RoRA: Role-Oriented Regional Allocation for Visual Token Pruning in MLLMs

Anonymous authors
Paper under double-blind review

Abstract

Multimodal large language models (MLLMs) encode images as long visual token sequences, making prefilling and KV-cache storage expensive. Existing training-free pruning methods select tokens by importance, diversity, or spatial coverage, but treat retained tokens as interchangeable and do not explicitly track which object-related regions are already covered. We present RoRA, a training-free framework that casts visual token pruning as role-oriented regional evidence allocation. Given a fixed budget, RoRA partitions tokens into a protected semantic core, complementary context, and fine-grained detail. It first calibrates text-conditioned attention with a positional prior and a prompt-calibrated object prior, then builds Attention-Anchored Regions (AARs) from high-confidence anchors as lightweight proxies for covered object support. Context is explored mainly outside AARs, while a small AAR-guided budget restores local detail; pairwise similarity is used only for context-stage redundancy filtering. Under matched budgets, RoRA consistently outperforms strong training-free baselines across LLaVA and Qwen-VL families, retaining most of the unpruned accuracy even at aggressive pruning ratios, *e.g.*, 96.5% of full performance at 88.9% pruning on LLaVA-1.5, and improving over D^2Pruner by about 5% on Qwen3-VL at 75–90% pruning. At a 66.7% pruning ratio, RoRA requires only 0.7ms for token selection and reduces end-to-end inference time by 24.6%, corresponding to a 1.33× speedup over unpruned inference on an NVIDIA H800.

1 Introduction

Multimodal large language models (MLLMs) represent images as long sequences of visual tokens. High-resolution images, multi-image inputs, and videos can yield hundreds to thousands of tokens, increasing prefilling cost, KV-cache size, and inference latency. Training-free visual token pruning aims to preserve task-relevant information using only a small subset of these tokens under a fixed retention budget.

Text-conditioned attention provides a natural signal for token selection, but often exhibits positional bias toward image borders or other systematic hotspots (Zou et al., 2026). Attention-only selection, as in FastV (Chen et al., 2024), can therefore retain tokens clustered around biased regions while missing key semantic evidence (Figure 1). Subsequent methods seek to preserve informative tokens that attention-only selection would discard.

Existing remedies fall into two main branches. The first promotes diversity and spatial coverage without explicitly correcting attention bias: DART (Wen et al., 2025) selects tokens relative to pivots, while HoloV (Zou et al., 2026) allocates crop-wise quotas using attention and feature variance. Although these strategies improve coverage, they remain agnostic to each token's semantic role. In particular, high local variance can give cluttered background crops high scores, causing HoloV to allocate tokens to spectators or texture-rich edges rather than the queried object (Figure 1). The second branch corrects positional bias before token ranking. D^2Pruner (Zhang et al., 2026) combines debiased attention with a dense token–token similarity graph (*e.g.*, 576×576) to suppress redundancy. This yields more reliable rankings, but global pairwise modeling is costly and largely structure-agnostic: without tracking regions already represented by object evidence, it must compare almost every token pair to approximate coverage.

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



Figure 1: Qualitative comparison of FastV, HoloV, and RoRA under the same visual token budget. FastV is affected by positional attention sinks, while HoloV's crop-wise allocation assigns tokens to high-variance but task-irrelevant regions. In contrast, RoRA uses AARs to allocate tokens across semantic core, complementary context, and fine-grained detail, thereby preserving the evidence needed to identify the jersey number and document year correctly.

Across both branches, tokens are selected by importance, diversity, or spatial coverage, but not by the *role* they should play under a fixed budget. We argue that multimodal queries require three distinct yet complementary types of visual evidence. First, high-confidence tokens must preserve the queried object (*core*); without them, the model loses the primary referent. Second, once the core is secured, additional tokens are more useful as scene or secondary-object cues beyond its support (*context*) than as near-duplicates around the same object. Third, fine-grained tokens are needed for text, boundaries, and small parts (*detail*), which coarse importance ranking or crop-level quotas can miss. Because these roles complement rather than replace one another, a homogeneous selection criterion or crop quotas that ignore object coverage cannot deliberately preserve all three.

We propose **RoRA**, a training-free framework for role-oriented regional evidence allocation. RoRA combines attention debiasing with a prompt-calibrated object prior to protect a semantic *core*. High-confidence anchors and their local neighborhoods on the 2D token grid then form Attention-Anchored Regions (AARs), lightweight spatial proxies for object-related support already covered by the core rather than exact object masks. RoRA allocates the remaining budget asymmetrically: it explores complementary *context* mainly outside AARs and preserves *detail* mainly inside them. Explicit coverage modeling enables core-referenced redundancy checks for candidate context tokens, avoiding a full-image pairwise similarity matrix. This yields clearer evidence allocation and substantially lower selection overhead (Figure 1).

We evaluate RoRA on LLaVA-1.5, LLaVA-NeXT, Qwen2.5-VL, and Qwen3-VL. Under equal retention budgets, RoRA consistently achieves higher normalized average scores than prior training-free methods, with negligible selection overhead, *e.g.*, 0.7ms per image. Our contributions are summarized as follows:

- **Role-aware budget allocation.** We organize retained visual tokens into three complementary roles, and apply different selection criteria under a shared budget.
- **Object region prior.** Beyond debiasing, we exploit a prompt-calibrated signal to identify object-centric regions and stabilize semantic-core protection.
- **Attention-anchored regions for structured pruning.** AARs identify object-related spatial support, enabling outside-AAR context exploration and inside-AAR detail preservation with core-referenced redundancy checks, without requiring dense global similarity graphs.
- **Extensive experiments.** Comparisons and ablations on several MLLMs verify RoRA's accuracy–efficiency gains.

2 Related Work

Visual Token Compression in MLLMs. Visual token compression has become an important technique for reducing the high inference cost of MLLMs caused by long visual sequences. Existing approaches can be performed before visual features enter the language model or during LLM inference. Vision-side methods reduce tokens through pooling, query-based compression, or token merging before multimodal fusion (Li et al., 2023a; Yao et al., 2024; Bolya et al., 2022; Shang et al., 2025; Chen et al., 2025). However, these methods generally lack direct interaction with the textual instruction. In contrast, training-free LLM-side pruning exploits text-conditioned signals during inference to remove visual tokens after they have acquired multimodal semantics (Chen et al., 2024; Wen et al., 2025; Zou et al., 2026; Zhang et al., 2026). We focus on this setting and aim to improve visual token selection under a fixed retention budget.

Training-Free Visual Token Selection. Existing training-free visual token selection methods mainly rely on three types of signals: **(i) Attention-based methods.** FastV and related approaches estimate token importance from text-conditioned attention (Chen et al., 2024; Zhang et al., 2025a). However, original text-conditioned attention can contain positional bias, causing important tokens to be confused with model-specific attention patterns. D^2Pruner addresses this issue by introducing attention debiasing and combining it with structural diversity modeling (Zhang et al., 2026). **(ii) Redundancy-aware methods.** Instead of relying only on importance scores, redundancy-aware methods select tokens according to feature similarity or duplication. DART and DivPrune reduce repeated visual representations by encouraging diverse token subsets (Wen et al., 2025; Alvar et al., 2025). However, these approaches mainly focus on redundancy reduction and do not explicitly distinguish the semantic roles of retained tokens. **(iii) Region-aware methods.** HoloV improves spatial coverage by dividing images into regions and allocating tokens according to regional attention and feature variance (Zou et al., 2026). Nevertheless, fixed regional allocation does not explicitly model which object-related evidence has already been covered. Different from previous methods, RoRA treats retained tokens as different evidence roles and performs role-aware allocation among semantic core, complementary context, and fine-grained detail.

3 Methods

In this section, we present RoRA, a training-free framework for role-oriented regional evidence allocation. As illustrated in Figure 2, RoRA first calibrates text-conditioned visual relevance and protects a high-confidence semantic core. It then constructs Attention-Anchored Regions (AARs) to represent object-related spatial support already covered by reliable evidence and allocates the remaining budget toward complementary context. Finally, a small-budget detail-repair stage restores fine-grained visual cues.

