QCPRUNER: QUERY-CONDITIONED POPULATION COVERAGE FOR VISUAL TOKEN PRUNING

Anonymous authors
Paper under double-blind review

ABSTRACT

The high visual-token load in multimodal large language models (MLLMs) motivates aggressive training-free pruning, but under a prescribed final token budget, deciding which visual evidence survives is critical to preserving downstream performance. Existing methods rank tokens, diversify selected subsets, or combine relevance with coverage, while the direct conditioning of visual-to-visual population coverage by contextual per-visual query utility remains underexplored. We introduce **QCPruner**, a training-free method that uses contextual query utility to condition population-level visual-token coverage. Using keyword-matched query anchors, QCPruner fuses two cross-modal cues into utility and incorporates it into a visual-affinity-based coverage objective, where utility modulates representative suitability and, in the full formulation, also weights target importance. The resulting nonnegative facility-location objective is monotone and submodular, retains the standard $(1 - 1/e)$ greedy guarantee, and requires no model training or parameter updates. Across LLaVA-1.5, LLaVA-NeXT, LLaVA-Video, and Qwen2.5-VL, QCPruner consistently preserves strong downstream performance under matched token budgets, with the clearest gains appearing under aggressive compression. At 32 of 576 tokens on LLaVA-1.5-7B, it retains 96.1% of unpruned performance, versus 93.9% for the strongest evaluated baseline. At 256 of 1296 tokens on Qwen2.5-VL-7B, the corresponding values are 96.7% and 92.5%. Role-wise ablations show that representative-side query conditioning accounts for most of the gain over visual-only coverage, while target-side weighting provides a smaller, setting-dependent refinement.

1 INTRODUCTION

MLLMs encode images or videos as visual-token prefixes and process them with a user instruction in a Transformer language model Vaswani et al. (2017); Liu et al. (2023; 2024a;b); Wang et al. (2024); Bai et al. (2025). A standard LLaVA-1.5 image yields 576 visual tokens, while high-resolution and video inputs generate thousands. These prefixes increase prefill computation and activation memory, motivating training-free pruning. Pre-decoder methods avoid processing discarded tokens throughout the decoder but rely on signals formed before decoder-side contextual cross-modal reasoning Yang et al. (2025); Alvar et al. (2025); Dong et al. (2026); Fang et al. (2026). In-decoder methods obtain contextual cross-modal states only after preceding layers process the full sequence Chen et al. (2024a); Zhang et al. (2024). In this work, we focus on high-fidelity pruning under a prescribed final visual-token budget: methods are compared primarily by downstream performance retention at matched K, while computational cost is characterized separately.

Independent attention- or relevance-based ranking can retain redundant neighboring patches Chen et al. (2024a); Zhang et al. (2024). Redundancy- and diversity-oriented methods preserve visual breadth, but visual redundancy alone need not identify evidence required by the current instruction Yang et al. (2025); Wen et al. (2025); Alvar et al. (2025). Conditional DPPs couple instruction relevance with selected-set diversity, yet focus on the retained subset and its internal interactions rather than explicit representation of the broader visual population Zhang et al. (2025). Coverage objectives measure representation against a broader target population. MMTok instantiates this approach with coverage over text and visual target populations Dong et al. (2026). It is query-aware and population-relative, but introduces the query as a separate text-side target rather than directly

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.

conditioning visual-to-visual coverage. We therefore ask: how can contextual query evidence directly guide population-level visual coverage rather than only scoring individual tokens?

QCPruner addresses this question by integrating contextual per-visual query utility into a visual population-coverage objective. The key mechanism is to condition representative suitability on the query, so that selected tokens are rewarded for being both visually representative and relevant to the current instruction. The full formulation additionally uses the same utility to weight target importance, emphasizing query-relevant evidence within the population being covered. Visual affinity links targets and representatives, yielding a unified query-conditioned coverage objective. Specifically, QCPruner derives contextual utility from keyword-matched query anchors and complementary cross-modal cues, then performs coverage-aware greedy selection under a fixed token budget.

Across four backbones spanning standard images, high-resolution images, and video, QCPruner consistently achieves strong retention under matched token budgets, with increasingly clear gains as compression becomes more aggressive. It retains 96.1% at 32/576 LLaVA-1.5 tokens, 96.7% at 256/1296 Qwen2.5-VL tokens, and 96.3% at 1024/10816 LLaVA-Video tokens. Matched-budget comparisons further give higher average relative performance than MMTok across all three image backbones. Accordingly, our primary claims concern performance retention at matched final token budgets. We characterize computational cost separately and do not claim compute- or latency-matched dominance over earlier-pruning methods.

Our contributions are summarized as follows:

- We introduce query-conditioned population coverage for visual-token pruning, directly integrating contextual per-visual query utility into visual-to-visual coverage. Representative utility conditions which tokens can effectively cover the visual population, while target-side utility provides a complementary weighting of which evidence deserves coverage. The resulting nonnegative objective preserves the standard greedy guarantee.
- Without training the MLLM or a pruning module, we derive contextual utility from keyword-matched prompt positions by combining hidden-state similarity and pre-softmax query–key alignment.
- We validate QCPruner across four MLLM backbones and twelve budget settings spanning standard images, high-resolution inputs, and video. Controlled role-wise ablations show that representative-side conditioning provides the dominant gain over visual-only coverage, while target-side weighting offers a smaller, setting-dependent refinement.

2 Related Work

MLLM visual-token layouts. MLLMs project features from visual encoders such as CLIP or SigLIP into a language model and process them jointly with textual instructions Radford et al. (2021); Zhai et al. (2023); Liu et al. (2023; 2024a); Zhu et al. (2024); Chen et al. (2024c); Yin et al. (2024). Their visual-token layouts range from fixed image grids to high-resolution tiles, repeated video-frame grids, and model-specific arrangements Liu et al. (2024b); Bai et al. (2025). These layouts differ in token count, spatial organization, and structural boundaries, leading to different token scales and structural constraints for visual-token pruning.

Training-free visual-token pruning. Training-free methods remove or merge visual tokens using attention, saliency, similarity, redundancy, or cross-modal relevance Chen et al. (2024a); Zhang et al. (2024); Shang et al. (2025); Yang et al. (2025); Jeddi et al. (2025); Ye et al. (2025); Wen et al. (2025); Fang et al. (2026). Pre-decoder methods reduce the sequence before language-model processing, while in-decoder methods exploit contextual cross-modal states formed by preceding layers. Many ranking-based approaches score tokens independently, which can retain individually salient but mutually redundant visual evidence. This motivates subset-level objectives that explicitly consider relations among retained tokens.

Subset-level visual-token selection. Subset-level methods model interactions among selected tokens or how well they represent a broader visual population. DivPrune optimizes visual diversity, while CDPruner combines instruction relevance with conditional-DPP selection to diversify the retained subset Alvar et al. (2025); Zhang et al. (2025). MMTok instead combines text–vision

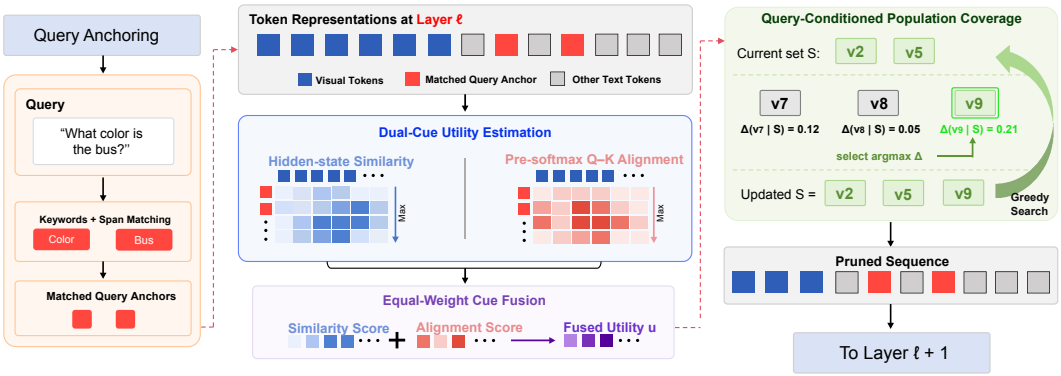

Figure 1: Overall framework of QCPruner. At the post-layer-ℓ hook, matched query anchors are used to compute hidden-state cosine similarity from the layer output and pre-softmax query–key alignment recorded within the same layer. Their fused utility conditions visual-affinity-based population coverage: representative utility modulates how effectively a selected token can cover other visual evidence, while target utility provides an additional weighting of the evidence being covered. Greedy marginal gain selects K representatives to maximize coverage of the full candidate population.

and vision–vision facility-location coverage, making selection both query-aware and population-relative Dong et al. (2026). SCOPE Deng et al. (2025) combines token saliency with set-coverage gains to jointly preserve salient and semantically complete visual evidence, while MoB Li et al. (2025) formulates prompt alignment and visual preservation as a bi-objective covering problem. QCPruner builds on population-level coverage but differs by deriving contextual per-visual query utility at an in-decoder hook and directly integrating it into visual-to-visual population coverage, rather than introducing the query as a separate target population or only conditioning interactions within the retained subset.

3 Method

QCPruner consists of three steps: keyword-matched query-anchor construction, dual-cue contextual utility estimation, and query-conditioned population-coverage selection under a fixed token budget. Figure 1 summarizes the framework.

3.1 Problem Setup

Let $\mathcal{V} = (v_1, \ldots, v_{N_v})$ be the ordered candidate patch tokens and $\mathcal{T} = (t_1, \ldots, t_{L_t})$ the prompt tokens. At the post-layer-ℓ hook, where ℓ denotes the zero-based decoder-layer index, contextual states $\mathbf{h}_i^{v,\ell}, \mathbf{h}_j^{t,\ell} \in \mathbb{R}^d$ are output by layer ℓ and pruning precedes layer $\ell + 1$. Model-specific separators and other non-patch structural tokens are excluded from $\mathcal{V}$ and remain in the sequence. We select indices

$$\mathcal{K} \subseteq \{1, \ldots, N_v\}, \qquad |\mathcal{K}| = K, \tag{1}$$

We treat K as the prescribed final visual-token budget. Each v_i can act as a *representative*, whose query utility and visual affinity determine how effectively it represents other tokens, or as a *target*, whose utility weights the importance of preserving its evidence.

3.2 Keyword-Matched Query Anchors

To avoid dilution by system scaffolding and generic instruction text, KeyBERT Grootendorst (2020) extracts at most six ranked content unigrams from the user-visible prompt. Each keyword is retokenized with the MLLM tokenizer and matched as a contiguous subsequence at every occurrence in the prompt; the union of the matched prompt-token positions, including all subwords, forms $\mathcal{I}_{\text{anc}}$. If no keyword matches, we deterministically fall back to non-template user-content positions after ex-

cluding special tokens and visual placeholders. These positions index contextualized MLLM states for subsequent scoring.

3.3 Dual-Cue Query-Conditioned Utility Estimation

At the post-layer-ℓ hook, the semantic cue uses contextual states output by decoder layer ℓ, whereas the alignment cue uses the pre-softmax query and key projections recorded while executing that same layer. We omit the layer index ℓ from subsequent notation. For each v_i, the following two scores are proxies for compatibility with at least one anchor in $\mathcal{I}_{\text{anc}}$.

Similarity-based semantic cue. The similarity cue uses contextual hidden-state cosine proximity as a representation-level proxy. We define the relevance of visual token v_i as

$$r_i^{\text{sim}} = \max_{j \in \mathcal{I}_{\text{anc}}} \cos\left(\mathbf{h}_j^t, \mathbf{h}_i^v\right). \tag{2}$$

Max aggregation retains a high score when any matched anchor aligns with a token and avoids diluting strong anchor-specific relevance through averaging.

Query–key alignment cue. Let $\mathbf{q}_j^{(h)}$ and $\mathbf{k}_i^{(h)}$ be the query and key vectors for anchor t_j and visual token v_i at head h, where H is the number of heads and d_h is the dimensionality of each head. We use the head-averaged, scaled dot product before softmax:

$$A_{j,i} = \frac{1}{H} \sum_{h=1}^{H} \frac{(\mathbf{q}_j^{(h)})^\top \mathbf{k}_i^{(h)}}{\sqrt{d_h}}. \tag{3}$$

The resulting alignment relevance is

$$r_i^{\text{attn}} = \max_{j \in \mathcal{I}_{\text{anc}}} A_{j,i}. \tag{4}$$

Thus r_i^{attn} is a pre-softmax alignment score rather than an attention probability.

Cue fusion. Because the cues have different scales, we min–max normalize each visual-token score vector within the current sample. For $\mathbf{r} \in \mathbb{R}^{N_v}$, define

$$\text{Norm}(\mathbf{r}) = \frac{\mathbf{r} - \min(\mathbf{r})}{\max(\mathbf{r}) - \min(\mathbf{r}) + \epsilon}. \tag{5}$$

Here $\epsilon = 10^{-6}$ is a numerical stabilizer. Let $\mathbf{r}^{\text{sim}} = [r_1^{\text{sim}}, \ldots, r_{N_v}^{\text{sim}}]$ and $\mathbf{r}^{\text{attn}} = [r_1^{\text{attn}}, \ldots, r_{N_v}^{\text{attn}}]$. We average the normalized cues and renormalize the fused scores to restore their sample-wise dynamic range before coverage weighting:

$$\mathbf{u} = \text{Norm}\left(\frac{\text{Norm}(\mathbf{r}^{\text{sim}}) + \text{Norm}(\mathbf{r}^{\text{attn}})}{2}\right). \tag{6}$$

Equal weighting introduces no dataset-specific fusion coefficient and is used in all full-QCPruner results. Supplementary Section C.7 compares it with the single-cue variants. In what follows, $u_i = [\mathbf{u}]_i$ denotes the query-conditioned utility of visual token v_i.

3.4 Query-Conditioned Population Coverage

QCPruner formulates visual-token selection as query-conditioned population coverage, in which contextual query utility is incorporated into visual-affinity-based representation of the full visual population. Each visual token acts as a target, while each selected token serves as a candidate representative. Representative utility conditions which selected tokens are suitable for covering the visual population, while target utility additionally weights the evidence being covered; visual affinity links the two roles.

