and down projections. Text tokens retain the full FFN width in the evaluated configurations. The standalone MWOP implementations and the Qwen2.5-VL combinations use $b = 64$, while the LLaVA-OneVision combinations use $b = 1$. Zero retained channels therefore incur no visual FFN projection cost. The decoder total includes all attention and FFN projections, retained attention interactions, and the last-token language-model head:

$$F_{\mathrm{total}} = \sum_{\ell=0}^{L-1} \left(F_{\mathrm{attn},\ell} + F_{\mathrm{FFN},\ell}\right) + 2dU,$$

where U is the vocabulary size, 152128 for LLaVA-OneVision and 152064 for Qwen2.5-VL. The attention and FFN columns exclude the language-model head. The single-image profiles above yield dense totals of 44.31 and 6.88 TFLOPs for LLaVA-OneVision and Qwen2.5-VL, respectively.

These counts include FFN alignment padding but exclude normalization, RoPE, softmax, elementwise operations, tensor movement, and excess work in boundary tiles. Comparison methods are counted using their actual token lengths, projection shapes, attention patterns, and FFN widths. In particular, ShortV layers that bypass visual queries still compute the visual K/V projections required by text queries. Online token scoring and ranking are excluded from both FLOPs and latency comparisons under the decoder-only evaluation protocol. This choice is conservative for MWOP, whose pruning masks are static and incur no online selection overhead at inference, while some token-pruning baselines require additional runtime scoring or ranking.

D.5 Efficient Inference Implementation

MWOP reduces decoder-prefill computation through path-sparse attention and compact visual FFNs.

Path-sparse attention. For the retained causal key set $\mathcal{A}_{\ell,h}(i)$ in the actual input sequence, including system tokens, the attention output for query i is computed as

$$o_i = \frac{\sum_{j\in\mathcal{A}_{\ell,h}(i)} \exp(s_{ij})v_j}{\sum_{j\in\mathcal{A}_{\ell,h}(i)} \exp(s_{ij})}, \qquad s_{ij} = q_i^\top k_j/\sqrt{d_h}.$$

The Triton kernel traverses only the retained key intervals and maintains a single online softmax state across them. This implements pre-softmax path masking while skipping pruned QK and attention-value products, without materializing the attention matrix. Causal and interval-end masks handle boundary tiles. Operation pruning preserves GQA mapping, projection widths, and KV-cache dimensions.

Modality-conditioned FFNs. The retained rows of the gate and up projections, along with the corresponding columns of the down projection, are packed into compact visual weights. Visual tokens use these weights, while text tokens use the full FFN. Empty visual channel sets yield a zero FFN update. The packed weights require additional storage alongside the full text weights. To reduce intermediate data movement, the optimized Qwen implementation fuses normalization and modality splitting, gated activation, and output merging with residual addition.

Latency evaluation. We measure batch-one BF16 prefill latency on an NVIDIA A100-SXM4-40GB using model weights and synthetic embedding-table inputs. Timing inputs include system tokens. For single-image measurements, these add 18 tokens for LLaVA-OneVision-7B and 11 for Qwen2.5-VL-7B to the profiles in Section D.4. Dense and pruned models use matching input lengths within each timing setting. The dense baseline uses FlashAttention-2. Qwen2.5-VL uses its native multimodal rotary operator with identical linear temporal and spatial coordinates. We complete compilation, weight packing, kernel tuning, and graph capture before timing, with kernel configurations fixed for each measured shape. After warmup, CUDA events measure CUDA Graph replay latency. The reported latency is the median of five window means, each computed from ten replay batches. Attention and FFN latencies are obtained by summing separately measured layer-module latencies, while total prefill latency is measured independently. Speedup is the ratio of dense to pruned total prefill latency for each input setting on the same hardware.

D.6 Post-training data composition

For post-training recovery, we use a mixture of 282,293 samples developed as part of a separate work. This specific training mixture has not yet been publicly released. Table 16 summarizes its source datasets and sample counts, covering general visual understanding, grounding, reasoning, OCR, and language-only instruction following. The recovery training data do not overlap with any of the evaluation sets used in this paper for single-image, multi-image, or video understanding.

Table 16: Composition of the post-training data mixture.