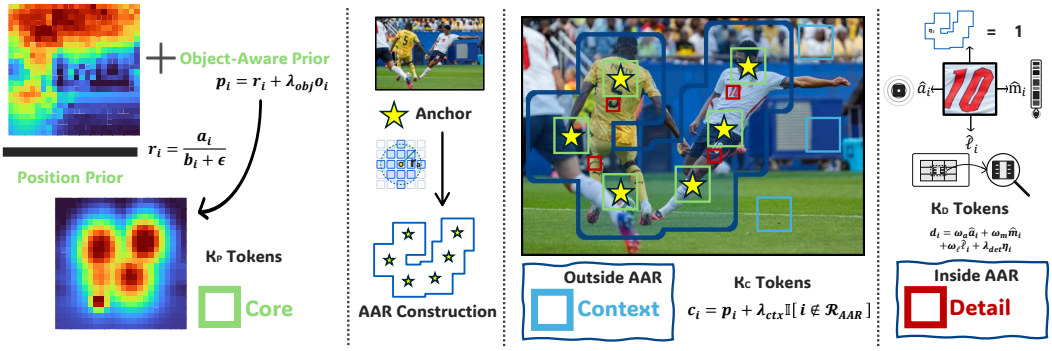


Figure 2: **Overview of RoRA.** RoRA decomposes the retained visual token budget into a **protected semantic core**, **complementary context**, and **fine-grained details**. **Attention-Anchored Regions** represent object-related spatial support already covered by high-confidence evidence, guiding outside-AAR context exploration and inside-AAR detail repair.

3.1 RoRA Framework Overview

RoRA is applied at an early LLM layer, where visual tokens have acquired text-conditioned information while pruning can still reduce the sequence to be processed by all subsequent layers (Chen et al., 2024; Zhang et al., 2026). Let ℓ_p denote the pruning layer. At this LLM layer, the hidden states of visual tokens can be defined as:

$$\mathcal{H}_V = \{\mathbf{h}_i\}_{i=1}^{N}. \tag{1}$$

RoRA retains a total budget of K for visual tokens. Unlike other methods with one homogeneous selection criterion for the entire visual sequence, it assigns the retained visual tokens with three distinct and non-interchangeable evidence roles:

$$\mathcal{S} = \mathcal{P} \dot{\cup} \mathcal{C} \dot{\cup} \mathcal{D}, \qquad |\mathcal{S}| = K, \qquad K_p + K_c + K_d = K. \tag{2}$$

In this decomposition, $\mathcal{S}$ denotes the final set of retained visual tokens, while $\mathcal{P}$ encapsulates protected semantic-core tokens that preserve the primary evidence required for query answering. The complementary-context set $\mathcal{C}$ subsumes secondary objects, relational structures, and background context that fall outside the semantic core. Meanwhile, the detail-token set $\mathcal{D}$ recovers fine-grained spatial evidence, including text, small objects, boundaries, and local textures. The symbol $\dot{\cup}$ denotes a disjoint union, so each retained token belongs to exactly one of the three sets. Here, $K_p = |\mathcal{P}|$, $K_c = |\mathcal{C}|$, and $K_d = |\mathcal{D}|$ denote their respective token counts, while K is the total retained-token budget. Thus, $|\mathcal{S}| = K$ and $K_p + K_c + K_d = K$.

3.2 Object-Calibrated Attention and Core Protection

Original text-conditioned attention provides an initial estimate of the correlation between the current text instruction and visual content. However, such methods are vulnerable to positional bias, which systematically favors visual tokens at specific spatial positions. Following the debiasing principle of D^{2}Pruner (Zhang et al., 2026), we calibrate the instance-level attention with an offline positional-attention prior. RoRA further incorporates a soft, prompt-calibrated object prior that smoothly steers attention toward regions more likely to contain object-centric evidence.

In practice, we sample 1,000 images from the GQA (Hudson & Manning, 2019) training split as an unlabeled calibration subset, using only images without questions, answers, or category annotations. For all calibration images, we use the same prompt, "Please describe the image.", to estimate the model's positional bias under generic image-description scenarios and obtain a general positional prior. We then repeat the same procedure with the object-inspection prompt, "List all visible objects in the image.", to derive relatively weak object-aware prior scores for tokens at each visual position.

This prior is intentionally designed as a subtle regularization term, which is motivated by the fact that object locations naturally vary across images and an excessively strong prior could compromise generalizability. Yet, a gentle prior suffices to encourage the model to focus on regions with higher semantic likelihood. Specifically, let a_i denote the attention assigned to the i-th visual token, b_i the positional-bias prior, and o_i the object-aware prior. The calibrated attention score is defined as:

$$r_i = \mathrm{Norm}\left(\frac{a_i}{b_i + \epsilon}\right), \qquad p_i = \mathrm{Norm}(r_i + \lambda_{\mathrm{obj}} o_i). \tag{3}$$

Here, ϵ is a numerical stabilizer, $\mathrm{Norm}(\cdot)$ denotes the token-wise score-normalization operation, and λ_{obj} controls the strength of object-aware calibration. Because the object prior is only a weak positional tendency rather than an image-specific object mask, λ_{obj} is intentionally set to a modest value to ensure that sample-specific attention remains the dominant signal. We then select the K_p tokens with the highest p_i scores to form the protected semantic-core set:

$$\mathcal{P} = \mathrm{TopK}_{K_p}\big(\{p_i\}_{i=1}^{N}\big). \tag{4}$$

These tokens constitute the semantic backbone of the compressed visual sequence. They are excluded from the candidate pools of subsequent allocation stages and preserved in the final retained set.

3.3 ATTENTION-ANCHORED REGIONAL ALLOCATION

The protected semantic core tokens preserve the most reliable semantic evidence. Consequently, selecting the remaining tokens solely by calibrated attention risks over-allocating the token budget to regions already sufficiently covered. RoRA therefore employs high-confidence tokens as spatial anchors, explicitly modeling the coverage of core evidence to ensure broader context diversity

We select M_a region anchors according to calibrated attention and expand their local neighborhoods on the two-dimensional visual token grid:

$$\mathcal{A} = \mathrm{TopK}_{M_a}\big(\{p_i\}_{i=1}^{N}\big), \qquad \mathcal{R}_{\mathrm{AAR}} = \bigcup\nolimits_{a\in\mathcal{A}} \mathcal{N}_{r_a}(a). \tag{5}$$

For each anchor, $\mathcal{N}_{r_a}(a)$ represents the local grid region centered at a with radius r_a, and $\mathcal{R}_{\mathrm{AAR}}$ denotes the union of all anchor neighborhoods.

Rather than enforcing a fixed token quota for each region, Attention-Anchored Regions (AAR) dynamically adjusts the scores of complementary-context candidates. Once primary object evidence is covered by $\mathcal{P}$, candidates outside AAR receive a small exploration boost, encouraging the remaining budget to cover secondary objects, spatial relations, and scene context. Based on AAR, we define the complementary-context score of the i-th token as

$$c_i = \mathrm{Norm}\big(p_i + \lambda_{\mathrm{ctx}}\mathbb{I}[\, i \notin \mathcal{R}_{\mathrm{AAR}}\,]\big). \tag{6}$$

The coefficient λ_{ctx} controls the exploration strength for candidates outside AAR, and $\mathbb{I}[\cdot]$ denotes the indicator function. When a token falls outside the AAR, its context score receives a small bonus, preventing the remaining token budget from over-concentrating in salient regions that are already adequately covered.

To further reduce redundancy, we use semantic similarity as a lightweight redundancy filter. If the semantic similarity between a candidate token and one already retained token exceeds the threshold τ_{sem}, this candidate will be skipped, thereby preventing repeated visual evidence from consuming the budget. The resulting complementary-context set is:

$$\mathcal{C} = \mathrm{Select}_{\tau_{\mathrm{sem}},K_c}\big(\{c_i \mid i \notin \mathcal{P}\}\big). \tag{7}$$

The operator $\mathrm{Select}_{\tau_{\mathrm{sem}}}$ ranks candidate tokens by descending c_i and skips any candidate whose cosine similarity to the currently retained tokens exceeds τ_{sem}, until K_c complementary-context tokens have been selected.

3.4 AAR-GUIDED DETAIL REPAIR

While the semantic core and complementary context preserve primary objects, relations, and scene information, fine-grained components—such as text, small objects, boundaries, and local textures—occupy few visual patches and often exhibit low global relevance. We therefore employ a small-budget detail-repair stage to recover this evidence.