Query-conditioned coverage weighting. Before constructing pairwise weights, we apply the fixed backbone-level calibration

$$\hat{u}_i = \rho + (1-\rho)u_i, \qquad \rho \in [0,1]. \tag{7}$$

For $0 \le \rho < 1$, this transformation preserves the ranking while contracting utility gaps. $\rho = 0$ disables the floor, whereas the diagnostic endpoint $\rho = 1$ yields visual-only coverage. We choose ρ once per backbone family on separate calibration data and freeze it for all reported test datasets and budgets. Section 4.1 and Supplementary Section C.2 give the protocol and sweeps.

We compute nonnegative contextual affinity from visual hidden states:

$$c_{i,j} = [\cos(\mathbf{h}_i^v, \mathbf{h}_j^v)]_+, \tag{8}$$

where $[x]_+ = \max(x, 0)$. Negative cosine similarity is treated as absence of coverage rather than as negative evidence. The query-weighted edge is

$$w_{i,j} = c_{i,j}\hat{u}_i\hat{u}_j. \tag{9}$$

Here $\hat{u}_i$ weights the value of covering target i, whereas $\hat{u}_j$ weights the suitability of representative j. Although their product is symmetric for a fixed edge, the two factors play different roles after maximization over selected representatives.

Coverage objective. For a selected index set $\mathcal{K}$, token v_i receives coverage

$$\mathrm{cover}_i(\mathcal{K}) = \max_{j\in\mathcal{K}} w_{i,j}, \qquad \mathrm{cover}_i(\emptyset) = 0. \tag{10}$$

Summing over all candidate tokens gives the query-conditioned facility-location objective Lin & Bilmes (2011); Dong et al. (2026):

$$F(\mathcal{K}) = \sum_{i=1}^{N_v} \mathrm{cover}_i(\mathcal{K}) = \sum_{i=1}^{N_v} \hat{u}_i \max_{j\in\mathcal{K}} c_{i,j}\hat{u}_j. \tag{11}$$

The exact problem is $\mathcal{K}^* \in \arg\max_{|\mathcal{K}|=K} F(\mathcal{K})$. The outer target weight $\hat{u}_i$ determines how much each target contributes to the objective, while the inner term $c_{i,j}\hat{u}_j$ determines how well selected token j can represent target i. The maximum prevents multiple representatives from receiving additive credit for the same target, and the outer sum rewards coverage across the full visual population.

Greedy marginal-gain selection. Given the current coverage vector, the exact gain of candidate k is

$$\Delta(k \mid \mathcal{K}) = \sum_{i=1}^{N_v} \left[w_{i,k} - \mathrm{cover}_i(\mathcal{K})\right]_+. \tag{12}$$

Since $w_{i,j} \ge 0$, this facility-location objective is normalized, monotone, and submodular; Supplementary Section A.6 gives the proof. Greedy selection therefore retains the standard $(1 - 1/e)$ approximation guarantee for the proxy objective Nemhauser et al. (1978); Dong et al. (2026), not for downstream accuracy. Starting from $\mathcal{K} = \emptyset$, we iteratively select the unselected candidate with the largest $\Delta(k \mid \mathcal{K})$ and update coverage until $|\mathcal{K}| = K$, yielding $\widehat{\mathcal{K}}$. With a precomputed dense affinity matrix, the direct implementation uses $O(N_v^2)$ memory and $O(KN_v^2)$ worst-case selector time, in addition to affinity construction. This cost motivates explicit efficiency analysis rather than an unqualified acceleration claim.

4 Experiments

4.1 Experimental Setup

Model architectures. We evaluate LLaVA-1.5-7B (576 image tokens), LLaVA-NeXT-7B (up to 2880 high-resolution tokens under native AnyRes preprocessing), LLaVA-Video-7B (64 frames, each with 169 patch tokens), and Qwen2.5-VL-7B (1296 image tokens). Their retained budgets are 128/64/32, 640/320/160, 4096/2048/1024, and 512/256/128 patch tokens, respectively. For video, QCPruner solves Eq. (11) independently within each frame under the same per-frame budget and

Layer	Avg. Rel.	Δ vs. L7
5	95.3	-1.4
6	95.2	-1.5
7	96.7	0.0
8	95.7	-1.0
9	**97.0**	$+0.3$

Table 1: Downstream sensitivity to the zero-based pruning layer on Qwen2.5-VL-7B at 256/1296 tokens. The gray row marks the default layer-7 setting; Δ denotes the Avg. Rel. difference from layer 7 in percentage points. Supplementary Section C.6 reports per-task results.

does not allocate tokens adaptively across frames. All methods preserve frame-grid newline tokens, which are excluded from the reported budgets. Supplementary Section A.1 and Appendix Table 7 summarize the backbone-specific pruning settings.

Evaluation benchmarks. For LLaVA image models, we use VQAv2 Goyal et al. (2017), GQA Hudson & Manning (2019), VizWiz Gurari et al. (2018), ScienceQA-IMG Lu et al. (2022), TextVQA Singh et al. (2019), POPE Li et al. (2023), MME Fu et al. (2023), MMBench-EN/CN Liu et al. (2024c), and MMVet Yu et al. (2023). The Qwen2.5-VL suite comprises TextVQA, ChartQA Masry et al. (2022), AI2D Kembhavi et al. (2016), OCRBench Liu et al. (2024d), MME, and MMBench-EN/CN. The video suite comprises MVBench Li et al. (2024), LongVideoBench Wu et al. (2024), and Video-MME Fu et al. (2025). We follow each benchmark's official split and evaluation metric. Supplementary Section A.3 provides additional benchmark details.

Average relative performance. We report average relative performance as $\mathrm{Avg.\,Rel.} = \frac{100}{M}\sum_{m=1}^{M} S_m^{\mathrm{pruned}}/S_m^{\mathrm{full}}$, where S_m^{pruned} and S_m^{full} are the pruned and unpruned scores for task or metric m and M is the number of reported entries. We use $M = 10$ for the main LLaVA image results, $M = 7$ for Qwen2.5-VL and controlled image ablations, and three top-level metrics for the video aggregate. The seven-task controlled LLaVA averages are distinct from the ten-task main results.

Baselines and budget matching. We compare with attention-, similarity-, diversity-, coverage-, and conditional-subset-based methods: FastV, SparseVLM, VisionZip, DART, DivPrune, CD-Pruner, MMTok, and PruneSID Chen et al. (2024a); Zhang et al. (2024); Yang et al. (2025); Wen et al. (2025); Alvar et al. (2025); Zhang et al. (2025); Dong et al. (2026); Fang et al. (2026), plus FastVID Shen et al. (2025) for video. All baseline and QCPruner scores within each backbone are measured in the same local environment rather than copied from source papers. Supplementary Sections A.4 and B report baseline-specific budget details and all compatible results obtained across architectures and budgets.

QCPruner settings. Before downstream evaluation, we select the zero-based layer-7 hook using a disjoint MMStar calibration set on LLaVA-1.5-7B and freeze it across downstream backbones, budgets, and test sets. Supplementary Section C.6 reports the internal layer-wise calibration trajectories. We additionally evaluate downstream sensitivity to nearby pruning layers on Qwen2.5-VL in Section 4.2. Cue weights are fixed equally, at most six keywords are extracted, and all settings are fixed per model family rather than per benchmark.

Utility-floor robustness. We select ρ once per backbone family using benchmarks disjoint from the reported evaluation suite and freeze it across all downstream datasets and budgets. The resulting values are 0, 0.4, and 0.5 for LLaVA image models, Qwen2.5-VL, and LLaVA-Video, respectively. Full sweeps over $\rho \in \{0, 0.1, \ldots, 1.0\}$ are provided in Supplementary Section C.2.

Experiments run on single NVIDIA RTX 5880 Ada GPU with 48 GB memory. Supplementary Sections A.5–C provide implementation details, complete results, and calibration analyses.

4.2 Downstream Sensitivity to the Pruning Layer

We evaluate zero-based pruning layers 5–9 on Qwen2.5-VL-7B at 256/1296 tokens with all other settings fixed, and report the seven-task Avg. Rel. relative to the default layer-7 hook.

Backbone	Target/Full	Baseline	Base.	Ours	Δ
LLaVA-1.5	128/576	CDPruner	97.9	**99.3**	+1.4
LLaVA-1.5	64/576	CDPruner	96.0	**98.2**	+2.2
LLaVA-1.5	32/576	CDPruner	93.9	**96.1**	+2.2
LLaVA-NeXT	640/2880	SparseVLM	97.8	**98.1**	+0.3
LLaVA-NeXT	320/2880	DART	95.0	**96.6**	+1.6
LLaVA-NeXT	160/2880	CDPruner	92.8	**94.9**	+2.1
LLaVA-Video	4096/10816	FastVID	99.4	**99.6**	+0.2
LLaVA-Video	2048/10816	FastVID	97.0	**98.1**	+1.1
LLaVA-Video	1024/10816	FastVID	94.3	**96.3**	+2.0
Qwen2.5-VL	512/1296	VisionZip	98.4	**98.5**	+0.1
Qwen2.5-VL	256/1296	VisionZip	92.5	**96.7**	+4.2
Qwen2.5-VL	128/1296	VisionZip	82.9	**92.8**	+9.9

Table 2: Avg. Rel. at all reported token budgets across four backbones. Base. is the highest displayed baseline score at the same nominal patch-token target; the Baseline column names one method attaining it. At LLaVA-1.5 128/576, CDPruner ties MMTok and VisionZip at the displayed precision. Δ is the absolute percentage-point difference computed before rounding.

Downstream retention is non-monotonic with pruning depth: layer 7 outperforms layers 5, 6, and 8 by 1.4, 1.5, and 1.0 Avg. Rel. points, respectively. Layer 9 scores 0.3 points higher but delays pruning by two additional decoder layers. Thus, the sweep complements the internal calibration and supports layer 7 as an earlier near-peak operating point rather than the highest-scoring tested hook.

4.3 Main Results Across Architectures and Modalities

Table 2 summarizes Avg. Rel. across all reported budgets and backbones. Appendix Tables 9–12 provide complete task-level results.

On LLaVA-1.5, QCPruner achieves Avg. Rel. scores of 99.3, 98.2, and 96.1 at 128, 64, and 32 tokens, exceeding the strongest displayed complete-system baseline by 1.4, 2.2, and 2.2 points, respectively. Against MMTok specifically, the gains are 1.4, 2.5, and 2.7 points. Appendix Table 9 shows consistent gains on VQAv2, GQA, MME, and MMBench, while QCPruner trails the best displayed result on VizWiz and POPE and at some MMVet budgets. The aggregate advantage therefore does not imply uniform task-wise dominance.

The advantage generally widens under tighter compression, from 0.3 to 2.1 points on LLaVA-NeXT, 0.2 to 2.0 on LLaVA-Video, and 0.1 to 9.9 on Qwen2.5-VL, with the clearest gains under aggressive compression.

4.4 Computational Cost at a Fixed Token Budget

At the matched final budget of 320/2880 visual tokens, full QCPruner achieves 88.4 POPE F1 but incurs 11.1 TFLOPs and 127.5 ms of LLM prefill, compared with 84.1 F1, 5.0 TFLOPs, and 41.8 ms for DivPrune. Thus, QCPruner provides stronger retention at the same final token count but does not dominate earlier-pruning methods in compute or latency.

QCPruner-Early isolates the effect of pruning depth: moving pruning before the decoder reduces the trajectory to 5.0 TFLOPs and 40.6 ms of prefill, but lowers F1 to 84.4. This comparison exposes the fidelity–cost trade-off underlying the full method: later contextual selection provides richer visual–text information for deciding which tokens to retain, at the cost of processing the dense visual sequence through layers 0–7.

Relative to unpruned inference, full QCPruner still reduces the model trajectory from 32.8 to 11.1 TFLOPs and prefill from 271.4 to 127.5 ms. However, our primary claim is improved downstream retention under a prescribed final token budget rather than compute- or latency-matched superiority. The LLM prefill timing in Table 3 includes QCPruner's pruning-hook execution but excludes query-anchor construction. Supplementary Section C.1 reports that initialized query-anchor construction averages 1.67 ms/sample outside the hook. Improving the cost of dense contextual selection and repeated marginal-gain evaluation is left to future work.

Method	FLOPs (T)	LLM Pre. (ms)	Mem. (GB)	F1
All 2880 Tokens (100%)				
Unpruned	32.8	271.4	15.8	86.5
320 Tokens (↓88.9%)				
DivPrune	**5.0**	41.8	**14.7**	84.1
QCPruner-Early	**5.0**	**40.6**	**14.7**	84.4
QCPruner	11.1	127.5	**14.7**	**88.4**

Table 3: Model-trajectory cost and POPE F1 on LLaVA-NeXT-7B at 320/2880 patch tokens, evaluated on the full POPE set (8,900 samples) and measured on an RTX 5880 Ada.

	Qwen2.5-VL		LLaVA-1.5
Edge weight	**128/1296**	**256/1296**	**64/576**
$c_{i,j}$	87.2	93.4	95.7
$c_{i,j}\hat{u}_i$	87.9	94.3	97.3
$c_{i,j}\hat{u}_j$	92.6	**96.7**	98.0
$c_{i,j}\hat{u}_i\hat{u}_j$	**92.8**	**96.7**	**98.6**

Table 4: Role-wise query-utility ablations across three settings (seven-task Avg. Rel.). Supplementary Section C.3 reports per-task scores.

4.5 Ablation Studies

We ablate the key components connecting query conditioning to coverage-based selection. Pruning-layer selection is calibrated separately as described in Section 4.1. Cue, anchor, role-wise utility, and selector ablations use the seven-task subset reported in the corresponding tables, with Avg. Rel. computed as defined above.

Query Utility and Role-wise Conditioning

At 32 tokens, similarity-only, QK-only, fusion with the full prompt, and fusion with matched anchors achieve 95.8, 95.6, 96.2, and 96.8 Avg. Rel., respectively. At 64 tokens, Fusion+Anc. reaches 98.6, compared with 98.1 and 97.4 for the QK-only and similarity-only variants. These results support both cue fusion and anchor refinement; complete per-task results are provided in Supplementary Section C.7.

Table 4 isolates where query utility contributes to population coverage. Representative-side conditioning provides most of the gain over visual-only coverage, improving Avg. Rel. from 87.2/93.4/95.7 to 92.6/96.7/98.0 across the three settings. Adding target-side weighting yields 92.8/96.7/98.6, showing that target utility is a smaller, setting-dependent refinement rather than the primary source of improvement.

Across two to eight keywords, Avg. Rel. ranges from 97.8 to 98.6 and remains at 98.6 at the reported precision for four, six, and eight keywords. Replacing SparseVLM's original text conditioning with the same keyword-matched anchors improves all seven tasks and raises Avg. Rel. from 90.9 to 91.9 (Supplementary Sections C.4 and D).