Dataset	Count	Category
COCO (Lin et al., 2014)	103,909	General
VG (Krishna et al., 2017)	55,504	Grounding
GQA (Hudson & Manning, 2019)	42,777	Reasoning
OCRVQA (Mishra et al., 2019)	27,346	General + OCR
ChartQA (Masry et al., 2022)	10,000	Reasoning + OCR
DVQA (Kafle et al., 2018)	10,000	Reasoning + OCR
DocVQA (Mathew et al., 2021)	10,000	OCR
AI2D (Kembhavi et al., 2016)	7,508	Reasoning
LLaVAR (Zhang et al., 2023)	6,785	OCR
TQA (Kembhavi et al., 2017)	6,501	Reasoning
ShareGPT (Chiang et al., 2023)	1,000	Language-only
TextVQA (Singh et al., 2019)	963	General + OCR
Total	**282,293**	**N/A**

D.7 Training Configuration

For each LLaVA-OneVision-7B candidate, we apply its cumulative pruning masks to $\mathcal{M}_0$ and independently finetune for one epoch on the mixture in Table 16. The selected configuration is given in Section D.2. Only the multimodal projector and language-model LoRA adapters are updated. The vision encoder and base language-model weights remain frozen. LoRA uses rank 128, $\alpha = 256$, and dropout 0.05. We use AdamW with a learning rate of 1×10^{-5} for both trainable components, zero weight decay, cosine scheduling, and a warmup ratio of 0.03. Training runs on eight A100s, with a per-device batch size of 1 and 32 gradient accumulation steps. The effective batch size is 256. Training uses BF16, gradient checkpointing, SDPA, and DeepSpeed ZeRO-2.

E Details of Comparison Methods

FastV (Chen et al., 2024a). FastV prunes visual tokens after shallow language-model layers using text-to-vision attention scores. It discards low-scoring tokens from subsequent layers, reducing sequence length and the associated attention and FFN computation.

PyramidDrop (Xing et al., 2024). PyramidDrop progressively prunes visual tokens at multiple depths of the language model. Instruction-guided attention determines which tokens to retain, allocating larger token budgets to shallow layers and smaller budgets to deeper layers.

VisionZip (Yang et al., 2025). VisionZip compresses visual representations before language-model processing by retaining dominant tokens and constructing contextual tokens. It combines attention-based selection with similarity-based aggregation to preserve salient visual content and complementary contextual information across the image.

DivPrune (Alvar et al., 2025). DivPrune formulates visual token selection as a Max-Min Diversity Problem. It greedily maximizes the minimum pairwise distance among retained tokens, reducing redundant representations while maintaining broad coverage of diverse visual content.

DART (Wen et al., 2025). DART performs duplication-aware visual token pruning using visual and textual pivot tokens. It retains tokens with low similarity to these pivots, emphasizing complementary visual information rather than relying solely on attention-based importance scores.

VisPruner (Zhang et al., 2025). VisPruner combines visual importance and diversity for token pruning without text-guided attention. It first selects important tokens using visual attention, then supplements them with diverse tokens through similarity-based duplicate removal.

FEATHER (Endo et al., 2025). FEATHER combines token selection based on RoPE-free attention scores with uniform spatial sampling. Its two-stage pruning strategy preserves spatial coverage during early selection and further reduces visual tokens in later layers.

MMTok (Dong et al., 2026). MMTok formulates visual token selection as multimodal coverage maximization. It jointly considers text-visual relevance and visual-visual coverage, greedily selecting tokens that represent the image while preserving information relevant to the input query.

PruMerge+ (Shang et al., 2025). PruMerge+ combines attention-guided token selection, similarity-based merging, and spatial sampling. Selected tokens aggregate similar visual features, while additional spatially sampled tokens preserve broader image coverage beyond regions highlighted by attention scores in the initial token selection step.

ZOO-Prune (Kim et al., 2026). ZOO-Prune estimates visual token sensitivity at the projector stage using zeroth-order perturbations. It combines sensitivity with diversity to retain informative tokens before language-model processing, without requiring gradient backpropagation.

ShortV (Yuan et al., 2025). ShortV identifies low-contribution layers through output divergence and freezes visual hidden states in those layers. It skips visual attention updates and FFN computation while preserving visual keys and values for text queries.

RedundancyLens (Li et al., 2025). RedundancyLens combines Hollow Attention with Probe-Activated Dynamic FFN to reduce visual computation. It restricts visual self-attention to local neighborhoods and uses sampled visual tokens to select active FFN channels in designated layers.

HalfV (Shi et al., 2026). HalfV distinguishes intrinsic visual redundancy from architecture-dependent saturation. It combines token pruning with backbone-aware reductions in later visual processing, adapting compression to how visual representations evolve across the language model.

DOP$_{\mathrm{V}}$ (Liu et al., 2025). DOP$_{\mathrm{V}}$ combines VisPruner-based visual token reduction with depth-wise operation pruning across attention and FFNs. Under a computational budget, it uses output divergence to select different visual processing depths for attention and FFNs.