Let $\widehat{a}_i$, $\widehat{m}_i$, and $\widehat{\ell}_i$ denote the attention response, feature magnitude, and local contrast of the i-th visual token, respectively:

$$\begin{aligned} \widehat{a}_i = \mathrm{Norm}(a_i), \qquad \widehat{m}_i = \mathrm{Norm}(\|\mathbf{h}_i\|_2)\,, \\ \Delta_{ij} = 1 - \cos(\mathbf{h}_i, \mathbf{h}_j), \qquad \widehat{\ell}_i = \mathrm{Norm}\left(\tfrac{1}{|\mathcal{N}_8(i)|}\sum\nolimits_{j\in\mathcal{N}_8(i)} \Delta_{ij}\right). \end{aligned} \tag{8}$$

For each visual token, $\mathcal{N}_8(i)$ denotes its eight neighborhood on the two-dimensional visual token grid. The attention response represents the attention received by the current visual token; the feature magnitude represents the strength of its hidden-state representation; and the local contrast measures the average feature difference between the current token and its neighboring tokens. Based on these quantities, we define an AAR indicator η_i and the detail score of the i-th visual token as

$$\eta_i = \mathbb{I}[\, i \in \mathcal{R}_{\mathrm{AAR}}\,], \qquad d_i = \omega_a\widehat{a}_i + \omega_m\widehat{m}_i + \omega_\ell\widehat{\ell}_i + \lambda_{\mathrm{det}}\eta_i. \tag{9}$$

The coefficients ω_a, ω_m, and ω_ℓ control the contributions of attention response, feature magnitude, and local contrast, respectively, while λ_{det} provides an additional bonus to detail tokens within AAR.

Importantly, AAR is not a hard constraint but a soft preference. We select the K_d highest-scoring tokens that have not already been assigned to the semantic-core or complementary-context sets:

$$\mathcal{D} = \mathrm{TopK}_{K_d}\big(\{d_i \mid i \notin \mathcal{P} \cup \mathcal{C}\}\big). \tag{10}$$

Finally, the protected semantic core $\mathcal{P}$, complementary context $\mathcal{C}$, and detail tokens $\mathcal{D}$ jointly form the retained visual token set $\mathcal{S}$.

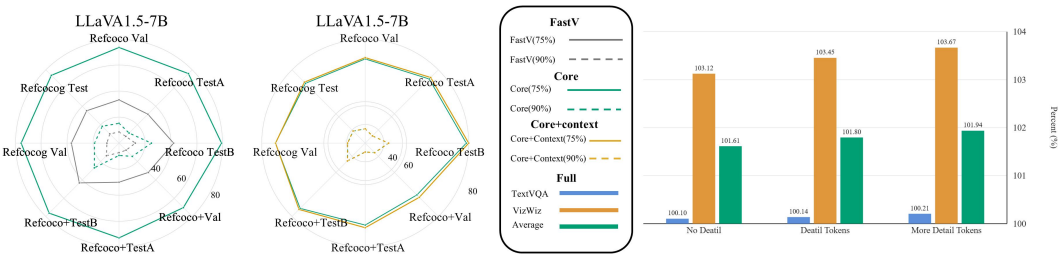


Figure 3: Component-wise ablation on LLaVA-1.5-7B. Left: effects of semantic-core protection and complementary-context allocation on eight RefCOCO-family localization splits at 75% and 90% pruning. Right: ablation about detail tokens at $K = 128$ on TextVQA and VizWiz.

4 Experiments

4.1 Experimental Settings

Models and baselines. We evaluate RoRA on LLaVA-1.5-7B, LLaVA-NeXT-7B, Qwen2.5-VL-7B-Instruct, and Qwen3-VL-8B-Instruct (Liu et al., 2024a;b; Bai et al., 2025b;a). Baselines include ToMe, FastV, MustDrop, LLaVA-PruMerge, PDrop, FiCoCo-V, HiRED, FitPrune, VisionZip, SparseVLM, FasterVLM, DART, DivPrune, HoloV, and D^{2}Pruner (Bolya et al., 2022; Chen et al., 2024; Liu et al., 2024c; Shang et al., 2025; Xing et al., 2025; Han et al., 2026; Arif et al., 2025; Ye et al., 2025; Yang et al., 2025; Zhang et al., 2024; 2025a; Wen et al., 2025; Alvar et al., 2025; Zou et al., 2026; Zhang et al., 2026).

Benchmarks. LLaVA models are evaluated on GQA, MMBench, MMBench-CN, MME, POPE, ScienceQA, VQAv2, TextVQA, and VizWiz (Hudson & Manning, 2019; Liu et al., 2024d; Fu et al., 2026; Li et al., 2023b; Lu et al., 2022; Goyal et al., 2017; Singh et al., 2019; Gurari et al., 2018). Qwen2.5-VL uses AI2D, MME, TextVQA, NaturalBench, MME-RealWorld-CN, TextCaps, and ScienceQA-IMG (Kembhavi et al., 2016; Fu et al., 2026; Singh et al., 2019; Li et al., 2024; Zhang et al., 2025b; Sidorov et al., 2020; Lu et al., 2022); Qwen3-VL uses BLINK-counting, IllusionVQA, VisRes, POPE, TextVQA, CV-Bench, and MUIRBench (Fu et al., 2024; Shahgir et al., 2024; Törtei et al., 2026; Li et al., 2023b; Singh et al., 2019; Tong et al., 2024; Wang et al., 2025). Ablations additionally use the RefCOCO family (Kazemzadeh et al., 2014; Mao et al., 2016). Unless otherwise specified, we report the mean of benchmark scores normalized by their corresponding unpruned results.

Implementation details. Each backbone–pruning-ratio pair uses a fixed RoRA configuration across all tasks, without benchmark-specific tuning. Baselines follow their official configurations. Complete hyperparameter settings are reported in the appendix.

4.2 Main Results

Results on LLaVA models. The upper block of Table 1 reports the results on **LLaVA-1.5-7B**. With visual token budgets of $K = 192$, 128, and 64, RoRA retains 99.8%, 99.1%, and 96.5% of the unpruned model's normalized average performance, respectively, achieving the best result among all compared methods at every matched budget. Moreover, the relative advantage of RoRA increases as pruning becomes more aggressive, indicating that role-aware regional allocation preserves visual evidence more effectively under severely constrained token budgets.

Table 1: Performance comparisons on LLaVA-1.5-7B and LLaVA-Next-7B across several understanding benchmarks.