Utility Faithfulness via Controlled Token Removal

We directly test utility faithfulness on the first 500 MMStar validation samples and 500 MME question instances with Qwen2.5-VL by removing the highest-utility 10%, lowest-utility 10%, or a random 10% of visual tokens after decoder layer 7. We hold the removal fraction and $\rho = 0.4$ fixed and measure teacher-forced answer-token $D_{\mathrm{KL}}(p_{\mathrm{dense}} \| p_{\mathrm{ablated}})$ against the dense model.

Top-utility removal induces substantially larger answer perturbations on both datasets (Table 5): 0.0176 versus 0.0040/0.0020 on MMStar and 0.0130 versus 0.0030/0.0032 on MME for Random/Bottom removal. Two-sided Paired Wilcoxon signed-rank tests give $p < 1.1 \times 10^{-18}$, and

Dataset	Removed	Answer KL ↑	$P(\text{Top} > X)$	p-value
MMStar	Bottom	0.001958	73.6%	2.85×10^{-31}
	Random	0.004027	69.2%	1.08×10^{-18}
	Top	**0.017598**	–	–
MME	Bottom	0.003247	82.6%	2.00×10^{-43}
	Random	0.003017	77.0%	3.00×10^{-33}
	Top	**0.013029**	–	–

Table 5: Utility-faithfulness analysis under controlled token removal. We remove 10% of visual tokens at decoder layer 7 according to their query utility. $P(\text{Top} > X)$ denotes the fraction of paired samples for which Top-utility removal induces a larger answer-token KL than group X.

	Qwen2.5-VL		**LLaVA-1.5**
Selector	**128/1296**	**256/1296**	**64/576**
Top-K	69.6	78.6	96.5
MMR	88.7	93.5	98.3
Visual-only Cov.	87.2	93.4	95.7
CDPruner-DPP	89.0	93.7	**98.6**
MMTok-style	89.7	95.1	97.8
QCPruner	**92.8**	**96.7**	**98.6**

Table 6: Controlled seven-task selector ablations (Avg. Rel.). Upstream signals and budget are fixed within each column. CDPruner-DPP and MMTok-style denote backend transplants rather than complete systems.

Top removal yields larger KL on 69.2–82.6% of paired samples. Thus, high utility preferentially identifies tokens whose removal more strongly affects the answer distribution, without implying monotonicity across the full ranking.

Effect of Selection Strategy

We next isolate the subset objective by fixing candidates, anchors, fused query utility, the post-layer-7 hook, utility floor, and budget within each setting. The alternatives are Top-K, visual-token maximal marginal relevance (MMR) Carbonell & Goldstein (1998) with $\lambda_{\text{MMR}} = 0.5$, visual-only facility location, a conditional-DPP/Fast-MAP backend, an MMTok-style additive-coverage backend Dong et al. (2026), and QCPruner. The DPP and MMTok-style entries are controlled backend transplants rather than complete-system results. The MMTok-style backend follows the official MMTok objective form and method-specific hyperparameters without retuning.

Table 6 shows that QCPruner exceeds the controlled CDPruner-DPP and MMTok-style backends by 3.8/3.0 and 3.1/1.6 Avg. Rel. points at Qwen-128/256, respectively, while tying CDPruner-DPP on LLaVA at the reported precision. Relative to visual-only coverage, QCPruner gains 5.6, 3.3, and 2.9 points across the three settings. These controlled comparisons support query-conditioned population coverage but do not imply universal superiority over the complete CDPruner or MMTok pipelines. Task-level results are provided in Supplementary Section C.5.

5 Conclusion

QCPruner introduces query-conditioned population coverage for training-free visual-token pruning, using contextual per-visual query utility to guide population-level representation. Across four MLLM backbones and twelve token budgets, it consistently preserves strong downstream performance under matched final token budgets, with the clearest gains under aggressive compression. Controlled analyses further show that representative-side conditioning provides the dominant benefit while target-side weighting offers a smaller complementary refinement. Full QCPruner trades additional prefill computation for this retention advantage, motivating future work on lower-cost query-conditioned coverage.

AI USE STATEMENT

Generative AI tools were used to assist with drafting and polishing parts of the manuscript and for retrieval and discovery of relevant literature. All AI-assisted text and literature suggestions were reviewed and verified by the authors. The authors take full responsibility for the final content of this work.

REPRODUCIBILITY STATEMENT

We provide detailed model, dataset, pruning-budget, calibration, and evaluation settings in the main paper and appendix. The appendix reports complete benchmark-wise results, utility-floor and pruning-layer calibration, controlled ablations, computational details, and the proof of the optimization property. An anonymized implementation and evaluation scripts are included in the supplementary materials.

REFERENCES

Saeed Ranjbar Alvar, Gursimran Singh, Mohammad Akbari, and Yong Zhang. Divprune: Diversity-based visual token pruning for large multimodal models. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pp. 9392–9401, 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.

Jaime G. Carbonell and Jade Goldstein. The use of MMR, diversity-based reranking for reordering documents and producing summaries. In *Proceedings of the 21st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pp. 335–336. ACM, 1998. doi: 10.1145/290941.291025.

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, and Feng Zhao. Are we on the right way for evaluating large vision-language models? In *Advances in Neural Information Processing Systems*, volume 37, 2024b.

Zhe Chen, Jiannan Wu, Wenhai Wang, Weijie Su, Guo Chen, Sen Xing, Muyan Zhong, Qinglong Zhang, Xizhou Zhu, Lewei Lu, et al. Internvl: Scaling up vision foundation models and aligning for generic visual-linguistic tasks. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 24185–24198, 2024c.

Jinhong Deng, Wen Li, Joey Tianyi Zhou, and Yang He. SCOPE: Saliency-coverage oriented token pruning for efficient multimodel LLMs. In *Advances in Neural Information Processing Systems*, volume 38, 2025. doi: 10.52202/085713-5394.

Sixun Dong, Juhua Hu, Mian Zhang, Ming Yin, Yanjie Fu, and Qi Qian. MMTok: Multimodal coverage maximization for efficient inference of VLMs. In *The Fourteenth International Conference on Learning Representations*, 2026. URL https://openreview.net/forum?id=GvPdSWZT31.

Zhengyao Fang, Pengyuan Lyu, Chengquan Zhang, Guangming Lu, Jun Yu, and Wenjie Pei. Prune redundancy, preserve essence: Vision token compression in vlms via synergistic importance-diversity. *arXiv preprint arXiv:2603.09480*, 2026.

Chaoyou Fu, Peixian Chen, Yunhang Shen, Yulei Qin, Mengdan Zhang, Xu Lin, Jinrui Yang, Xiawu Zheng, Ke Li, Xing Sun, et al. MME: A comprehensive evaluation benchmark for multimodal large language models, 2023.

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 *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 24108–24118, 2025.

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.

Maarten Grootendorst. Keybert: Minimal keyword extraction with bert., 2020. URL `https://doi.org/10.5281/zenodo.4461265`.

Danna Gurari, Qing Li, Abigale J Stangl, Anhong Guo, Chi Lin, Kristen Grauman, Jiebo Luo, and Jeffrey P Bigham. Vizwiz grand challenge: Answering visual questions from blind people. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 3608–3617, 2018.

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.

Ahmadreza Jeddi, Negin Baghbanzadeh, Elham Dolatabadi, and Babak Taati. Similarity-aware token pruning: Your vlm but faster. *arXiv preprint arXiv:2503.11549*, 2025.

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.

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, 2024.

Yangfu Li, Hongjian Zhan, Tianyi Chen, Qi Liu, Yu-Jie Xiong, and Yue Lu. Why 1 + 1 ¡ 1 in visual token pruning: Beyond naive integration via multi-objective balanced covering. In *Advances in Neural Information Processing Systems*, volume 38, 2025. doi: 10.52202/085713-2417.

Yifan Li, Yifan Du, Kun Zhou, Jinpeng Wang, Xin Zhao, and Ji-Rong Wen. Evaluating object hallucination in large vision-language models. In *Proceedings of the 2023 conference on empirical methods in natural language processing*, pp. 292–305, 2023.

Hui Lin and Jeff Bilmes. A class of submodular functions for document summarization. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics*, pp. 510–520, 2011.

Haotian Liu, Chunyuan Li, Qingyang Wu, and Yong Jae Lee. Visual instruction tuning. *Advances in neural information processing systems*, 36:34892–34916, 2023.

Haotian Liu, Chunyuan Li, Yuheng Li, and Yong Jae Lee. Improved baselines with visual instruction tuning. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 26296–26306, 2024a.

Haotian Liu, Chunyuan Li, Yuheng Li, Bo Li, Yuanhan Zhang, Sheng Shen, and Yong Jae Lee. LLaVA-NeXT: Improved reasoning, OCR, and world knowledge. LLaVA project blog, 2024b. URL `https://llava-vl.github.io/blog/2024-01-30-llava-next/`.

Yuan Liu, Haodong Duan, Yuanhan Zhang, Bo Li, Songyang Zhang, Wangbo Zhao, Yike Yuan, Jiaqi Wang, Conghui He, Ziwei Liu, et al. Mmbench: Is your multi-modal model an all-around player? In *European conference on computer vision*, pp. 216–233. Springer, 2024c.

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, 2024d.

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.

Ahmed Masry, Xuan Long Do, Jia Qing Tan, Shafiq Joty, and Enamul Hoque. 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.

George L. Nemhauser, Laurence A. Wolsey, and Marshall L. Fisher. An analysis of approximations for maximizing submodular set functions. *Mathematical Programming*, 14(1):265–294, 1978.

Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language supervision. In *International conference on machine learning*, pp. 8748–8763. PmLR, 2021.

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

Leqi Shen, Guoqiang Gong, Tao He, Yifeng Zhang, Pengzhang Liu, Sicheng Zhao, and Guiguang Ding. FastVID: Dynamic density pruning for fast video large language models. In *Advances in Neural Information Processing Systems*, volume 38, pp. 123553–123581, 2025.

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.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

Peng Wang, Shuai Bai, Sinan Tan, Shijie Wang, Zhihao Fan, Jinze Bai, Keqin Chen, Xuejing Liu, Jialin Wang, Wenbin Ge, et al. Qwen2-vl: Enhancing vision-language model's perception of the world at any resolution. *arXiv preprint arXiv:2409.12191*, 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. 9961–9980, 2025.

Haoning Wu, Dongxu Li, Bei Chen, and Junnan Li. Longvideobench: A benchmark for long-context interleaved video-language understanding. *Advances in Neural Information Processing Systems*, 37:28828–28857, 2024.

Junbin Xiao, Xindi Shang, Angela Yao, and Tat-Seng Chua. NExT-QA: Next phase of question-answering to explaining temporal actions. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 9777–9786, 2021.

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 *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 19792–19802, 2025.

Weihao Ye, Qiong Wu, Wenhao Lin, and Yiyi Zhou. Fit and prune: Fast and training-free visual token pruning for multi-modal large language models. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 39, pp. 22128–22136, 2025.

Shukang Yin, Chaoyou Fu, Sirui Zhao, Ke Li, Xing Sun, Tong Xu, and Enhong Chen. A survey on multimodal large language models. *National Science Review*, 11(12):nwae403, 2024.

Weihao Yu, Zhengyuan Yang, Linjie Li, Jianfeng Wang, Kevin Lin, Zicheng Liu, Xinchao Wang, and Lijuan Wang. Mm-vet: Evaluating large multimodal models for integrated capabilities. *arXiv preprint arXiv:2308.02490*, 2023.

Xiaohua Zhai, Basil Mustafa, Alexander Kolesnikov, and Lucas Beyer. Sigmoid loss for language image pre-training. In *Proceedings of the IEEE/CVF international conference on computer vision*, pp. 11975–11986, 2023.

Qizhe Zhang, Mengzhen Liu, Lichen Li, Ming Lu, Yuan Zhang, Junwen Pan, Qi She, and Shanghang Zhang. Beyond attention or similarity: Maximizing conditional diversity for token pruning in MLLMs. In *Advances in Neural Information Processing Systems*, volume 38, pp. 25438–25468, 2025.

Yuan Zhang, Chun-Kai Fan, Junpeng Ma, Wenzhao Zheng, Tao Huang, Kuan Cheng, Denis Gudovskiy, Tomoyuki Okuno, Yohei Nakata, Kurt Keutzer, et al. Sparsevlm: Visual token sparsification for efficient vision-language model inference. *arXiv preprint arXiv:2410.04417*, 2024.

Deyao Zhu, Xiaoqian Shen, Xiang Li, Mohamed Elhoseiny, et al. MiniGPT-4: Enhancing vision-language understanding with advanced large language models. In *International Conference on Learning Representations*, 2024.

SUPPLEMENTARY MATERIAL

This supplementary material is organized as follows. Appendix A provides experimental details. It also proves submodularity of the full QCPruner objective and its greedy approximation guarantee. Appendix B reports complete benchmark-level results. Appendix C provides additional ablations, including the pruning-layer stability analysis under separate calibration. Appendix D studies the transferability of keyword-matched query anchors. Appendix E discusses limitations and broader impact. Appendix F provides the qualitative motivation examples omitted from the main paper due to space constraints. Throughout, full QCPruner denotes query-conditioned population coverage, whereas QCPruner-Early is a separate visual-only control.

A ADDITIONAL EXPERIMENTAL DETAILS

A.1 MODEL ARCHITECTURES AND PRUNING SETTINGS

Table 7 summarizes the evaluated backbones, patch-token targets, and full-QCPruner pruning layer and backbone-level hyperparameters. Layer indices throughout the paper follow the implementation-level zero-based decoder indexing: layer ℓ supplies the contextual states used for pruning, which is performed after layer ℓ and before layer $\ell+1$. For full QCPruner, the post-layer-7 hook is shared across backbones, while equal-weight cue fusion and the selected utility-floor coefficient are fixed across datasets and budgets within each backbone. QCPruner-Early is specified separately in Appendix A.2.