YOPO (Zhang et al., 2024a). YOPO reduces visual computation through head pruning, local visual attention, late-layer operation removal, and FFN channel pruning. These reductions establish a computation pattern shared across inputs rather than selected dynamically for each sample.

Skip-Vision (Zeng et al., 2025). Skip-Vision allows selected visual tokens to bypass FFN transformations while remaining available to attention. It combines token aggregation with learned summarization and removes skipped-token cache entries after prefilling to reduce decoding overhead.

F Details of Benchmarks

F.1 Single-image benchmarks

GQA (Hudson & Manning, 2019). GQA evaluates compositional visual reasoning using scene-graph annotations from Visual Genome. It requires models to combine information about object attributes and relationships to answer questions grounded in the visual scene.

VQA-v2 (Goyal et al., 2017). VQA-v2 is a balanced, open-ended visual question answering benchmark designed to reduce language priors. It encourages models to examine visual evidence and produce image-grounded answers rather than rely primarily on question patterns.

OK-VQA (Marino et al., 2019). OK-VQA evaluates knowledge-based visual question answering, where image content is often insufficient. Models must connect visual observations with external or commonsense knowledge to answer questions that extend beyond directly observable information.

ScienceQA (Lu et al., 2022). ScienceQA evaluates multimodal scientific reasoning across natural, language, and social sciences. Its image-based subset requires models to combine visual evidence with question context and relevant scientific knowledge to identify the correct answer.

AI2D (Kembhavi et al., 2016). AI2D evaluates diagram understanding through questions involving labels, symbols, visual structures, and spatial relations. Models must connect these elements and interpret their relationships to reason about the information conveyed by a diagram.

MMStar (Chen et al., 2024b). MMStar evaluates multimodal understanding with curated questions that require visual evidence. Its samples span multiple visual and reasoning capabilities and are selected to reduce language-only shortcuts and potential data leakage. We include it in the Reasoning category when reporting category-level performance retention in our experiments.

TextVQA (Singh et al., 2019). TextVQA evaluates scene-text understanding by requiring models to recognize and reason over text in natural images. Answering its questions involves integrating textual content with the surrounding visual context rather than performing text recognition alone.

DocVQA (Mathew et al., 2021). DocVQA evaluates question answering over document images, emphasizing OCR, layout understanding, and information localization. Models must locate relevant document regions and interpret their textual content in the context of the question.

OCRBench (Liu et al., 2024). OCRBench evaluates OCR-related capabilities of MLLMs across text recognition, document understanding, information extraction, and related tasks. It assesses the accuracy of text perception and the ability to use textual information for downstream understanding.

RefCOCO (Kazemzadeh et al., 2014; Yu et al., 2016). RefCOCO evaluates referring expression comprehension by requiring models to localize objects described in natural language. Models must match object attributes and relationships to the expression to distinguish the target.

RefCOCO+ (Kazemzadeh et al., 2014; Yu et al., 2016). RefCOCO+ disallows location words during annotation, emphasizing appearance-based referring expressions. Models must identify the target from visual attributes such as color, shape, and clothing.

RefCOCOg (Mao et al., 2016; Yu et al., 2016). RefCOCOg contains longer and more descriptive referring expressions, testing fine-grained and context-rich visual grounding. Models must integrate multiple cues, including object attributes and relationships, to identify the intended target.

F.2 Multi-image benchmarks

Q-Bench2 (Zhang et al., 2024b). Q-Bench2 evaluates low-level visual perception through questions about image pairs. Models must jointly inspect and compare visual attributes, such as clarity, lighting, and distortions, to identify subtle differences and similarities between images.

Mantis-Eval (Jiang et al., 2024). Mantis-Eval evaluates multi-image reasoning and co-reference through manually curated multiple-choice and short-answer questions. Models must compare visual content and combine evidence across images to answer questions requiring joint interpretation.

BLINK (Fu et al., 2024). BLINK evaluates core visual perception through tasks such as relative depth estimation, visual correspondence, and multi-view reasoning. It requires models to interpret fine-grained visual and spatial cues from single or multiple images.

F.3 Video benchmarks

MVBench (Li et al., 2024b). MVBench evaluates temporal video understanding through 20 tasks spanning perception and reasoning. Models must combine evidence across frames to recognize actions, track state changes, and understand event sequences rather than rely on a single frame.

Video-MME (Fu et al., 2025). Video-MME evaluates video understanding across diverse domains and durations, from short clips to hour-long videos. Tasks assess visual perception, temporal reasoning, contextual understanding, and information integration across time.

NExT-QA (Xiao et al., 2021). NExT-QA evaluates video question answering with an emphasis on causal and temporal action reasoning. Models must understand event order, object interactions, and the reasons behind actions, moving beyond recognition and description of visible content.