Method	GQA	MMB	MMB-CN	MME	POPE	SQA	VQA^{V2}	VQA^{Text}	VizWiz	Avg.
LLaVA-1.5-7B	*Upper Bound, 576 Tokens* **(100%)**									
Vanilla	61.9	64.7	58.1	1862	85.9	69.5	78.4	58.2	50.0	100.0%
LLaVA-1.5-7B	*Retain 192 Tokens* (↓ 66.7%)									
ToMe (ICLR23)	54.3	60.5	-	1563	72.4	65.2	68.0	52.1	-	88.5%
FastV (ECCV24)	52.7	61.2	57.0	1612	64.8	67.3	67.1	52.5	50.8	90.5%
MustDrop (arXiv24)	58.2	62.3	55.8	1787	82.6	69.2	76.0	56.5	51.4	97.2%
LLaVA-PruMerge (ICCV25)	54.3	59.6	52.9	1632	71.3	67.9	70.6	54.3	50.1	91.4%
PDrop (CVPR25)	57.1	63.2	56.8	1766	82.3	68.8	75.1	56.1	51.1	96.7%
VisionZip (CVPR25)	59.3	**64.5**	**57.3**	1767	**86.4**	68.9	76.8	57.3	51.6	98.1%
SparseVLM (ICML25)	57.6	62.5	53.7	1721	83.6	69.1	75.6	56.1	50.5	96.1%
DART (arxiv25)	60.3	64.1	56.3	1839.8	85.2	**69.4**	76.5	57.9	51.3	99.3%
HoloV (NeurIPS25)	58.7	63.5	53.6	1782.9	86.3	68.5	74.8	56.1	**51.8**	97.5%
D²Pruner (AAAI25)	60.9	64.2	55.7	1854.7	85.5	69.1	77.0	58.6	51.2	99.5%
RoRA (Ours)	**61.2**	**64.5**	56.2	**1861.0**	85.5	69.1	**77.1**	**58.7**	51.4	**99.8%**
LLaVA-1.5-7B	*Retain 128 Tokens* (↓ 77.8%)									
ToMe (ICLR23)	52.4	53.3	-	1343	62.8	59.6	63.0	49.1	-	80.4%
FastV (ECCV24)	49.6	56.1	56.4	1490	59.6	60.2	61.8	50.6	51.3	85.4%
MustDrop (arXiv24)	56.9	61.1	55.2	1745	78.7	68.5	74.6	56.3	**52.1**	95.7%
LLaVA-PruMerge (ICCV25)	53.3	58.1	51.7	1554	67.2	67.1	68.8	54.3	50.3	89.4%
PDrop (CVPR25)	56.0	61.1	56.6	1644	82.3	68.3	72.9	55.1	51.0	94.9%
VisionZip (CVPR25)	57.6	63.4	**56.7**	1768	84.7	68.8	75.6	56.8	52.0	97.2%
SparseVLM (ICML25)	56.0	60.0	51.1	1696	80.5	67.1	73.8	54.9	51.4	93.8%
DART (arxiv25)	58.7	**63.8**	54.9	1826	85.0	69.1	75.2	56.5	51.6	98.1%
HoloV (NeurIPS25)	57.3	63.2	52.8	1757.8	84.0	67.8	73.5	56.1	52.0	96.3%
D²Pruner (AAAI25)	**60.0**	63.4	54.3	1846.8	85.7	69.1	**76.3**	58.3	51.5	98.8%
RoRA (Ours)	**60.0**	63.7	54.9	**1849.6**	**86.1**	**69.2**	76.2	**58.4**	51.8	**99.1%**
LLaVA-1.5-7B	*Retain 64 Tokens* (↓ 88.9%)									
ToMe (ICLR23)	48.6	43.7	-	1138	52.5	50.0	57.1	45.3	-	70.1%
FastV (ECCV24)	46.1	48.0	52.7	1256	48.0	51.1	55.0	47.8	50.8	76.7%
MustDrop (arXiv24)	53.1	60.0	53.1	1612	68.0	63.4	69.3	54.2	51.2	90.1%
LLaVA-PruMerge (ICCV25)	51.9	55.3	49.1	1549	65.3	68.1	67.4	54.0	50.1	87.7%
PDrop (CVPR25)	41.9	33.3	50.5	1092	55.9	68.6	69.2	45.9	50.7	77.5%
VisionZip (CVPR25)	55.1	60.1	**55.4**	1690	77.0	69.0	72.4	55.5	52.9	94.5%
SparseVLM (ICML25)	52.7	56.2	46.1	1505	75.1	62.2	68.2	51.8	50.1	87.3%
DART (arxiv25)	56.1	61.6	53.4	1755	82.8	**69.1**	72.0	52.7	52.0	95.2%
HoloV (NeurIPS25)	55.1	60.0	51.3	1696.8	80.5	68.9	70.8	54.6	**53.2**	94.1%
D²Pruner (AAAI25)	56.9	**62.4**	52.4	1771.9	83.3	69.0	73.2	56.2	51.7	96.2%
RoRA (Ours)	**57.5**	62.0	53.1	**1773.2**	**84.2**	69.0	**73.8**	**56.3**	51.6	**96.5%**
LLaVA-Next-7B	*Upper Bound, 2880 Tokens* **(100%)**									
Vanilla	64.2	67.4	60.6	1851	86.5	70.1	81.8	64.9	57.6	100.0%
LLaVA-Next-7B	*Retain 320 Tokens* (↓ 88.9%)									
FastV (ECCV24)	55.9	61.6	51.9	1661	71.7	62.8	71.9	55.7	53.1	88.0%
LLaVA-PruMerge (ICCV25)	53.6	61.3	55.3	1534	60.8	66.4	69.7	50.6	54.0	85.6%
PDrop (CVPR25)	56.4	63.4	**56.2**	1663	77.6	67.5	73.5	54.4	54.1	90.9%
MustDrop (arXiv24)	57.3	62.8	55.1	1641	82.1	68.0	73.7	59.9	54.0	92.2%
FasterVLM (ICCV25)	56.9	61.6	53.5	1701	83.6	66.5	74.0	56.5	52.6	91.1%
SparseVLM (ICML25)	56.1	60.6	54.5	1533	82.4	66.1	71.5	58.4	52.0	89.7%
DART (arxiv25)	61.1	**66.3**	44.1	1699.9	87.7	68.7	77.6	58.4	54.8	95.2%
HoloV (NeurIPS25)	59.2	65.1	47.6	**1750.2**	87.4	66.5	76.3	55.7	**55.3**	94.9%
D²Pruner (AAAI25)	61.5	66.1	41.7	1712.7	88.0	68.4	**78.4**	60.6	54.3	95.1%
RoRA (Ours)	**61.6**	66.0	42.3	1713.6	**88.1**	**68.9**	**78.4**	**60.7**	54.9	**95.5%**

The lower block of Table 1 further presents results on **LLaVA-NeXT-7B**, whose any-resolution visual encoder produces a dynamic number of visual tokens according to the input resolution. When retaining a fixed budget of $K = 320$ visual tokens, RoRA achieves a normalized average performance of 95.5%, outperforming all compared baselines.

Results on Qwen models. Table 2 reports results on the Qwen series. On Qwen2.5-VL-7B, RoRA achieves normalized averages of 106.9% and 96.7% at 75% and 90% pruning, respectively, outperforming all compared baselines. On Qwen3-VL-8B, it reaches 98.0% and 87.9%, surpassing D^{2}Pruner by 5.4 and 5.3 points, respectively. These results demonstrate that RoRA generalizes consistently across newer dynamic-resolution MLLMs.

4.3 Ablation Studies

We evaluate the semantic core and complementary context on localization benchmarks, as these components are designed to preserve the queried object and its surrounding spatial evidence. The

Table 2: Performance comparisons on Qwen2.5-VL-7B-Instruct and Qwen3-VL-8B-Instruct.

Method	Qwen2.5-VL-7B								Qwen3-VL-8B							
	AI2D	MME	T-VQA	NB	RW	TC	SQA	Avg.	BLK	Ill	VR	POPE	T-VQA	CV	MUIR	Avg.
	Upper Bound, All Tokens **(100%)**								*Upper Bound, All Tokens* **(100%)**							
Vanilla	74.7	2246.6	76.4	24.0	27.1	53.5	73.0	100.0%	69.1	53.7	25.6	89.1	80.9	85.9	50.9	100.0%
	Retain 25% Tokens in Average ($\downarrow 75\%$)								*Retain 25% Tokens in Average* ($\downarrow 75\%$)							
HoloV	71.0	1915.2	64.7	21.8	21.7	46.2	74.8	89.2%	58.3	51.0	26.5	85.0	57.1	81.0	48.4	91.1%
FastV	71.3	1915.3	73.3	23.2	30.6	**65.6**	74.1	101.4%	58.3	**53.1**	24.0	85.0	69.5	81.8	48.4	92.6%
DART	69.3	1900.7	66.9	25.4	27.8	54.4	73.2	96.4%	55.0	51.9	25.9	85.5	69.0	81.6	49.7	93.0%
DivPrune	21.5	2014.4	69.8	25.1	26.1	58.4	73.4	88.6%	59.1	49.2	24.5	**87.8**	**72.7**	81.3	**50.5**	93.6%
D^2Pruner	73.3	2080.3	73.6	28.4	31.7	60.9	75.9	105.6%	58.3	49.2	24.6	85.8	70.4	82.0	49.7	92.6%
RoRA	**73.9**	**2106.0**	**73.8**	**29.4**	**32.4**	61.3	**76.0**	**106.9%**	**62.5**	**53.1**	**29.5**	86.6	72.4	**82.8**	50.4	**98.0%**
	Retain 10% Tokens in Average ($\downarrow 90\%$)								*Retain 10% Tokens in Average* ($\downarrow 90\%$)							
HoloV	64.6	1760.3	38.7	14.0	17.3	37.5	**75.2**	72.9%	45.8	49.4	25.8	75.0	40.7	73.4	45.0	81.0%
FastV	65.8	1288.9	65.0	11.6	23.5	**66.1**	70.7	83.6%	45.8	48.2	22.9	75.0	46.3	74.0	45.0	80.2%
DART	64.5	1610.3	56.1	17.5	24.3	44.7	72.3	82.4%	45.0	49.8	23.8	77.8	49.0	72.4	47.9	82.4%
DivPrune	19.9	1592.4	56.9	17.7	19.3	44.3	69.2	70.6%	48.3	46.6	26.0	**83.3**	**59.2**	74.4	**48.5**	86.6%
D^2Pruner	**72.5**	1778.5	69.5	18.3	27.1	63.9	73.6	94.8%	45.0	49.2	23.0	78.9	49.8	76.6	47.1	82.6%
RoRA	**72.5**	**1780.6**	**69.6**	**20.8**	**27.7**	64.3	73.9	**96.7%**	**49.2**	**50.8**	**28.6**	79.4	54.2	**77.1**	47.0	**87.9%**

detail stage is evaluated on TextVQA and VizWiz, which are more sensitive to OCR cues, small objects, and localized visual details. Localization results are normalized by our unpruned model under the same prompting, parsing, and IoU-based evaluation protocol, while the detail results are normalized by the corresponding vanilla upper bounds.