For Qwen2.5-VL and LLaVA-Video, we apply a utility-floor calibration to mitigate overly concentrated utility distributions during population-coverage selection. Given the fused utility score u_i, the calibrated utility is defined as

$$\hat{u}_i = \rho + (1-\rho)u_i \tag{13}$$

For $0 < \rho < 1$, this transformation preserves the ranking induced by u_i while contracting utility differences, preventing moderately relevant tokens from being excessively downweighted during coverage selection. When $\rho = 0$, the calibration is disabled. $\rho = 1$ makes every calibrated utility equal to one and is used only as a visual-only diagnostic endpoint. We set $\rho = 0$ for LLaVA image backbones, $\rho = 0.4$ for Qwen2.5-VL, and $\rho = 0.5$ for LLaVA-Video, selected once on separate calibration data and then frozen across downstream datasets and budgets (Appendix C.2).

For LLaVA-Video-7B, QCPruner performs selection independently within each frame under a fixed per-frame patch-token budget. All newline tokens separating frame-level visual grids are preserved and excluded from the reported pruning budget.

A.2 QCPRUNER-EARLY VARIANT

QCPruner-Early is a separate visual-only pre-decoder control that shortens the token sequence processed by the LLM. It operates on visual-token positions in the assembled input-embedding tensor immediately before decoder layer 0 and does not use the full query-conditioned objective. Let $\mathbf{E}_{\text{in}}$ denote the assembled input-embedding tensor, and let $\mathbf{x}_i = \text{float32}(\mathbf{E}_{\text{in}}[p_i, :]) \in \mathbb{R}^d$ for candidate position p_i.

Visual affinity and norm prior. The variant first computes a nonnegative cosine-affinity matrix:

$$c^{\text{early}}_{i,j} = [\cos(\mathbf{x}_i, \mathbf{x}_j)]_+, \qquad c^{\text{early}}_{i,i} = 1. \tag{14}$$

The diagonal is explicitly reset to one after clipping. Because query-conditioned contextual cues are not yet available from decoder-layer interactions at this early insertion point, we use the L2 norm of each raw visual embedding as a lightweight candidate prior:

$$n_i = \|\mathbf{x}_i\|_2 \tag{15}$$

For each sample, the implementation uses the 5th and 95th percentiles $Q_{.05}$ and $Q_{.95}$ of the embedding norms to construct a robust min–max score. We clip the robust min–max score as

$$\tilde{n}_i = \text{clip}\left(\frac{n_i - Q_{.05}}{Q_{.95} - Q_{.05} + \epsilon}, 0, 1\right), \tag{16}$$

Backbone	Patch tokens full → retained	Prune after layer	ρ
LLaVA-1.5-7B	$576 \rightarrow 128/64/32$	7	0.0
LLaVA-NeXT-7B	$2880^{\dagger} \rightarrow 640/320/160$	7	0.0
Qwen2.5-VL-7B	$1296 \rightarrow 512/256/128$	7	0.4
LLaVA-Video-7B	$10816 \rightarrow 4096/2048/1024$	7	0.5

Table 7: Backbone-specific pruning settings and nominal patch-token targets. † Maximum full patch-token count, while the actual count is input-dependent. The full LLaVA-Video input contains 64×169 patch tokens. QCPruner-Early operates before decoder layer 0 and does not use ρ.

where $\epsilon = 10^{-6}$, and define

$$a_i = 1 + \beta \tilde{n}_i, \tag{17}$$

with $\beta = 0.20$ in all reported QCPruner-Early experiments, giving $a_i \in [1, 1.2]$. Neither a_i nor $c_{i,j}^{\text{early}}$ uses text- or query-derived information.

Coverage-inspired selection. For $N_v > K$, the first selected token combines aggregate visual affinity with the norm prior:

$$j_1 = \arg\max_j \left(a_j \sum_{i=1}^{N_v} c_{i,j}^{\text{early}} \right). \tag{18}$$

For a selected set $\mathcal{K}$, define the current coverage of token v_i as

$$\text{cover}_i^{\text{early}}(\mathcal{K}) = \max_{j \in \mathcal{K}} c_{i,j}^{\text{early}}. \tag{19}$$

Each subsequent step scores a remaining candidate by its uncoveredness relative to the selected set, weighted by the norm prior:

$$j_t = \arg\max_{j \notin \mathcal{K}} \left[1 - \text{cover}_j^{\text{early}}(\mathcal{K}) \right]_+ a_j. \tag{20}$$

After adding j_t, the vector is updated by

$$\text{cover}_i^{\text{early}}(\mathcal{K} \cup \{j_t\}) = \max\left(\text{cover}_i^{\text{early}}(\mathcal{K}), c_{i,j_t}^{\text{early}} \right). \tag{21}$$

The procedure selects exactly K visual tokens when $N_v > K$. Otherwise, the visual sequence is left unchanged.

Importantly, the score above uses candidate j's own uncoveredness rather than the summed facility-location marginal gain over all targets used by full QCPruner. QCPruner-Early is therefore a coverage-inspired, norm-weighted self-novelty heuristic and does not inherit the full objective's approximation guarantee. Constructing the dense affinity matrix requires $O(N_v^2 d)$ time and $O(N_v^2)$ memory. Given this matrix, column summation and K vector updates require $O(N_v^2 + KN_v)$ time. Its low model-side cost in the main-paper retention–cost comparison comes from pruning before all LLM blocks rather than eliminating dense-affinity preprocessing.

LLaVA-1.5, Qwen2.5-VL, and LLaVA-Video use fixed visual-token layouts within each backbone. In contrast, LLaVA-NeXT retains its native AnyRes multi-scale preprocessing, resulting in input-dependent full-token counts. This setup evaluates transfer across controlled backbone settings while separately examining robustness to native multi-scale inputs.

A.3 Evaluation Benchmarks

We provide detailed descriptions of the evaluation benchmarks used in our experiments in the following subsections. All evaluations follow the official protocol of each benchmark, and we report the corresponding task-specific metric. For a method m, we compute average relative performance retention as

$$\text{Avg. Rel.}(m) = \frac{100}{T} \sum_{t=1}^{T} \frac{s_{m,t}}{s_{\text{full},t}},$$

where $s_{m,t}$ and $s_{\text{full},t}$ are the method and unpruned scores on benchmark t, respectively, and T is the number of top-level benchmarks in the corresponding table. Thus, Avg. Rel. gives every benchmark equal weight despite differences in its native score scale. For the video results, the three top-level terms are MVBench test, LongVideoBench validation, and Video-MME without subtitles. The Short/Medium/Long Video-MME columns are diagnostic breakdowns and are not averaged again.

General Image Benchmarks

VQAv2. VQAv2 Goyal et al. (2017) evaluates general visual question answering over natural images, covering objects, attributes, scenes, and simple relations. For visual token pruning, it provides a broad test of whether the retained visual subset preserves task-relevant evidence across diverse question types. We use the test-dev split and report the official VQA accuracy, which measures agreement with the human reference answers.

GQA. GQA Hudson & Manning (2019) focuses on compositional visual reasoning over objects, attributes, and relationships. Its emphasis on structured and relational reasoning makes it useful for evaluating whether pruning preserves both salient visual evidence and the supporting context required for compositional answers. We use the balanced test-dev split and report answer accuracy, the benchmark's direct measure of compositional question-answering performance.

VizWiz. VizWiz Gurari et al. (2018) contains real-world images and questions collected from blind users. The images are often blurry, poorly framed, or visually ambiguous, while the questions can be short or highly referential. These characteristics make VizWiz challenging for query-aware pruning because some queries, such as "What is this?", provide limited semantic guidance for identifying relevant visual evidence. We use the test split and report the official VQA accuracy against crowdsourced reference answers.

ScienceQA-IMG. ScienceQA-IMG Lu et al. (2022) evaluates multimodal science reasoning with image context. Its questions often require combining visual evidence with commonsense or domain knowledge. It therefore evaluates whether pruning preserves the visual evidence needed for multimodal reasoning. We use the image-context subset of the test split and report multiple-choice accuracy, which directly measures the fraction of science questions answered correctly.

POPE. POPE Li et al. (2023) evaluates object hallucination in multimodal models by asking whether specific objects are present in an image. It is relevant to query-aware token pruning because correct predictions depend on preserving object-level evidence for the queried category. POPE therefore evaluates performance retention on hallucination-sensitive object-presence decisions rather than serving as a direct token-grounding metric. We use the test split and report F1, which balances precision and recall for the binary object-presence decisions.

MME. MME Fu et al. (2023) is a comprehensive benchmark covering both perception and cognition. Its perception tasks include object existence, counting, position, color, OCR, and fine-grained recognition, while its cognition tasks involve commonsense reasoning, calculation, translation, and code-related reasoning. Its broad task coverage provides a comprehensive evaluation of performance retention under visual token pruning. We report the official aggregate MME score over the perception and cognition evaluations, whose task-wise scoring rewards both correct positive and negative decisions. For LLaVA-1.5 and LLaVA-NeXT, we report the official MME perception score. For Qwen2.5-VL, we report the sum of the perception and cognition scores, following the corresponding backbone-specific evaluation pipeline. The task-wise scoring rewards both correct positive and negative decisions. Because the evaluation scope differs, raw MME scores should be compared only within the same backbone and evaluation scope.

MMBench-EN and MMBench-CN. MMBench Liu et al. (2024c) evaluates broad multimodal capabilities using multiple-choice questions. We report both the English and Chinese splits to broaden language coverage, without treating them as a controlled cross-lingual robustness test based on translated or paraphrased versions of the same prompts. For both official evaluation splits, we report

CircularEval accuracy, which reduces option-position bias by requiring consistency across circularly permuted choices.

MMVet. MMVet Yu et al. (2023) evaluates integrated multimodal capabilities, including recognition, OCR, knowledge, language generation, spatial awareness, and mathematics. Its questions often require combining multiple skills rather than relying on a single visual cue. This makes MMVet challenging for aggressive pruning, as removing complementary visual evidence can degrade performance on multi-skill reasoning tasks. We report the official LLM-judged MM-Vet score for open-ended responses, which evaluates correctness across the benchmark's integrated capabilities.

TEXT-ORIENTED IMAGE BENCHMARKS

TextVQA. TextVQA Singh et al. (2019) evaluates a model's ability to read and reason over scene text in natural images. Because answers often depend on fine-grained OCR evidence occupying small image regions, the benchmark is sensitive to whether pruning preserves localized text information together with its supporting visual context. We use the validation split and report the official VQA accuracy against human reference answers.

ChartQA. ChartQA Masry et al. (2022) focuses on question answering over charts. It requires visual parsing of chart elements together with logical or arithmetic reasoning over values, labels, and trends. For visual token pruning, ChartQA is challenging because small textual or graphical elements may contain evidence critical to answering the question. We use the test split and report relaxed accuracy, which accommodates the benchmark's prescribed numerical tolerance while still requiring correct non-numerical answers.

AI2D. AI2D Kembhavi et al. (2016) evaluates diagram understanding using science-related diagrams and multiple-choice questions. Compared with natural images, diagrams contain structured visual elements, symbolic labels, and spatial relations. For visual token pruning, AI2D tests whether sparse but task-relevant diagram elements are preserved. We use the test split and report multiple-choice accuracy, the direct fraction of diagram questions answered correctly.

OCRBench. OCRBench Liu et al. (2024d) evaluates OCR-oriented multimodal understanding across diverse text-centric visual tasks. It covers text recognition, scene-text VQA, document-oriented VQA, key information extraction, and handwritten expression recognition. Because OCR evidence is often fine-grained and spatially localized, OCRBench is particularly sensitive to aggressive visual token pruning. We use the official 1,000-question evaluation set and report the aggregate number of correctly answered items across its five task groups. The maximum is 1,000 and higher is better.

VIDEO BENCHMARKS

MVBench. MVBench Li et al. (2024) evaluates temporal understanding in multimodal video models. It includes tasks involving actions, events, temporal order, and dynamic scene changes. For visual token pruning, MVBench tests whether frame-level evidence needed for temporal reasoning is preserved. We use the test split and report mean multiple-choice accuracy across its 20 tasks.

LongVideoBench. LongVideoBench Wu et al. (2024) evaluates long-form video understanding. It requires models to retrieve and reason over information distributed across extended temporal contexts. For visual token pruning, LongVideoBench is challenging because aggressive compression may remove sparse but task-relevant temporal evidence. We use the validation split and report multiple-choice accuracy, which directly measures long-video question-answering correctness.

Video-MME. Video-MME Fu et al. (2025) is a comprehensive video understanding benchmark covering videos of varying durations and reasoning categories. We use the no-subtitle setting to focus evaluation on visual evidence from video frames. For visual token pruning, Video-MME evaluates whether query-relevant temporal and visual evidence is preserved under large-scale token reduction. We report multiple-choice accuracy in the no-subtitle setting, both overall and for the

official short-, medium-, and long-duration subsets. The overall score is used as the top-level Video-MME term in Avg. Rel. The duration-specific columns are diagnostic breakdowns and are not averaged again.

A.4 Comparison Methods

We report compatible training-free visual token pruning and compression baselines spanning attention-based pruning, text-guided pruning, visual redundancy reduction, and diversity-oriented token selection.

FastV. FastV Chen et al. (2024a) is a training-free pruning method that removes visual tokens according to attention-derived importance. It serves as a representative baseline for visual token pruning within the language model.

SparseVLM. SparseVLM Zhang et al. (2024) uses text-related attention signals to guide visual token sparsification. It serves as a representative baseline for text-guided visual token pruning.

VisionZip. VisionZip Yang et al. (2025) compresses visual tokens by combining dominant and contextual tokens to preserve both salient and contextual visual information. It serves as a representative baseline for visual-side token compression.

DART. DART Wen et al. (2025) focuses on reducing visual-token redundancy, treating token duplication as a key source of inefficiency in visual token pruning. It serves as a representative redundancy-reduction baseline.

DivPrune. DivPrune Alvar et al. (2025) performs diversity-oriented visual token pruning to preserve diverse visual information in the retained subset. It provides a direct comparison with diversity-based subset selection that does not explicitly condition on the current query.

CDPruner. CDPruner Zhang et al. (2025) performs visual token pruning through conditional-diversity modeling with a DPP-style subset selection objective. The CDPruner rows in the complete benchmark tables refer to the full method and are distinct from the controlled CDPruner-DPP selector transplant in Appendix C.5, where CDPruner's native relevance input is replaced with QCPruner-derived query utility.

MMTok. MMTok Dong et al. (2026) is closely related to QCPruner in its use of facility-location-style subset selection. Its pre-decoder objective combines text–vision and vision–vision facility-location terms, yielding a monotone submodular objective optimized by standard greedy selection. In contrast, QCPruner derives query-conditioned visual-token utility from keyword-matched contextual MLLM states at the pruning layer and uses it to modulate both target importance and representative suitability within visual-population coverage. We locally evaluate MMTok at the same nominal patch-token budgets on LLaVA-1.5, LLaVA-NeXT, and Qwen2.5-VL. The controlled selector study in Appendix C.5 additionally includes an *MMTok-style* objective-level backend transplant on Qwen2.5-VL and LLaVA-1.5. This entry is distinct from the complete MMTok system reported in the benchmark tables.