Semantic core and complementary context. Figure 3 reports localization results on eight RefCOCO-family splits (Kazemzadeh et al., 2014; Mao et al., 2016). All variants retain the same number of visual tokens. At $K = 144$, the normalized average increases from 35.44% for FastV (Chen et al., 2024) to 74.44% after introducing the protected semantic core. At the more aggressive $K = 58$ setting, the average similarly increases from 9.57% to 17.36%. Core outperforms FastV on every split at both budgets, confirming that protecting calibrated object-related evidence is essential for localization.

Adding AAR-guided complementary context further improves the average from 74.44% to 75.03% at $K = 144$, and from 17.36% to 17.47% at $K = 58$. It improves seven of eight splits at $K = 144$ and introduces no degradation at $K = 58$. Although the gain is smaller than that of the semantic core, it shows that allocating part of the fixed budget to previously uncovered spatial evidence provides additional localization support.

AAR-guided detail repair. We evaluate detail repair at $K = 128$ on TextVQA (Singh et al., 2019) and VizWiz (Gurari et al., 2018). Without detail tokens, the normalized scores are 100.10% on TextVQA and 103.12% on VizWiz, with an average of 101.61%. Adding the standard detail configuration increases the average to 101.80%, while allocating 18 detail tokens further improves it to 101.94%. The monotonic gains on both benchmarks indicate that detail repair recovers fine-grained evidence not fully captured by the semantic-core and context stages. Its improvement is modest, which is consistent with its intended role as a small-budget complementary module.

4.4 Efficiency Analysis

We evaluate LLaVA-1.5-7B on full POPE (9,000 yes-or-no questions) using one NVIDIA H800 GPU, batch size 1, and max new tokens 1. This prefill-dominated protocol measures visual-token selection and LLM prefilling overhead, with $K = 192$ and $K = 58$ reported in Table 3. Table 4 further evaluates LLaVA-NeXT-7B on one RTX PRO 6000 GPU; due to its much longer visual sequences, selector overhead accounts for a smaller share of total runtime.

At $K = 192$, RoRA completes the full evaluation in 5min43s and achieves an average per-sample latency of $0.038\,\mathrm{s}$, both of which are the lowest measured values among the compared methods. Its selector overhead is only $0.704\,\mathrm{ms}$, while it retains 98.4% of the unpruned model's POPE perfor-

Table 3: Real inference comparison on LLaVA-1.5 and POPE at 66.7% and 90% pruning ratios, measured on a single NVIDIA H800 GPU with batch size 1. All competing methods are evaluated under the same settings using their official open-source implementations.

Methods	Time	Select	Latency	Acc.	Time	Select	Latency	Acc.
Upper Bound, 576 Tokens	7:35	–	0.05s	100%	7:35	–	0.05s	100%
LLaVA-1.5-7B	*Retain 192 Tokens* (↓ **66.70%**)				*Retain 58 Tokens* (↓ **90%**)			
FastV (ECCV24)	6:36	0.1ms	0.04s	75.4%	6:44	0.1ms	0.05s	66.8%
D^{2}Pruner (AAAI26)	19:58	71.7ms	0.13s	96.8%	19:53	68.1ms	0.13s	87.7%
Faster (ICCV25)	5:51	0.1ms	0.04s	**99.2%**	5:48	0.1ms	0.04s	92.3%
MustDrop (arXiv24)	7:24	0.2ms	0.05s	95.4%	9:29	0.2ms	0.06s	86.6%
DART (arXiv25)	6:17	3.2ms	0.04s	97.5%	6:09	2.0ms	0.04s	90.6%
HoloV (NeurIPS25)	5:48	1.0ms	0.04s	98.9%	5:38	2.4ms	0.04s	92.4%
RoRA (Ours)	**5:43**	0.7ms	0.04s	98.4%	**5:22**	0.6ms	0.04s	**92.5%**

Table 4: Speed comparison on LLaVA-Next-7B and POPE under 66.7% and 90% pruning ratios. Prefilling time, total runtime, FLOPs, and KV cache are reported on a single RTX PRO 6000 GPU with batch size 1 and max new tokens 1.

Methods	Forward latency ↓ (ms/sample)	Generation latency ↓ (ms/sample)	FLOPs ↓ (T)	KV Cache ↓ (MB)
Upper Bound, 2880 Tokens	132.8 (1.00×)	133.2 (1.00×)	16.9 (1.00×)	1512 (1.00×)
LLaVA-Next-7B				
D^{2}Pruner (AAAI25) (33.4%)	126.3 (1.05×)	125.9 (1.06×)	6.0 (2.82×)	526 (2.87×)
RoRA (Ours, 33.4%)	**108.1 (1.23×)**	**106.8 (1.25×)**	6.0 (2.82×)	526 (2.87×)
D^{2}Pruner (AAAI25) (11.2%)	69.0 (1.92×)	68.9 (1.93×)	2.8 (6.10×)	198 (7.63×)
RoRA (Ours, 11.2%)	**66.5 (2.00×)**	**65.7 (2.03×)**	2.8 (6.10×)	198 (7.63×)

mance. Compared with the unpruned runtime of 7 min 35 s, this corresponds to a 24.6% reduction in end-to-end inference time. Under the more aggressive $K = 58$ setting, RoRA further reduces the total runtime to 5 min 22 s. It achieves not only the lowest inference latency among all compared methods, but also the best accuracy among the compressed models, retaining 92.5% of the unpruned performance.

RoRA is efficient because token similarity is used only as a lightweight redundancy filter in complementary-context selection: residual candidates highly similar to retained evidence are skipped, instead of injecting pairwise similarity into the primary importance score. Let $N_t < N$ be the residual candidates after fixing the semantic core, and C the number of HoloV crops. RoRA has complexity $\mathcal{O}(N + N_t^2)$, which is much lighter in practice than D^{2}Pruner's $\mathcal{O}(N^2)$ full-graph construction and comparable to HoloV's crop-wise $\mathcal{O}(N^2/C)$.

5 Conclusion

In this work, we presented RoRA, a novel training-free visual token pruning framework that views context reduction as role-aware regional evidence allocation. By organizing visual tokens into a protected semantic core, complementary context, and fine-grained detail evidence, RoRA explicitly improves coverage of retained tokens through lightweight Attention-Anchored Regions (AARs). This design enables asymmetric budget allocation, expanding complementary evidence outside attended regions while recovering local details inside them, without constructing costly dense pairwise redundancy graphs. Extensive evaluations across the multiple representative MLLM architectures, including LLaVA and Qwen-VL families show that RoRA achieves a superior accuracy–efficiency profile, consistently outperforming existing training-free baselines across different models even at aggressive pruning ratios.

AI USE STATEMENT

We used generative AI tools to polish selected sentences in the manuscript. We also used these tools to assist with code writing.

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.

Kazi Hasan Ibn Arif, JinYi Yoon, Dimitrios S Nikolopoulos, Hans Vandierendonck, Deepu John, and Bo Ji. Hired: Attention-guided token dropping for efficient inference of high-resolution vision-language models. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 39, pp. 1773–1781, 2025.

Shuai Bai, Yuxuan Cai, Ruizhe Chen, Keqin Chen, Xionghui Chen, Zesen Cheng, Lianghao Deng, Wei Ding, Chang Gao, Chunjiang Ge, et al. Qwen3-vl technical report. *arXiv preprint arXiv:2511.21631*, 2025a.

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, 2025b. URL https://arxiv.org/abs/2502.13923.

Daniel Bolya, Cheng-Yang Fu, Xiaoliang Dai, Peizhao Zhang, Christoph Feichtenhofer, and Judy Hoffman. Token merging: Your vit but faster. *arXiv preprint arXiv:2210.09461*, 2022.

Liang Chen, Haozhe Zhao, Tianyu Liu, Shuai Bai, Junyang Lin, Chang Zhou, and Baobao Chang. An image is worth 1/2 tokens after layer 2: Plug-and-play inference acceleration for large vision-language models. In *European Conference on Computer Vision*, pp. 19–35. Springer, 2024.

Yuan Chen, Zichen Wen, Yuzhou Wu, Xuyang Liu, Shuang Chen, Junpeng Ma, Weijia Li, Conghui He, and Linfeng Zhang. Ipcv: Information-preserving compression for mllm visual encoders. *arXiv preprint arXiv:2512.18747*, 2025.

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. *Advances in Neural Information Processing Systems*, 38, 2026.

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

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

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.

Yuhang Han, Xuyang Liu, Zihan Zhang, Pengxiang Ding, Junjie Chen, Honggang Chen, Donglin Wang, Qingsen Yan, and Siteng Huang. Filter, correlate, compress: Training-free token reduction for mllm acceleration. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 40, pp. 4601–4609, 2026.

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.

Sahar Kazemzadeh, Vicente Ordonez, Mark Matten, and Tamara Berg. Referitgame: Referring to objects in photographs of natural scenes. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing*, pp. 787–798. Association for Computational Linguistics, 2014.

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

Baiqi Li, Zhiqiu Lin, Wenxuan Peng, Jean de Dieu Nyandwi, Daniel Jiang, Zixian Ma, Simran Khanuja, Ranjay Krishna, Graham Neubig, and Deva Ramanan. Naturalbench: Evaluating vision-language models on natural adversarial samples. *Advances in Neural Information Processing Systems*, 37:17044–17068, 2024.

Junnan Li, Dongxu Li, Silvio Savarese, and Steven Hoi. Blip-2: Bootstrapping language-image pre-training with frozen image encoders and large language models. In *International conference on machine learning*, pp. 19730–19742. PMLR, 2023a.

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, 2023b.

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. Llavanext: Improved reasoning, ocr, and world knowledge, 2024b.

Ting Liu, Liangtao Shi, Richang Hong, Yue Hu, Quanjun Yin, and Linfeng Zhang. Multi-stage vision token dropping: Towards efficient multimodal large language model. *arXiv preprint arXiv:2411.10803*, 2024c.

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, 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.

Junhua Mao, Jonathan Huang, Alexander Toshev, Oana Camburu, Alan L. Yuille, and Kevin Murphy. Generation and comprehension of unambiguous object descriptions. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 11–20, 2016.

Haz Sameen Shahgir, Khondker Salman Sayeed, Abhik Bhattacharjee, Wasi Uddin Ahmad, Yue Dong, and Rifat Shahriyar. Illusionvqa: A challenging optical illusion dataset for vision language models. *arXiv preprint arXiv:2403.15952*, 2024.

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.

Oleksii Sidorov, Ronghang Hu, Marcus Rohrbach, and Amanpreet Singh. Textcaps: a dataset for image captioning with reading comprehension. In *European conference on computer vision*, pp. 742–758. Springer, 2020.

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.

Shengbang Tong, Ellis Brown, Penghao Wu, Sanghyun Woo, Manoj Middepogu, Sai C Akula, Jihan Yang, Shusheng Yang, Adithya Iyer, Xichen Pan, et al. Cambrian-1: A fully open, vision-centric exploration of multimodal llms. *Advances in Neural Information Processing Systems*, 37:87310–87356, 2024.

Brigitta Malagurski Törtei, Yasser Dahou, Ngoc Dung Huynh, Wamiq Reyaz Para, Phúc H Lê Khac, Ankit Singh, Sofian Chaybouti, and Sanath Narayan. Visres bench: On evaluating the visual reasoning capabilities of vlms. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 33185–33195, 2026.

Fei Wang, Xingyu Fu, James Y Huang, Zekun Li, Qin Liu, Xiaogeng Liu, Mingyu Derek Ma, Nan Xu, Wenxuan Zhou, Kai Zhang, et al. Muirbench: A comprehensive benchmark for robust multi-image understanding. In *International Conference on Learning Representations*, volume 2025, pp. 62624–62650, 2025.

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

Long Xing, Qidong Huang, Xiaoyi Dong, Jiajie Lu, Pan Zhang, Yuhang Zang, Yuhang Cao, Conghui He, Jiaqi Wang, Feng Wu, et al. Conical visual concentration for efficient large vision-language models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 14593–14603, 2025.

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.

Linli Yao, Lei Li, Shuhuai Ren, Lean Wang, Yuanxin Liu, Xu Sun, and Lu Hou. Deco: Decoupling token compression from semantic abstraction in multimodal large language models. *arXiv preprint arXiv:2405.20985*, 2024.

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.

Evelyn Zhang, Fufu Yu, Aoqi Wu, Zichen Wen, Ke Yan, Shouhong Ding, Biqing Qi, and Linfeng Zhang. D^2pruner: Debiased importance and structural diversity for mllm token pruning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 40, pp. 12412–12420, 2026.

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

YiFan Zhang, Huanyu Zhang, Haochen Tian, Chaoyou Fu, Shuangqing Zhang, Junfei Wu, Feng Li, Kun Wang, Qingsong Wen, Zhang Zhang, et al. Mme-realworld: Could your multimodal llm challenge high-resolution real-world scenarios that are difficult for humans? In *International Conference on Learning Representations*, volume 2025, pp. 89655–89701, 2025b.

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.

Xin Zou, Di Lu, Yizhou Wang, Yibo Yan, Yuanhuiyi Lyu, Xu Zheng, Linfeng Zhang, and Xuming Hu. Don't just chase "highlighted tokens" in mllms: Revisiting visual holistic context retention. *Advances in Neural Information Processing Systems*, 38:39800–39832, 2026.

A Overall Implementation

This appendix records implementation details for RoRA. RoRA is inserted at an early LLM pruning layer ℓ_p after the visual tokens have been projected into the language-model token space. The selector only removes visual tokens. All system, instruction, and generated text positions are preserved, then the retained visual tokens are merged back with the non-visual sequence positions.

Let the visual token span at the pruning boundary be $\mathcal{V} = \{1, \ldots, N\}$ with hidden states $\mathcal{H}_V = \{h_i\}_{i=1}^{N}$. RoRA first computes calibrated core scores p_i, protects semantic-core tokens $\mathcal{P}$, builds an Attention-Anchored Region (AAR) as a soft regional prior from high-confidence anchors, allocates complementary context tokens $\mathcal{C}$, and finally repairs fine-grained detail tokens $\mathcal{D}$. The final retained visual-token set is

$$\mathcal{S} = \mathcal{P} \dot{\cup} \mathcal{C} \dot{\cup} \mathcal{D}, \qquad |\mathcal{S}| = K. \tag{11}$$

The selected visual indices are sorted by their original sequence order before being passed to all subsequent LLM layers.

Algorithm 1 RoRA selection at pruning layer ℓ_p

Require: visual states $\mathcal{H}_V$, query/key states at ℓ_p, priors b_i, o_i, budget K

1: Extract raw visual attention a_i from the last text-token query to visual keys.
2: Load reusable offline priors b_i and o_i for the current layer and visual length.
3: Compute calibrated core score p_i using Eq. (19).
4: Select protected core $\mathcal{P}$ with target count $K_p^\star$ and the redundancy guard enabled when configured.
5: Build the AAR soft regional prior $\mathcal{R}_{\mathrm{AAR}}$ from top anchors on the 2D visual grid.
6: Compute context score c_i using the AAR inside/outside values and the optional tail signal.
7: Select $\mathcal{C}$ from $\mathcal{V} \setminus \mathcal{P}$ under cosine-similarity redundancy filtering.
8: Compute detail score d_i from normalized attention, feature magnitude, local contrast, and the AAR indicator.
9: Select $\mathcal{D}$ from $\mathcal{V} \setminus (\mathcal{P} \cup \mathcal{C})$ by tail replacement or residual fill.
10: Form $\mathcal{S} = \mathcal{P} \dot{\cup} \mathcal{C} \dot{\cup} \mathcal{D}$.
11: **return** $\mathcal{S}$ after sorting by original token order.