PruneSID. PruneSID Fang et al. (2026) is a recent training-free visual token pruning method included for additional comparison under matched nominal token budgets.

FastVID. For video experiments, we additionally compare with FastVID Shen et al. (2025), a video-oriented pruning baseline evaluated under matched total token budgets.

A.5 Implementation and Evaluation Protocol

QCPruner uses equal-weight cue fusion together with the backbone-specific pruning layer and utility-floor coefficient ρ listed in Table 7, whereas QCPruner-Early follows Appendix A.2.

Family	Family-specific software stack
LLaVA Image	Python 3.10.20; PyTorch 2.1.2; CUDA 11.8; cuDNN 8.7.0; Transformers 4.37.2; Accelerate 0.21.0; official LLaVA evaluation.
LLaVA-Video	Python 3.10.20; PyTorch 2.1.2+cu121; CUDA 12.1; cuDNN 8.9.0; Transformers 4.40.0.dev0; Accelerate 1.13.0; lmms-eval 0.7.1.
Qwen2.5-VL	Python 3.12.13; PyTorch 2.11.0+cu128; CUDA 12.8; cuDNN 9.19.0; Transformers 5.5.4; Accelerate 1.13.0; lmms-eval 0.7.1.

Table 8: Family-specific software environments. Hardware and operating-system details shared by all families are stated in the text.

Keyword-matched query anchors are constructed using KeyBERT v0.9.0 Grootendorst (2020) with the model2vec v0.8.1 StaticModel `minishlab/potion-base-8M`.[1] Candidate extraction uses unigram range $(1, 1)$, KeyBERT's default English stop-word filtering, and at most six ranked keywords, with optional MMR/diversity reranking disabled. Each returned keyword is matched back to all corresponding token spans in the original prompt, and the matched tokens are used as textual anchors for visual utility estimation. If no keyword match is found, we fall back to valid non-template text tokens after excluding special tokens and visual placeholders.

For both LLaVA-1.5 and high-resolution LLaVA-NeXT image experiments, we use the official LLaVA codebase.[2] For LLaVA-Video experiments, we use the LLaVA-NeXT codebase[3] for model implementation and multimodal preprocessing and lmms-eval[4] for benchmark evaluation. For Qwen2.5-VL, we use the official model implementation and lmms-eval for evaluation. All benchmarks use deterministic greedy decoding with sampling disabled and official evaluation settings. Each benchmark score is obtained from one deterministic evaluation pass, while latency is averaged over a single pass through all 8,900 POPE examples. We do not report repeated-run uncertainty for benchmark-level scores. The utility-faithfulness intervention in the main paper instead uses paired Wilcoxon signed-rank tests over matched samples.

Experiments are conducted on a single NVIDIA RTX 5880 Ada GPU with 48 GB of memory and an Intel Core i9-14900K CPU under Ubuntu 22.04 and Linux 6.8.0. Because different backbone families rely on different official codebases and software dependencies, we use separate environments for each family. Table 8 summarizes the corresponding software environments. An anonymized implementation and evaluation scripts are included in the Code and Data Supplement. The code will be publicly released upon publication under a license permitting free use for research purposes.

A.6 Submodularity and Greedy Guarantee

For fixed nonnegative weights $w_{i,j} = c_{i,j}\hat{u}_i\hat{u}_j$, the full QCPruner objective is $F(\mathcal{K}) = \sum_{i=1}^{N_v} f_i(\mathcal{K})$, where $f_i(\mathcal{K}) = \max_{j\in\mathcal{K}} w_{i,j}$ and $f_i(\emptyset) = 0$. Thus $F(\emptyset) = 0$, and adding a representative cannot decrease any f_i, so F is normalized and monotone. For $\mathcal{A} \subseteq \mathcal{B}$ and $k \notin \mathcal{B}$, monotonicity gives $f_i(\mathcal{A}) \le f_i(\mathcal{B})$ for every target i. The marginal gains therefore satisfy

$$f_i(\mathcal{A}\cup\{k\}) - f_i(\mathcal{A}) = [w_{i,k} - f_i(\mathcal{A})]_+ \ge [w_{i,k} - f_i(\mathcal{B})]_+ = f_i(\mathcal{B}\cup\{k\}) - f_i(\mathcal{B}). \tag{22}$$

Summing over targets establishes diminishing returns, hence submodularity of F. Under the cardinality constraint $|\mathcal{K}| = K$, greedy maximization of the exact marginal gain therefore obtains at least $(1 - 1/e)$ of the optimal value of this proxy objective Nemhauser et al. (1978); the guarantee does not extend to downstream task accuracy.

B Complete Benchmark Tables

The main paper reports compact tables for readability. Here we provide the complete benchmark-wise results obtained for all compatible baseline, architecture, and retained-token combinations.

[1] `https://huggingface.co/minishlab/potion-base-8M`
[2] `https://github.com/haotian-liu/LLaVA`
[3] `https://github.com/LLaVA-VL/LLaVA-NeXT`
[4] `https://github.com/EvolvingLMMs-Lab/lmms-eval`

Method	VQAv2	GQA	VizWiz	SQA	TextVQA	POPE	MME	MMB-E	MMB-C	MMVet	Avg. Rel.
All 576 Tokens (100%)											
LLaVA-1.5	78.5	61.9	50.1	69.5	58.2	85.9	1506.5	64.7	58.1	31.3	100.0
128 Tokens (↓77.8%)											
FastV	73.2	55.4	51.4	68.1	56.4	72.3	1442.1	61.2	56.3	30.0	94.7
SparseVLM	75.3	59.4	50.1	68.6	56.7	79.6	1292.7	**63.8**	**57.9**	29.1	95.8
VisionZip	75.6	57.6	52.1	68.8	56.8	83.1	1433.3	61.3	56.7	**32.9**	97.9
DART	76.0	58.8	51.6	**69.2**	56.5	80.2	1485.5	62.5	57.4	29.2	97.2
DivPrune	76.0	59.4	52.8	68.5	55.9	87.0	1401.2	60.8	54.8	30.7	97.3
CDPruner	76.6	59.6	52.7	69.0	56.1	**87.5**	1426.5	62.4	55.1	30.2	97.9
PruneSID	75.4	58.1	52.0	68.0	54.4	84.7	1416.0	61.4	56.2	30.1	96.5
MMTok	76.4	59.2	53.0	68.9	56.8	86.5	1425.8	61.0	55.5	30.8	97.9
QCPruner-Early	76.1	59.2	**53.2**	68.6	56.6	86.9	1383.9	60.6	54.7	30.6	97.3
QCPruner	**77.7**	**61.2**	50.9	**69.2**	**57.7**	86.6	**1500.1**	63.4	57.6	30.5	**99.3**
64 Tokens (↓88.9%)											
FastV	66.3	51.6	51.5	67.3	54.7	59.5	1246.0	57.3	50.3	26.7	87.4
SparseVLM	70.2	53.7	50.1	**69.6**	53.4	77.4	1290.1	59.3	52.4	24.9	90.5
VisionZip	72.4	55.1	52.9	68.9	55.4	77.0	1364.2	59.3	55.3	**31.7**	94.9
DART	72.7	56.2	51.5	68.7	54.3	74.1	1408.6	60.9	53.8	26.7	93.0
DivPrune	74.2	57.7	**53.8**	67.9	54.5	85.5	1345.0	59.1	52.3	28.6	94.8
CDPruner	75.3	58.6	53.4	68.0	55.1	**87.5**	1403.1	60.2	53.3	28.3	96.0
PruneSID	74.1	57.1	52.4	68.4	54.1	84.3	1366.1	59.5	54.3	26.5	94.2
MMTok	75.2	58.2	**53.8**	68.8	55.8	85.6	1402.3	59.4	53.9	27.5	95.7
QCPruner-Early	74.4	57.9	**53.8**	67.7	55.3	85.8	1347.8	59.7	52.6	28.0	95.0
QCPruner	**76.9**	**60.8**	50.6	69.5	**56.8**	86.7	**1473.2**	**63.7**	**56.5**	29.1	**98.2**
32 Tokens (↓94.4%)											
FastV	57.1	46.8	40.7	65.8	51.5	40.5	987.2	50.5	41.5	21.9	74.5
VisionZip	67.3	51.7	52.7	68.6	53.1	68.7	1243.8	56.8	50.2	26.3	88.5
DART	67.9	52.9	50.3	69.1	52.0	65.3	1297.4	58.0	48.9	22.8	87.0
DivPrune	71.2	54.9	53.4	68.7	52.9	81.5	1288.0	56.8	49.1	26.8	91.4
CDPruner	73.5	56.9	53.1	69.4	53.2	**87.7**	1371.5	58.8	49.5	27.2	93.9
PruneSID	70.4	54.6	52.2	67.8	52.4	79.5	1340.1	55.3	49.4	**28.0**	91.1
MMTok	73.1	56.2	**54.5**	68.8	53.5	85.9	1350.1	58.1	49.3	27.0	93.4
QCPruner-Early	71.6	55.3	53.4	68.5	53.6	81.9	1317.6	57.5	49.8	27.4	92.3
QCPruner	**75.2**	**59.6**	50.1	**70.0**	**54.6**	86.4	**1418.6**	**62.2**	**55.6**	27.4	**96.1**

Table 9: Full benchmark-wise results on LLaVA-1.5-7B. We report every obtained benchmark-wise result for the compatible baselines under different retained patch-token targets. Avg. Rel. is computed with respect to the full-token reference.

Tables 9–12 report the full results for LLaVA-1.5-7B, LLaVA-NeXT-7B, LLaVA-Video-7B, and Qwen2.5-VL-7B under their respective retained-token budgets. Overall, QCPruner achieves strong average performance retention across the evaluated complete systems, with larger gains generally emerging under tighter token budgets.

Result overview. On LLaVA-1.5, QCPruner achieves the highest observed Avg. Rel. at all three budgets, including 96.1% at 32 tokens, 2.2 points above the strongest evaluated baseline, although some individual tasks still show regressions. On LLaVA-NeXT, the gain over the strongest evaluated baseline increases from +0.3 points at 640 tokens to +2.1 points at 160 tokens. For video, QCPruner remains competitive at 4096 tokens and achieves the highest observed average retention at 2048 and 1024 tokens among the evaluated methods under matched token budgets. Adaptive cross-frame allocation is not evaluated. On Qwen2.5-VL, QCPruner and VisionZip are nearly tied at 512 tokens, while QCPruner exceeds the strongest evaluated baseline by 4.2 and 9.9 points at 256 and 128 tokens, respectively. Across the three image backbones, QCPruner also achieves higher Avg. Rel. than MMTok at every matched budget, with the performance gap widening under tighter LLaVA-NeXT and Qwen2.5-VL budgets.

Budget-dependent pattern. QCPruner shows particularly large aggregate gains at several tighter token budgets, reaching 2.1 points on LLaVA-NeXT at 160 tokens, 2.0 points on LLaVA-Video at 1024 tokens, and 9.9 points on Qwen2.5-VL at 128 tokens. Although the gains are often larger under tighter budgets, this trend is not strictly monotonic. On LLaVA-1.5, the margins at 64 and 32 tokens are similar. The aggregate gains reflect more consistent performance retention across heterogeneous tasks rather than uniform improvements on every individual benchmark. For video, all methods are

Method	VQAv2	GQA	VizWiz	SQA-IMG	TextVQA	POPE	MME	MMB-EN	MMB-CN	MMVet	Avg. Rel.
All 2880 Tokens (100%)											
LLaVA-NeXT-7B	81.8	64.2	57.1	70.1	64.9	86.5	1519.0	67.4	60.6	40.4	100.0
640 Tokens (↓77.8%)											
FastV	78.4	61.9	55.7	68.8	60.1	84.2	1494.0	66.1	59.7	38.2	96.7
SparseVLM	79.8	62.2	54.2	68.9	60.5	86.8	1493.7	**67.8**	**60.7**	**39.4**	97.8
VisionZip	79.1	61.2	**57.5**	67.9	60.1	86.0	1462.1	65.8	58.4	39.2	96.9
DART	79.4	63.2	55.9	69.3	**60.6**	85.8	1497.3	66.1	59.3	38.6	97.5
DivPrune	79.7	61.9	54.9	68.9	56.1	86.2	1476.8	65.4	59.0	35.4	95.4
CDPruner	79.9	62.6	55.5	67.8	58.6	87.2	1467.9	65.9	57.8	38.6	96.6
PruneSID	78.6	61.7	55.3	68.1	54.7	86.1	1488.8	64.3	57.4	32.0	93.7
MMTok	79.3	62.6	55.5	68.4	58.9	87.0	**1504.2**	65.7	57.8	37.4	96.5
QCPruner-Early	79.7	62.0	55.0	68.8	56.0	86.5	1479.1	65.5	59.0	36.4	95.7
QCPruner (Ours)	**81.0**	**63.4**	55.3	**69.5**	59.3	**88.0**	1488.8	67.3	60.5	38.9	**98.1**
320 Tokens (↓88.9%)											
FastV	76.6	60.0	54.2	68.5	58.6	78.4	1367.9	64.7	56.7	36.5	93.1
SparseVLM	75.3	58.5	52.9	67.6	56.8	82.7	1414.6	63.9	56.7	33.5	92.0
VisionZip	76.2	59.0	**56.7**	67.3	**58.9**	82.1	1409.8	62.6	55.2	**39.5**	94.1
DART	78.4	61.2	54.8	68.3	58.5	83.5	1425.8	64.8	56.6	38.7	95.0
DivPrune	77.5	60.4	54.2	67.8	54.2	84.1	1451.4	64.4	57.8	33.4	93.0
CDPruner	78.4	61.4	55.2	67.4	57.3	87.2	1442.7	64.8	55.7	36.7	94.7
PruneSID	77.2	60.4	54.4	67.0	54.0	84.9	1464.0	62.8	56.6	34.5	92.9
MMTok	77.7	61.1	55.3	67.5	56.8	85.9	**1481.1**	64.1	56.6	35.5	94.4
QCPruner-Early	77.7	60.6	54.1	68.0	55.2	84.4	1423.4	64.2	57.3	33.5	93.0
QCPruner (Ours)	**80.3**	**63.2**	54.0	**68.8**	58.4	**88.4**	1447.5	**66.1**	**59.7**	37.4	**96.6**
160 Tokens (↓94.4%)											
FastV	67.8	53.5	50.7	68.2	52.6	62.3	1183.8	59.9	53.0	30.2	83.4
VisionZip	71.4	55.3	**55.9**	68.1	**56.1**	74.9	1291.7	59.6	52.8	31.6	88.0
DART	73.6	57.4	52.8	**68.5**	55.2	77.3	1370.2	60.0	53.0	31.6	88.9
DivPrune	75.0	58.8	53.5	68.2	52.6	81.5	1362.1	63.0	55.7	31.1	90.2
CDPruner	76.7	60.7	54.7	67.0	55.5	87.9	1421.8	63.6	54.5	33.5	92.8
PruneSID	74.1	58.1	54.5	67.2	52.2	80.1	1406.9	61.6	53.7	33.6	90.1
MMTok	75.7	60.0	55.8	67.5	54.6	83.9	1408.0	63.4	54.6	32.9	91.9
QCPruner-Early	75.2	59.2	53.6	67.3	53.9	82.3	1388.6	63.1	55.7	30.9	90.6
QCPruner (Ours)	**79.2**	**62.3**	52.5	**68.5**	56.0	**88.4**	**1424.3**	**65.5**	**58.8**	**35.9**	**94.9**

Table 10: Full generalization results on LLaVA-NeXT-7B. We report complete benchmark-wise results under different retained-token budgets. Best results among reported entries under each budget are in bold.

compared under matched nominal total token budgets, but FastVID uses dynamic/global allocation whereas QCPruner prunes frame-wise. Thus, the comparison controls the total token budget but not the allocation strategy.

C Ablation Studies

This section reports query-anchor construction overhead, utility-floor calibration, bilateral target–representative weighting, keyword-count sensitivity, controlled MMR/DPP selector comparisons, and pruning-layer stability under separate calibration.

C.1 Query-Anchor Construction Overhead

Query-anchor construction. The measured query-anchor construction time is **1.67 ms/sample** on LLaVA-NeXT-7B, including **1.65 ms/sample** for keyword extraction and span matching and **0.02 ms/sample** for anchor attachment. The one-time keyword-extractor loading cost is **26.89 ms** and is excluded from the per-sample latency. These measurements characterize post-initialization anchor-construction overhead and exclude cue computation, dense affinity computation, and greedy selection.

C.2 Independent Selection of the Utility Floor

Rather than tuning ρ on any benchmark in the reported evaluation suite, we select it once for each model family or setting using separate calibration data and then freeze it across downstream datasets and token budgets. We sweep $\rho \in \{0, 0.1, \ldots, 1.0\}$, with $\rho = 0$ corresponding to disabling the utility floor. The image-model sweeps use the full MMStar dataset Chen et al. (2024b) without

Method	Granularity	MVBench	LongVideoBench	Video-MME				Rel.
		test	val	w/o Sub.	Short	Medium	Long	
All 64 × 169 Patch Tokens (100%)								
LLaVA-Video-7B	–	60.8	58.9	64.3	77.3	62.4	53.2	100.0
Target: 4096 Patch Tokens, equivalent to 64 × 64 (↓62.1%)								
FastV	frame-wise	59.4	57.8	63.4	74.8	62.9	52.6	98.1
DivPrune	frame-wise	57.8	58.5	62.7	74.7	61.7	51.7	97.3
FastVID	dynamic/global	**60.5**	58.2	64.3	**76.4**	62.0	**54.6**	99.4
QCPruner (Ours)	frame-wise	60.1	**58.6**	**64.5**	76.1	**63.0**	54.2	**99.6**
Target: 2048 Patch Tokens, equivalent to 64 × 32 (↓81.1%)								
FastV	frame-wise	56.8	55.3	62.2	73.1	61.8	51.8	94.7
DivPrune	frame-wise	56.3	57.0	60.4	71.6	58.3	51.3	94.4
FastVID	dynamic/global	**59.4**	56.3	62.9	74.4	**61.9**	52.3	97.0
QCPruner (Ours)	frame-wise	59.2	**58.0**	**63.3**	**74.7**	**61.9**	**53.2**	**98.1**
Target: 1024 Patch Tokens, equivalent to 64 × 16 (↓90.5%)								
FastV	frame-wise	54.8	52.8	59.0	69.3	57.8	49.9	90.5
DivPrune	frame-wise	55.0	53.3	59.3	69.6	58.2	50.2	91.1
FastVID	dynamic/global	58.1	55.3	60.1	70.9	58.7	50.8	94.3
QCPruner (Ours)	frame-wise	**58.5**	**57.0**	**61.6**	**72.7**	**60.3**	**51.7**	**96.3**

Table 11: Complete results on LLaVA-Video-7B with 64 frames per video, evaluated on MVBench, LongVideoBench, and Video-MME. All methods are matched by nominal total patch-token budget. Non-frame-wise methods such as FastVID are matched by total rather than per-frame token counts. Best reported results within each budget are bolded. Rel. is the equal-weight average of relative scores on MVBench test, LongVideoBench val, and Video-MME without subtitles. The Short/Medium/Long columns are diagnostic breakdowns and are excluded from Rel.

Method	TextVQA	ChartQA	AI2D	OCRBench	MME	MMB-EN	MMB-CN	Avg. Rel.
All 1296 Tokens (100%)								
Qwen2.5-VL-7B	82.6	82.1	84.1	738	2332.8	83.5	80.7	100.0
512 Tokens (↓60.5%)								
FastV	**82.4**	79.3	82.5	692	2336.4	82.4	78.8	97.8
DivPrune	80.0	72.8	82.1	678	2325.5	81.9	78.4	95.7
VisionZip	80.4	**81.1**	**83.0**	698	**2356.9**	**83.6**	79.5	98.4
MMTok	78.2	75.8	82.6	667	2274.5	82.5	79.1	95.7
QCPruner (Ours)	81.6	80.5	**83.0**	**709**	2319.1	83.2	**79.6**	**98.5**
256 Tokens (↓80.2%)								
FastV	79.9	70.8	78.8	579	2299.8	80.6	76.8	92.2
DivPrune	74.2	62.7	80.5	544	2247.3	81.4	77.6	89.4
VisionZip	73.9	72.1	80.8	586	**2339.8**	81.5	78.3	92.5
MMTok	72.5	68.2	81.5	581	2268.4	71.0	77.5	89.3
QCPruner (Ours)	**80.7**	**77.2**	**82.1**	**672**	2336.1	**82.1**	**79.0**	**96.7**
128 Tokens (↓90.1%)								
FastV	74.0	54.2	71.4	420	2110.3	73.7	70.1	80.4
DivPrune	65.4	48.4	76.0	411	2125.0	76.8	74.8	80.0
VisionZip	62.8	56.6	77.4	457	2194.2	78.4	75.4	82.9
MMTok	61.7	49.8	78.1	435	2177.9	78.3	75.1	81.1
QCPruner (Ours)	**77.3**	**70.7**	**80.4**	**583**	**2336.9**	**81.6**	**78.6**	**92.8**

Table 12: Full Qwen2.5-VL-7B generalization results. Avg. Rel. denotes the average relative performance retention with respect to the full-token Qwen2.5-VL-7B reference. Best results among reported entries under each budget are in bold.

subsampling. For video, we use a fixed 600-question subset of the NExT-QA Xiao et al. (2021) `nextqa_mc_test` split, obtained with random seed 42 by stratified sampling of 200 questions from each of its three question categories. Each sweep uses the middle reported token budget: 64 tokens for LLaVA-1.5, 256 for Qwen2.5-VL, and 2048 for LLaVA-Video. The value selected on LLaVA-1.5 is transferred to LLaVA-NeXT without further tuning.

Backbone	Calib.	Budget	$\rho^\star$	$\rho = 0$	Selected	Δ
LLaVA-1.5	MMStar	64	**0.0**	0.3294	**0.3294**	+0.0000
Qwen2.5-VL	MMStar	256	**0.4**	0.5756	**0.6018**	+0.0262
LLaVA-Video	NExT-QA	2048	**0.5**	0.3783	**0.3933**	+0.0150

Table 13: Independent calibration of the utility floor on benchmarks disjoint from the reported downstream evaluation suite. $\rho^\star$ denotes the selected coefficient; $\rho = 0$ is the no-floor baseline; Selected is the calibration accuracy obtained with $\rho^\star$; and Δ is the absolute accuracy change relative to $\rho = 0$. Table 14 reports the complete sweep over $\rho \in \{0, 0.1, \ldots, 1.0\}$.

ρ	LLaVA-1.5 MMStar	Qwen2.5-VL MMStar	LLaVA-Video NExT-QA
0.0	**0.3294**	0.5756	0.3783
0.1	0.3245	0.5885	0.3883
0.2	0.3268	0.5925	0.3850
0.3	0.3273	0.5901	0.3617
0.4	0.3293	**0.6018**	0.3800
0.5	0.3159	0.6012	**0.3933**
0.6	0.3225	0.5887	0.3783
0.7	0.3179	0.5958	0.3683
0.8	0.3249	0.6005	0.3667
0.9	0.3226	0.5894	0.3333
1.0	0.3213	0.5902	0.3750

Table 14: Utility-floor calibration on separate calibration benchmarks that are not included as reported test sets. Bold indicates the deployed coefficient, and underline indicates the highest observed calibration score in each column.

We use MMStar for image calibration because it provides broad multimodal coverage while remaining disjoint from the reported downstream test suite. We use NExT-QA for video calibration because its three question categories support stratified sampling while remaining disjoint from the reported video benchmarks. For both calibration datasets, the sweep score is multiple-choice accuracy averaged over the corresponding calibration set.

For any two tokens i and j with $0 \leq \rho < 1$,

$$\hat{u}_i - \hat{u}_j = (1 - \rho)(u_i - u_j), \tag{23}$$

so the transformation preserves pairwise utility orderings while contracting score gaps by a factor of $1 - \rho$. At $\rho = 1$, all calibrated utilities equal one and the selector reduces to coverage-only selection. We include this value only as a diagnostic endpoint.

Table 13 summarizes the selected coefficients and their comparison with $\rho = 0$; Table 14 reports the complete calibration sweep. On LLaVA-1.5, disabling the utility floor with $\rho = 0$ yields the highest observed calibration score of 0.3294, narrowly above 0.3293 at $\rho = 0.4$. Because no positive value improves upon the no-floor setting, we retain $\rho = 0$ for the LLaVA image family. Qwen2.5-VL reaches its highest observed calibration score of 0.6018 at $\rho = 0.4$, while LLaVA-Video reaches 0.3933 at $\rho = 0.5$. The selected coefficients improve independent calibration accuracy by 0.0, 2.62, and 1.50 percentage points over $\rho = 0$, respectively. We keep these values fixed across the reported downstream benchmarks and token budgets.

C.3 Target- and Representative-Side Utility Weighting

We isolate the two utility factors on LLaVA-1.5-7B at 64/576 patch tokens and Qwen2.5-VL-7B at 128/1296 and 256/1296 patch tokens. For an edge from target token i to candidate representative j, we compare visual-only affinity $c_{i,j}$, target-side weighting $c_{i,j}\hat{u}_i$, representative-side weighting $c_{i,j}\hat{u}_j$, and the full bilateral form $c_{i,j}\hat{u}_i\hat{u}_j$. Within each backbone–budget setting, the query anchors, fused utility, post-layer-7 pruning position, candidate set, greedy coverage selector, and token budget are fixed. The deployed utility floor is $\rho = 0$ for LLaVA and $\rho = 0.4$ for all query-weighted Qwen settings. Visual-only rows use no utility weighting.

(a) LLaVA-1.5-7B, 64/576 tokens

Edge Weight	GQA	POPE	MME	SQA	Text	M-E	M-C	Rel.
$c_{i,j}$	60.2	82.7	1425.4	68.2	54.0	62.3	54.8	95.7
$c_{i,j}\hat{u}_i$	60.6	86.1	1448.9	68.5	55.2	62.7	56.1	97.3
$c_{i,j}\hat{u}_j$	60.3	**86.7**	1449.2	**69.6**	56.2	63.3	56.3	98.0
$c_{i,j}\hat{u}_i\hat{u}_j$ (full)	**60.8**	**86.7**	**1473.2**	69.5	**56.8**	**63.7**	**56.5**	**98.6**

(b) Qwen2.5-VL-7B

Edge Weight	Text	Chart	AI2D	OCR	MME	M-E	M-C	Rel.
256/1296 tokens								
$c_{i,j}$	77.6	70.2	81.2	640	2286.0	81.4	77.3	93.4
$c_{i,j}\hat{u}_i$	78.8	70.9	81.1	652	2293.3	**82.3**	78.1	94.3
$c_{i,j}\hat{u}_j$	80.4	**77.2**	**82.4**	**676**	2327.6	**82.3**	78.9	**96.7**
$c_{i,j}\hat{u}_i\hat{u}_j$ (full)	**80.7**	**77.2**	82.1	672	**2336.1**	82.1	**79.0**	**96.7**
128/1296 tokens								
$c_{i,j}$	70.5	60.0	78.8	526	2254.4	79.6	76.9	87.2
$c_{i,j}\hat{u}_i$	71.8	61.2	79.4	529	2242.2	80.0	77.3	87.9
$c_{i,j}\hat{u}_j$	77.2	70.1	**80.7**	**589**	2325.7	81.0	78.2	92.6
$c_{i,j}\hat{u}_i\hat{u}_j$ (full)	**77.3**	**70.7**	80.4	583	**2336.9**	**81.6**	**78.6**	**92.8**

Table 15: Utility-factor ablations across three backbone–budget settings. Text/Chart/OCR and M-E/M-C denote TextVQA/ChartQA/OCRBench and MMBench-EN/CN, respectively. Rel. denotes average relative performance retention against the corresponding full-token reference. Bold indicates the best displayed value within each backbone–budget block.

Max. Keywords	GQA	POPE	MME	Text	SQA	M-E	M-C	Rel.
2	60.6	86.8	1446.1	55.9	69.1	63.6	55.7	97.8
4	60.8	86.7	1480.9	56.6	69.2	63.5	56.8	98.6
6	60.8	86.7	1473.2	56.8	69.5	63.7	56.5	98.6
8	60.8	86.7	1474.7	57.0	69.5	63.5	56.4	98.6

Table 16: Sensitivity to the maximum number of extracted query keywords on LLaVA-1.5-7B at 64/576 patch tokens. Text/M-E/M-C abbreviate TextVQA/MMBench-EN/MMBench-CN. Rel. is the seven-task average relative performance against the unpruned model.