Budget accounting. For fixed-budget LLaVA-style runs, K is the retained visual-token count. For ratio-based Qwen-style runs, $K = \mathrm{round}(N(1-\rho))$, where ρ is the pruning ratio. The implementation first sets the target protected-core count

$$K_p^\star = \min(K, \max(1, \lfloor \rho_p K \rfloor)). \tag{12}$$

After the redundancy guard is applied, the accepted protected set defines $K_p = |\mathcal{P}|$. Given the configured target detail count k_d, RoRA then sets

$$K_d = \min(k_d, K - K_p), \qquad K_c = K - K_p - K_d. \tag{13}$$

Detail repair is implemented as replacement of lower-scoring non-protected tail tokens whenever the selected base set has already reached K. Thus the final counts always satisfy

$$\begin{aligned} K &= K_p + K_c + K_d, \\ \mathcal{P} \cap \mathcal{C} &= \mathcal{P} \cap \mathcal{D} = \mathcal{C} \cap \mathcal{D} = \emptyset. \end{aligned} \tag{14}$$

B Attention and Prior Implementation

Raw attention a_i. RoRA uses the query of the last text token at the pruning boundary because this token is the immediate language-side aggregation point for the current answer step. For attention head h, let q_h be the last text-token query and let $k_{j,h}$ be the key of sequence position j. With head dimension d_h, the implementation computes

$$\begin{aligned} z_{h,j} &= \frac{q_h k_{j,h}^\top}{\sqrt{d_h}}, \\ \alpha_{h,j} &= \frac{\exp(z_{h,j})}{\sum_m \exp(z_{h,m})}. \end{aligned} \tag{15}$$

For grouped-query attention backbones, key heads are repeated to match the query-head count before the dot product. RoRA then averages over heads and slices the visual-token span:

$$a_i = \frac{1}{H}\sum_{h=1}^{H} \alpha_{h,s+i}, \qquad i = 1, \ldots, N, \tag{16}$$

where s is the start offset of the visual span in the full sequence. In implementations where the selector is called immediately before layer ℓ_p, the attention tensors may be captured by a hook on the preceding self-attention computation; the pruning boundary remains ℓ_p.

Normalization and numerical stability. All scalar score vectors are flattened over visual tokens and cleaned before normalization:

$$\mathrm{Clean}(x_i) = \begin{cases} x_i, & x_i \text{ finite}, \\ 0, & \text{otherwise}, \end{cases}. \tag{17}$$

With $x_{\min} = \min_j \mathrm{Clean}(x_j)$ and $x_{\max} = \max_j \mathrm{Clean}(x_j)$,

$$\mathrm{Norm}(x)_i = \frac{\mathrm{Clean}(x_i) - x_{\min}}{x_{\max} - x_{\min} + \epsilon}. \tag{18}$$

The stabilizer is $\epsilon = 10^{-7}$ in the implementation. Negative raw attention values are not expected after softmax; detail scoring still clamps the attention branch at zero before normalization as a defensive numerical step.

Offline positional prior b_i. The positional prior is built once per backbone and pruning layer from an unlabeled 1,000-image calibration split sampled from GQA training images, using images only. Each calibration image is forwarded with the same generic description prompt. For each requested layer, RoRA stores the mean last-text-query attention over visual positions. The resulting vector is normalized and cached. If the current visual length differs from the calibration length, the prior vector is linearly resized to length N before use.

Offline weak object-oriented prior o_i. The weak prompt-induced object-oriented spatial prior is built with the same 1,000 calibration images and the same attention extraction path, replacing the generic prompt with an object-listing prompt. The implementation forms a robust object-to-generic ratio, clips extreme ratio values at a high quantile, z-normalizes the vector, and clamps it to a bounded range. This produces a reusable spatial tendency o_i that is independent of labels and reused across all evaluation samples for the same backbone/layer/visual length. No additional prompt is appended to an evaluation sample, and no extra model forward is introduced at inference time. RoRA is therefore training-free, detector-free, and does not require per-benchmark prior construction.

C Object-Calibrated Core Protection

RoRA combines instance attention, positional debiasing, and the weak object-oriented prior through

$$\begin{aligned} r_i &= \mathrm{Norm}\left(\frac{a_i}{b_i + \epsilon}\right), \\ p_i &= \mathrm{Norm}(r_i + \lambda_{\mathrm{obj}} o_i). \end{aligned} \tag{19}$$

In the LLaVA-family runtime, the same operation is implemented with a z-normalized attention branch before adding the z-normalized object-oriented prior; in the Qwen-family runtime, the debiased attention branch is min-max normalized before prior addition. Both forms preserve the same role of p_i: a calibrated, sample-conditioned score for semantic-core protection.

Without the redundancy guard, the protected semantic core is selected as

$$\mathcal{P} = \mathrm{TopK}_{K_p^\star}(p_i). \tag{20}$$

When graph-safe selection is enabled, candidates are visited in descending p_i order until the target $K_p^\star$ is reached or all finite unblocked candidates have been visited. A candidate is skipped if its

normalized feature has cosine similarity above the configured threshold to an already accepted token. The optional spatial weight augments this redundancy score with a small grid-distance term. The guard is only a redundancy filter: p_i remains the primary semantic-core score, and any unfilled slots are returned to the residual context/detail allocation so that the final budget is conserved.

Selection reference set and fallback. The implementation uses the currently retained visual tokens as the redundancy reference set. Let $\mathcal{R}$ denote the accepted set at the moment a candidate is evaluated. A candidate i is accepted only if

$$\max_{j \in \mathcal{R}} \cos(\bar{h}_i, \bar{h}_j) \leq \tau. \tag{21}$$

After accepting i, RoRA updates $\mathcal{R} \leftarrow \mathcal{R} \cup \{i\}$ and blocks its high-similarity neighbors. Thus context candidates are compared not only with $\mathcal{P}$ but also with context tokens already accepted earlier. If all remaining finite candidates are blocked before the requested count is reached, RoRA fills the residual slots by descending finite score while excluding already selected indices. This fallback is deterministic and preserves the requested budget.

D Attention-Anchored Regional Allocation

Region anchors. RoRA constructs AAR from high-confidence tokens rather than from external annotations. The anchor set is

$$\mathcal{A} = \mathrm{TopK}_A(p_i), \tag{22}$$

where A is the configured anchor count. These anchors are used only to form a soft regional prior for subsequent budget allocation.

2D grid mapping. For fixed-grid LLaVA-1.5, $N = 576$ visual tokens are mapped to a 24×24 grid by row-major order. For dynamic-resolution backbones, the runtime selector receives only the flattened visual span and infers a near-rectangular grid: if N is a square, it uses $\sqrt{N} \times \sqrt{N}$; otherwise it chooses the largest factor $H \leq \sqrt{N}$ and sets $W = \lceil N/H \rceil$. Padded cells beyond N are ignored. Special non-visual sequence positions are outside the visual span and are never candidates for pruning. Local neighborhoods do not wrap around row boundaries, and boundary tokens use only valid in-span neighbors.

Radius expansion and AAR construction. Given radius r_a, each anchor $u \in \mathcal{A}$ expands to a square neighborhood on the visual grid:

$$\mathcal{N}_{r_a}(u) = \{v : \ |x_v - x_u| \leq r_a, \ |y_v - y_u| \leq r_a\}. \tag{23}$$

The AAR soft regional prior is the union of these neighborhoods,

$$\mathcal{R}_{\mathrm{AAR}} = \bigcup_{u \in \mathcal{A}} \mathcal{N}_{r_a}(u). \tag{24}$$

RoRA uses this set as a soft regional prior only. It does not impose hard per-region quotas; it changes the scoring of residual candidates through inside/outside values.

Context score c_i. For residual tokens outside $\mathcal{P}$, RoRA defines

$$g_i = \begin{cases} v_{\mathrm{in}}, & i \in \mathcal{R}_{\mathrm{AAR}}, \\ v_{\mathrm{out}}, & i \notin \mathcal{R}_{\mathrm{AAR}}, \end{cases} . \tag{25}$$

The corresponding context score is

$$c_i = \mathrm{Norm}(p_i + \lambda_{\mathrm{aar}} g_i + \lambda_{\mathrm{tail}} t_i) \, . \tag{26}$$

Here v_{in} and v_{out} are the configured inside/outside values, λ_{aar} is the AAR weight, and t_i is an optional detail-tail score used in Qwen-family settings. Since $v_{\mathrm{out}} > v_{\mathrm{in}}$ in all fixed RoRA configurations, tokens outside already anchored neighborhoods receive a small complementary-evidence boost. Context tokens $\mathcal{C}$ are then accepted from high to low c_i, skipping tokens whose feature cosine similarity exceeds the configured threshold relative to the retained set. This keeps $\mathcal{C}$ focused on non-duplicative evidence while preserving the original budget.