Table 15 reports the task-wise utility-factor ablations. On LLaVA, target-side and representative-side weighting improve Rel. from 95.7% to 97.3% and 98.0%, respectively, while the bilateral form further reaches 98.6% and improves five of seven displayed task scores over representative-only weighting. On Qwen at 256 tokens, the corresponding Rel. values are 93.4%, 94.3%, 96.7%, and 96.7%. The bilateral and representative-only forms each lead on three tasks and tie on one. At 128 tokens, the values are 87.2%, 87.9%, 92.6%, and 92.8%, while the bilateral form leads representative-only on five of seven tasks. Overall, representative-side utility provides the stronger one-sided contribution across all three settings. Bilateral weighting yields an additional aggregate gain on LLaVA and Qwen-128, while its incremental benefit over representative-only weighting remains modest and setting-dependent.

C.4 Sensitivity to the Number of Query Keywords

On LLaVA-1.5-7B at 64/576 patch tokens, we vary only the maximum keyword count while fixing the extractor, span matching, post-layer-7 pruning position, equal-weight cue fusion, and coverage selector.

Table 16 shows that Avg. Rel. increases from 97.8% with two keywords to 98.6% with four keywords and remains at 98.6% at the reported precision with six or eight. Thus, performance is stable once at least four keywords are available, and the default maximum of six does not depend on a narrow optimum.

(a) Qwen2.5-VL-7B

Selector	Tokens	Text	Chart	AI2D	OCR	MME	M-E	M-C	Rel.
Full	1296	82.6	82.1	84.1	738	2332.8	83.5	80.7	100.0
Top-K	128	49.1	25.1	75.4	185	2217.9	79.3	74.4	69.6
MMR	128	75.2	62.2	79.2	513	2315.6	80.2	76.8	88.7
CDPruner-DPP	128	77.0	62.8	79.4	533	2286.2	79.5	75.3	89.0
MMTok-style	128	76.0	62.4	80.1	566	2286.2	79.3	76.5	89.7
Visual-only Cov.	128	70.5	60.0	78.8	526	2254.4	79.6	76.9	87.2
QCPruner	128	**77.3**	**70.7**	**80.4**	**583**	**2336.9**	**81.6**	**78.6**	**92.8**
Top-K	256	63.5	38.7	79.3	310	2260.9	**82.2**	76.5	78.6
MMR	256	78.8	71.5	81.5	599	2306.3	82.1	78.0	93.5
CDPruner-DPP	256	79.7	71.2	81.4	615	2318.4	81.1	77.4	93.7
MMTok-style	256	80.4	72.6	82.0	**677**	2272.3	80.8	77.7	95.1
Visual-only Cov.	256	77.6	70.2	81.2	640	2286.0	81.4	77.3	93.4
QCPruner	256	**80.7**	**77.2**	**82.1**	672	**2336.1**	82.1	**79.0**	**96.7**

(b) LLaVA-1.5-7B

Selector	Tokens	GQA	POPE	Text	M-E	M-C	MME	SQA	Rel.
Full	576	61.9	85.9	58.2	64.7	58.1	1506.5	69.5	100.0
Top-K	64	60.8	86.2	53.5	62.2	56.4	1398.9	68.9	96.5
MMR	64	60.8	86.6	56.3	63.1	**57.0**	1459.5	**69.6**	98.3
CDPruner-DPP	64	**61.2**	86.6	**56.9**	63.4	56.8	1467.3	69.3	**98.6**
MMTok-style	64	60.5	85.8	56.7	63.5	56.0	1434.2	69.4	97.8
Visual-only Cov.	64	60.2	82.7	54.0	62.3	54.8	1425.4	68.2	95.7
QCPruner	64	60.8	**86.7**	56.8	**63.7**	56.5	**1473.2**	69.5	**98.6**

Table 17: Controlled selector comparison across Qwen2.5-VL-7B and LLaVA-1.5-7B. Text/Chart/OCR and M-E/M-C denote TextVQA/ChartQA/OCRBench and MMBench-EN/CN, respectively. Rel. denotes seven-task average relative performance retention with respect to the corresponding full-token reference. Candidate patches, query anchors, fused cues, pruning position, utility floor, and token budget are fixed within each backbone–budget setting. Only the subset selector changes. Bold indicates the best selector result within each budget. Visual-only Cov. is an internal control, while MMTok-style is an objective-level backend transplant and is distinct from the complete MMTok system.

C.5 Controlled Comparison of Subset Selectors

Visual-token MMR. For R candidates, let $K' = \min(K, R)$ and normalize QCPruner's fused query utility as

$$\bar{r}_i = \frac{r_i - r_{\min}}{\max(r_{\max} - r_{\min}, \varepsilon)}, \qquad \varepsilon = 10^{-6}.$$

We apply the same backbone-level utility floor used by the corresponding QCPruner setting,

$$q_i^{\mathrm{MMR}} = \rho_b + (1 - \rho_b)\bar{r}_i,$$

with $\rho_b = 0.4$ for Qwen2.5-VL and $\rho_b = 0$ for LLaVA-1.5. The visual-similarity matrix is the same candidate-wise affinity matrix used by QCPruner. Let $\widetilde{C} = [\mathrm{float32}(C)]_+$ elementwise and $d_i^{(0)} = 0$. The controlled MMR selector is

$$\begin{aligned} j_t &= \arg\max_{i \notin \mathcal{K}_{t-1}} \left[0.5 q_i^{\mathrm{MMR}} - 0.5 d_i^{(t-1)}\right], \\ d_i^{(t)} &= \max\{d_i^{(t-1)}, \widetilde{C}_{i,j_t}\}, \qquad t = 1, \ldots, K'. \end{aligned} \tag{24}$$

Thus, MMR uses the same query utility, visual affinity, candidate set, and token budget as QCPruner, while replacing only the subset-selection rule.

MMTok-style objective transplant. For the Qwen2.5-VL and LLaVA-1.5 selector comparisons, we additionally replace the subset objective with MMTok's additive text–vision and vision–vision coverage formulation Dong et al. (2026), while retaining the same candidate patches, post-layer-7

pruning position, query anchors, upstream representations, and token budgets. The objective form and all MMTok-specific hyperparameters follow the official configuration without modification or task-specific retuning. Because this transplant operates on QCPruner's in-decoder inputs rather than MMTok's original pre-decoder pipeline, we denote it *MMTok-style* and distinguish it from the complete MMTok system in the benchmark tables.

CDPruner-DPP transplant. For the controlled selector comparison, we transplant CDPruner's conditional-DPP/Fast-MAP backend Zhang et al. (2025) while fixing the candidate patches, post-layer-7 pruning position, keyword-matched anchors, fused query utility, and token budget. We use $K \in \{128, 256\}$ for Qwen2.5-VL and $K = 64$ for LLaVA-1.5. This *CDPruner-DPP* setting is a selector-level transplant and is distinct from the complete CDPruner system reported in the benchmark tables. Its DPP/Fast-MAP selection logic follows the official CDPruner implementation without algorithmic modification.[5]

Let $S \in \mathbb{R}^{R\times R}$ denote the candidate-wise visual-similarity matrix and u_i the fused query utility. The transplanted quality mapping and DPP kernel are

$$\begin{aligned} \bar{u}_i &= \frac{u_i - u_{\min}}{\max(u_{\max} - u_{\min}, \varepsilon)}, \\ q_i &= \rho + (1-\rho)\bar{u}_i, \\ L^{\mathrm{DPP}} &= \mathrm{diag}(\mathbf{q}) S \,\mathrm{diag}(\mathbf{q}), \end{aligned} \tag{25}$$

where $\rho = \rho_b$ and $\varepsilon = 10^{-6}$. Thus, QCPruner utility replaces CDPruner's native relevance signal, while the DPP similarity treatment, determinant objective, and Fast-MAP subset-selection backend remain unchanged.

Table 17 reports the controlled selector results. On Qwen2.5-VL, QCPruner achieves 92.8 and 96.7 Rel. at 128 and 256 tokens, exceeding CDPruner-DPP by 3.8 and 3.0 points and MMTok-style by 3.1 and 1.6 points, respectively. QCPruner leads all seven displayed tasks at 128 tokens and five at 256 tokens. MMTok-style and Top-K lead OCRBench and MMBench-EN, respectively, at the latter budget. On LLaVA-1.5 at 64 tokens, CDPruner-DPP and QCPruner both reach 98.6 Rel. at the reported precision, compared with 97.8 for MMTok-style. The task-level leaders are distributed across QCPruner, CDPruner-DPP, and MMR on three, two, and two tasks, respectively. Compared with visual-only coverage, query-weighted coverage improves Rel. by 5.6, 3.3, and 2.9 points at Qwen-128, Qwen-256, and LLaVA-64, respectively. Because the visual affinity, coverage selector, and token budget are fixed in these comparisons, these gains isolate the contribution of query weighting within the coverage formulation. Overall, QCPruner achieves higher average retention than the controlled MMTok-style backend in all three settings and than the DPP backend on Qwen, while QCPruner and DPP perform similarly on LLaVA.

C.6 Pruning-Layer Stability on Separate Calibration Data

Before evaluating the reported downstream test benchmarks, we analyze pruning-layer stability on all 1,500 MMStar samples using LLaVA-1.5-7B at 64/576 patch tokens. We use unpruned forward passes to maintain a fixed visual-token population across layers, while keeping the query anchors, equal-weight cue fusion, and scoring procedure fixed with $\rho = 0$. All layer indices use zero-based decoder indexing. Thus, the deployed pruning position at $\ell = 7$ is after decoder layer 7 and before layer 8.

For sample n, let $\mathbf{u}_n^\ell$ denote the fused utility vector at layer ℓ. We define a sample-specific late-layer consensus proxy as

$$\bar{\mathbf{u}}_n^{\mathrm{late}} = \frac{1}{3}\sum_{r=9}^{11} \mathbf{u}_n^r. \tag{26}$$

Layer-wise rank stability is measured by the average within-sample Spearman correlation,

$$S_\ell = \frac{1}{1500}\sum_{n=1}^{1500} \mathrm{Spearman}\big(\mathbf{u}_n^\ell, \bar{\mathbf{u}}_n^{\mathrm{late}}\big). \tag{27}$$

[5] https://github.com/Theia-4869/CDPruner

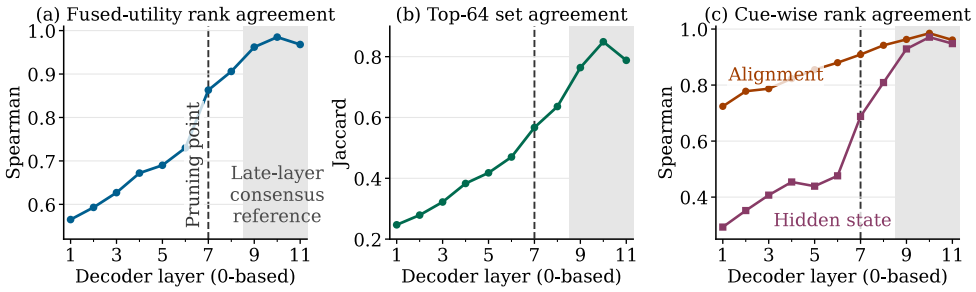

Figure 2: Full layer-wise calibration trajectories on 1,500 MMStar Chen et al. (2024b) examples. (a) Fused-utility Spearman agreement, (b) Top-64 Jaccard agreement, and (c) alignment- and hidden-state-cue Spearman agreement with their corresponding per-sample late-layer consensus proxies. The dashed line marks decoder layer 7. The deployed hook removes tokens after layer 7 and before layer 8. The shaded layers 9–11 constitute the same-model late-layer consensus reference and are displayed for completeness rather than treated as independent validation points. All indices are implementation-level and zero-based.

Here, $\mathrm{Spearman}(\mathbf{a}, \mathbf{b})$ is the Pearson correlation between their rank-transformed entries, so it measures agreement in visual-token ordering rather than raw score scale, with larger values indicating more stable rankings. We compute this correlation within each sample and then average it across the 1,500 samples.

To measure agreement among the highest-utility tokens, let $\mathcal{T}_n^\ell = \mathrm{Top64}(\mathbf{u}_n^\ell)$ and $\mathcal{T}_n^{\mathrm{late}} = \mathrm{Top64}(\bar{\mathbf{u}}_n^{\mathrm{late}})$. We compute the average per-sample Jaccard overlap

$$J_\ell = \frac{1}{1500} \sum_{n=1}^{1500} \frac{|\mathcal{T}_n^\ell \cap \mathcal{T}_n^{\mathrm{late}}|}{|\mathcal{T}_n^\ell \cup \mathcal{T}_n^{\mathrm{late}}|}. \tag{28}$$

Cue-wise correlations are computed analogously using the corresponding cue average over layers 9–11 as the consensus proxy. Figure 2 visualizes the full trajectories, and Table 18 reports the corresponding numerical values.

Layer ℓ	Fused Spear.	Top-64 Jac.	Align. Spear.	Hidden Spear.
1	0.565	0.247	0.724	0.293
2	0.593	0.279	0.778	0.352
3	0.627	0.322	0.787	0.407
4	0.672	0.383	0.823	0.454
5	0.690	0.418	0.855	0.439
6	0.730	0.470	0.880	0.476
7	**0.863**	**0.567**	**0.909**	**0.688**
8	0.906	0.636	0.942	0.809
9	0.962	0.764	0.963	0.929
10	0.985	0.849	0.985	0.971
11	0.968	0.788	0.961	0.948

Table 18: Mean layer-wise agreement over 1,500 MMStar samples. Fused and cue-specific Spearman correlations use the corresponding per-sample layer-9–11 consensus proxies, and Top-64 reports Jaccard overlap. Bold marks the deployed pruning position at layer 7 rather than the maximum value. Gray rows form the consensus reference and are not independent evaluation points. All layer indices are zero-based.

Across all 1,500 calibration samples, Figure 2 shows that fused-utility agreement with the same-model layers 9–11 consensus has its largest early increase from layer 6 to 7, with Spearman rising from 0.730 to 0.863 and Top-64 Jaccard from 0.470 to 0.567. At layer 8, fused Spearman and Top-64 Jaccard rise further to 0.906 and 0.636, respectively. Later layers reach higher agreement, as expected given that layers 9–11 define the late-layer consensus proxy.

Layer	TextVQA	ChartQA	AI2D	OCRBench	MME	MMB-EN	MMB-CN	Avg. Rel.
5	80.3	75.6	81.8	634	2309.2	82.6	78.3	95.3
6	79.6	74.1	81.5	636	2336.5	82.0	79.0	95.2
7	80.7	77.2	82.1	672	2336.1	82.1	79.0	96.7
8	80.6	74.5	81.9	654	2309.3	82.6	78.8	95.7
9	81.3	77.4	82.0	679	2333.3	82.1	79.3	**97.0**

Table 19: Qwen2.5-VL-7B pruning-layer sensitivity at 256/1296 visual tokens. All layer indices are zero-based. The gray row is the fixed layer-7 configuration used in the main comparisons; the other rows are layer-only variants. MME is the sum of perception and cognition scores. Avg. Rel. equally averages the seven task-wise ratios against the full-token reference in Table 12.

The cue decomposition further clarifies this transition. Alignment-cue stability is already high by layer 5 and increases smoothly from 0.855 to 0.942 across layers 5–8. In contrast, hidden-state similarity rises sharply from 0.476 at layer 6 to 0.688 at layer 7 (+0.212). This stronger change in the semantic cue coincides with the increase in fused-ranking stability at layer 7.

We therefore choose layer 7 as the earliest post-transition stability–efficiency operating point, allowing one additional decoder layer to process the pruned sequence relative to layer 8. The consensus is an internal proxy rather than independent validation. This pruning position is fixed for the main benchmark comparisons across backbone–budget settings without test-set retuning.

Downstream sensitivity on Qwen2.5-VL. We additionally vary only the zero-based pruning hook across layers 5–9 on Qwen2.5-VL-7B at 256/1296 visual tokens, retaining the same seven evaluation tasks and other QCPruner settings. Table 19 gives the per-task scores; Avg. Rel. uses the full-token Qwen2.5-VL reference in Table 12.

Layer 7 exceeds layers 5, 6, and 8 in Avg. Rel. by 1.4, 1.5, and 1.0 percentage points, respectively. Layer 9 improves Avg. Rel. by 0.3 points over layer 7, although it leaves two additional decoder layers to process the dense sequence before pruning. In particular, the lower layer-8 score shows that stronger agreement with the late-layer calibration proxy does not necessarily yield higher downstream performance at every adjacent hook. The fixed layer-7 choice thus reflects the earlier transition in the separate calibration analysis and the cost of later dense processing, rather than the maximum score in this Qwen2.5-VL sweep.

C.7 Cue and Query-Anchor Ablations at 32 and 64 Tokens

We compare cue and anchor variants at 32/576 and 64/576 patch tokens while fixing the post-layer-7 hook, population-coverage selector, and all other settings. Tables 20 and 21 report the complete per-task results.

LLaVA-1.5-7B, Retain 32 Tokens								
Configuration	**SQA**	**GQA**	**POPE**	**MME**	**Text**	**M-E**	**M-C**	**Rel.**
Unpruned	69.5	61.9	85.9	1506.5	58.2	64.7	58.1	100.0
Sim.+Anc.	68.4	59.2	**86.5**	1413.1	53.5	62.0	54.7	95.8
QK+Anc.	69.0	58.7	83.4	1424.1	54.8	61.9	54.6	95.6
Fusion+Prompt	68.2	58.9	86.1	**1425.5**	**55.1**	62.0	55.0	96.2
Fusion+Anc.	**70.0**	**59.6**	86.4	1418.6	54.6	**62.2**	**55.6**	**96.8**

Table 20: Cue and query-anchor ablations on LLaVA-1.5-7B at 32/576 patch tokens. Anc. denotes matched anchors. Text/M-E/M-C abbreviate TextVQA/MMBench-EN/MMBench-CN. Rel. is seven-task Avg. Rel. Bold marks the best pruned result.

Table 21 shows that the fused-anchor variant reaches 98.6 Avg. Rel., compared with 98.1 for QK alignment alone and 97.4 for hidden-state similarity alone. Thus, equal-weight fusion provides a modest aggregate gain at this budget, while the task-level leaders remain distributed across variants. Replacing matched anchors with the full prompt reduces Avg. Rel. from 98.6 to 98.3.

Configuration	SQA	GQA	POPE	MME	Text	M-E	M-C	Rel.
Sim.+Anc.	69.1	60.1	**86.7**	1463.7	55.2	62.6	55.4	97.4
QK+Anc.	69.5	60.4	85.4	**1478.1**	56.3	**63.7**	56.1	98.1
Fusion+Prompt	**69.6**	60.1	86.3	1470.7	56.6	63.4	**56.6**	98.3
Fusion+Anc.	69.5	**60.8**	**86.7**	1473.2	**56.8**	**63.7**	56.5	**98.6**

Table 21: Cue and query-anchor ablations on LLaVA-1.5-7B at 64/576 patch tokens. Anc. denotes keyword-matched anchors. Text/M-E/M-C abbreviate TextVQA/MMBench-EN/MMBench-CN. Rel. is the seven-task average relative performance against the unpruned model. Bold marks the best pruned result.

D Transferability of Query Anchors

To examine whether keyword-matched query anchors can transfer beyond QCPruner, we integrate them into SparseVLM, an existing text-guided visual token pruning method. Specifically, we replace SparseVLM's original textual condition with our keyword-matched anchors while keeping the remaining pruning pipeline unchanged. This controlled replacement isolates the contribution of textual conditioning within the SparseVLM pipeline.

As shown in Table 22, replacing SparseVLM's original textual condition with keyword-matched anchors improves all seven benchmarks and increases Avg. Rel. from 90.9% to 91.9%. Overall, these results show that the proposed query anchors are not tied to QCPruner's coverage selector and can transfer to another text-guided pruning pipeline.

E Limitations and Broader Impact

Limitations. QCPruner relies on keyword-matched anchors produced by an external extractor with English stop-word filtering. Short, highly referential, multilingual, or OCR-oriented prompts may therefore provide incomplete textual conditions, although the deterministic fallback reduces complete matching failures. The population-coverage selector also requires dense pairwise affinity computation and direct $O(KN_v^2)$ greedy selection, which introduces additional pruning overhead. Moreover, the shared pruning position is selected using LLaVA-1.5 on MMStar and then transferred across evaluated backbones, while the utility floor remains calibrated at the model-family level rather than fixed globally. Performance gains are also task- and setting-dependent, with occasional regressions on individual benchmarks. Finally, downstream QA retention does not directly measure region-level grounding quality, which remains an important direction for further evaluation.

Broader Impact. This work aims to improve the practicality of multimodal large language models by retaining compact visual-token subsets that preserve query-relevant information. Reducing the visual-token sequence processed by later decoder layers may lower model-side computation while maintaining strong task performance. However, pruning can alter model behavior when relevant visual evidence is removed. Applications requiring high reliability, including medical, legal, and other safety-critical settings, should therefore validate pruning behavior carefully before deployment.

Method	GQA	POPE	SQA	Text	MME	M-E	M-C	Rel.
SparseVLM	53.7	77.4	69.6	53.4	1290.1	59.3	52.4	90.9
+ anchors	**54.0**	**78.6**	**69.7**	**53.5**	**1330.0**	**60.3**	**52.8**	**91.9**
Δ	+0.3	+1.2	+0.1	+0.1	+39.9	+1.0	+0.4	+1.0

Table 22: Transfer of keyword-matched query anchors to SparseVLM on LLaVA-1.5-7B at 64/576 patch tokens. Avg. Rel. is computed over the seven displayed benchmarks.

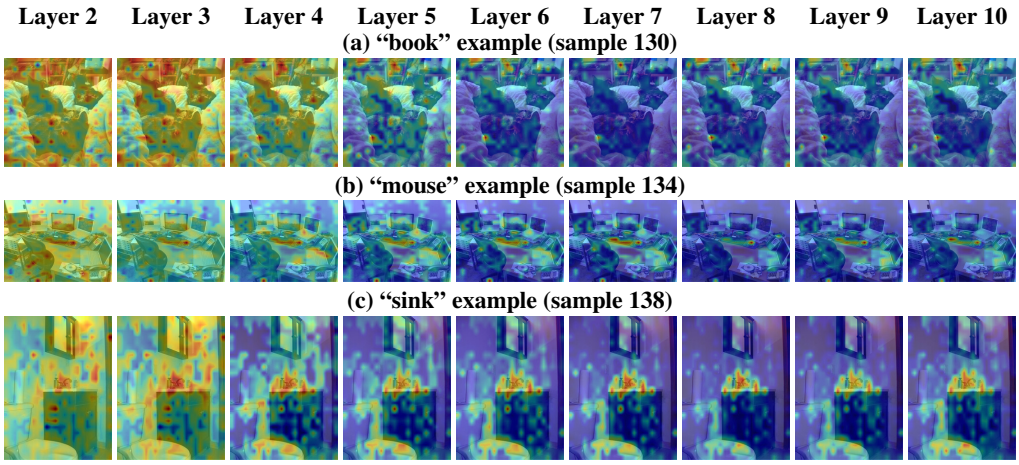

Figure 3: Layer-wise fused responses for the “book,” “mouse,” and “sink” examples. Each heading denotes the response after the named decoder layer, with layers 2–10 running from left to right. Red and yellow denote larger displayed responses within each independently normalized map. These examples illustrate the layer transition rather than provide a grounding metric.

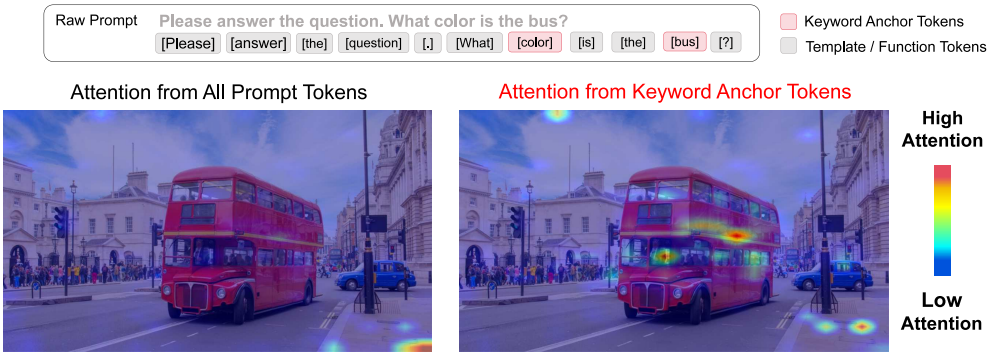

Figure 4: Query-anchor motivation. Full-prompt alignment is compared with alignment conditioned on matched content phrases. Each heat map is independently normalized for visualization.

F Qualitative Motivation Examples

These examples complement the quantitative analyses by visualizing layer-wise query-conditioned responses, query-anchor conditioning, and the complementary behavior of the two utility cues. Each heat map is independently normalized and is intended for qualitative interpretation.

F.1 Qualitative View of the Layer Transition

Figure 3 visualizes the equal-weight fused query-conditioned response across successive decoder layers for three examples. Early-layer responses are comparatively diffuse, while later layers show more concentrated response patterns. This qualitative evolution is consistent with the increase in ranking stability observed around layer 7 in Appendix C.6.

F.2 Query Anchors

Prompts mix task-relevant content with system scaffolding, punctuation, and generic instruction words. As illustrated in Figure 4, full-prompt alignment can be spatially dispersed, whereas matched content phrases such as “color” and “bus” produce more focused query-conditioned responses. These visualizations motivate using keyword-matched phrases as focused textual anchors while retaining the fallback mechanism described in Appendix A.5.

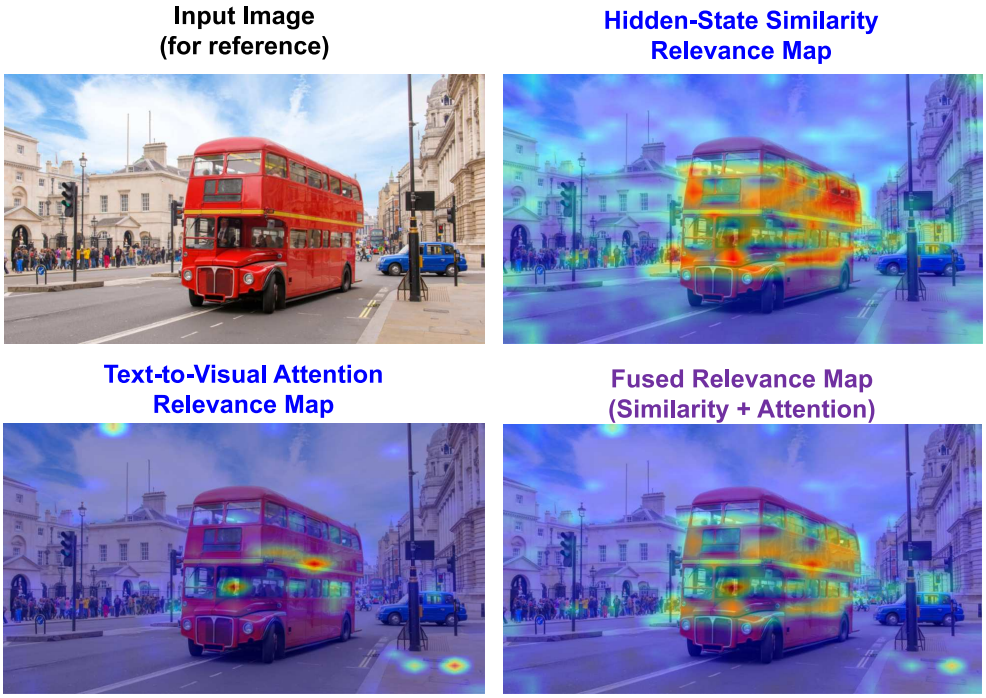

Figure 5: Complementary query-conditioned cues. Hidden-state cosine similarity and head-averaged pre-softmax query–key alignment exhibit distinct response patterns, which QCPruner combines after normalization using equal weights.

F.3 Complementary Query-Conditioned Cues

Figure 5 contrasts hidden-state cosine similarity, head-averaged pre-softmax query–key alignment, and their equal-weight fusion for the same query anchors. The two cues exhibit complementary response patterns: similarity tends to distribute responses across semantically related regions, whereas alignment can be more concentrated around a smaller set of locations. QCPruner therefore combines their normalized responses with equal weights, consistent with the quantitative cue ablation reported in the main paper.