E AAR-GUIDED DETAIL REPAIR

Detail repair uses three normalized token-level signals. The attention response is

$$\widehat{a}_i = \mathrm{Norm}(\max(a_i, 0)). \tag{27}$$

The feature magnitude is

$$\widehat{m}_i = \mathrm{Norm}(\|h_i\|_2). \tag{28}$$

The local contrast uses the eight-neighbor grid neighborhood $\mathcal{N}_8(i)$:

$$\ell_i = \frac{1}{|\mathcal{N}_8(i)|} \sum_{j \in \mathcal{N}_8(i)} \left(1 - \cos(\bar{h}_i, \bar{h}_j)\right). \tag{29}$$

It is then normalized as

$$\widehat{\ell}_i = \mathrm{Norm}(\ell_i). \tag{30}$$

Here $\bar{h}_i = h_i / \|h_i\|_2$. Boundary tokens use only valid neighbors.

The default detail profile used in the reported settings is

$$\begin{aligned} u_i &= \mathrm{Norm}(0.60\widehat{a}_i + 0.25\widehat{m}_i + 0.15\widehat{\ell}_i), \\ v_i &= \mathrm{Norm}(0.75\widehat{a}_i + 0.25\widehat{m}_i), \\ d_i &= \mathrm{Norm}\Big(0.70v_i + 0.30u_i + 0.10\widehat{\ell}_i + \lambda_{\mathrm{det}}\eta_i\Big), \end{aligned} \tag{31}$$

where $\eta_i = \mathbb{I}[i \in \mathcal{R}_{\mathrm{AAR}}]$. The AAR term is a soft preference that raises detail candidates near protected semantic evidence without changing the candidate universe.

RoRA selects $\mathcal{D}$ from tokens not already assigned to $\mathcal{P}$ or $\mathcal{C}$. In tail-replacement mode, the base selector first forms a K-token set. The protected tokens are marked non-removable, the lowest-scoring removable tail tokens are dropped, and the top K_d detail candidates by d_i are inserted. Qwen-family runs additionally restrict detail candidates to a configured candidate pool ranked by calibrated base score, and the graph-safe flag removes detail candidates already blocked by the redundancy guard. The final visual set is sorted by original sequence position before the next LLM layer receives it.

The detail stage intentionally uses the raw attention branch only as one bounded signal inside d_i. The calibrated score p_i already determines protected core and context allocation. In contrast, detail repair is a small-count recovery step; using $\widehat{a}_i$ helps recover visually responsive tokens that may have been down-weighted by positional debiasing, while feature magnitude, local contrast, the AAR indicator, the candidate pool, and graph-safe filtering keep this repair stage constrained.

F MATHEMATICAL AND ALGORITHMIC ANALYSIS

The analysis below describes deterministic properties of the implemented selector. It is intended to make the budget flow and score interactions explicit, without introducing new empirical claims.

Monotone score fusion. Every RoRA stage ranks candidates after applying $\mathrm{Norm}(\cdot)$ to a scalar vector over visual tokens. Min-max normalization is monotone: if $x_i > x_j$ and the denominator is nonzero, then $\mathrm{Norm}(x)_i > \mathrm{Norm}(x)_j$. When all entries are equal, the stabilizer makes the normalized vector finite and flat. Therefore normalization changes scale but preserves candidate order within each score branch. This is why the implementation can combine attention, prior, AAR, and detail terms with small weights while keeping the original attention evidence interpretable.

Budget conservation. RoRA maintains the requested visual budget by construction. The core stage selects from $\mathcal{V}$, the context stage selects from $\mathcal{V} \setminus \mathcal{P}$, and the detail stage selects from tokens not already retained or replaces removable tail tokens while keeping $\mathcal{P}$ fixed. In set form,

$$\begin{aligned} &\mathcal{P} \subseteq \mathcal{V}, \\ &\mathcal{C} \subseteq \mathcal{V} \setminus \mathcal{P}, \\ &\mathcal{D} \subseteq \mathcal{V} \setminus (\mathcal{P} \cup \mathcal{C}). \end{aligned} \tag{32}$$

The implementation also stores a Boolean selected indicator, so duplicate indices are removed before the final sequence is formed. If the redundancy guard blocks too many candidates, RoRA fills the residual slots from the highest finite tail scores. Thus, for $N \geq K$, the output contains exactly K visual tokens and satisfies

$$\mathcal{S} = \mathcal{P} \dot{\cup} \mathcal{C} \dot{\cup} \mathcal{D}. \tag{33}$$

Protected-core invariance. Tail replacement never removes protected semantic-core tokens. Let $\mathcal{S}_{\text{base}}$ be the K-token set before detail repair and let $\mathcal{R} = \mathcal{S}_{\text{base}} \setminus \mathcal{P}$ be the removable part. Detail repair drops tokens only from $\mathcal{R}$ and adds tokens only from $\mathcal{V} \setminus \mathcal{S}_{\text{base}}$. Consequently,

$$\mathcal{P} \subseteq \mathcal{S} \tag{34}$$

always holds. This property is useful in implementation because the semantic-core guarantee does not depend on the later detail count, gate, or candidate-pool size.

AAR as a residual-allocation bias. AAR changes residual allocation through the scalar value g_i, not through a hard constraint. For two residual tokens with identical p_i and identical optional tail score t_i, the pre-normalization score gap between an outside-AAR token and an inside-AAR token is

$$\Delta_{\text{out}-\text{in}} = \lambda_{\text{aar}}(v_{\text{out}} - v_{\text{in}}). \tag{35}$$

All fixed RoRA configurations use $v_{\text{out}} > v_{\text{in}}$. Therefore, when $\lambda_{\text{aar}} > 0$, AAR gives a positive residual-allocation preference to tokens outside already anchored neighborhoods. When $\lambda_{\text{aar}} = 0$, as in the LLaVA-1.5 $K = 128$ setting, the AAR still participates in detail gating but does not change context-tail ranking.

Cosine redundancy guard. The redundancy guard maintains a blocked set around every accepted token. With normalized features $\bar{h}_i$, threshold τ, and current candidate pool $\mathcal{U}$, accepting token j blocks

$$\mathcal{B}(j) = \{i \in \mathcal{U} : \cos(\bar{h}_i, \bar{h}_j) > \tau\}. \tag{36}$$

Subsequent context or graph-safe detail candidates are skipped if they fall in the accumulated blocked set. This creates a simple ordering rule: high p_i or c_i candidates are still preferred, but near-duplicates of already retained evidence do not consume additional budget unless the residual fill step is needed to reach K.

Complexity and memory. Let H be the number of attention heads, d the hidden size, A the number of anchors, r_a the AAR radius, and N_t the residual candidate count after core protection. The selector cost can be written as

$$\begin{aligned} T_{\text{sel}} = {} & O(HN) + O(N) + O(A(2r_a + 1)^2) \\ & + O(8Nd) + O(N_t^2). \end{aligned} \tag{37}$$

The first two terms correspond to attention aggregation and scalar score fusion. The AAR term is small because A and r_a are fixed constants in all reported configurations. The $O(8Nd)$ term comes from eight-neighbor local contrast. The final $O(N_t^2)$ term appears when the cosine guard is materialized as an adjacency table; an implementation can also stream candidate-to-retained comparisons to trade memory for repeated dot products. The memory footprint is $O(N)$ for scalar score vectors and AAR indicators, plus either $O(N_t^2)$ for a materialized adjacency table or $O(N_t d)$ for streamed normalized features.

G Protocol Checks and Minimal Evidence

This section adds only protocol-level checks and existing completed diagnostics. It avoids introducing unsupported claims or numbers that were not present in saved launcher/result metadata.

Normalized average. For a method m and benchmark set $\mathcal{B}$, RoRA reports an equal-weight normalized average against the full-token reference score F_b:

$$\text{NAvg}(m) = \frac{100}{|\mathcal{B}|} \sum_{b \in \mathcal{B}} \frac{R_{m,b}}{F_b}. \tag{38}$$

The benchmark set $\mathcal{B}$ must be identical for all methods compared in the same normalized-average column. If a baseline has missing entries for any benchmark in $\mathcal{B}$, the appendix should either recompute all methods on the common subset or avoid reporting a directly comparable average for that baseline. We use "full-token reference" rather than "upper bound," since pruning can occasionally exceed the full-token score on individual benchmarks.

Budget protocol. The retained visual budget is matched at the sample level whenever the runtime exposes the visual length before pruning. For fixed-grid LLaVA-1.5 runs, $N = 576$ and K is fixed. For dynamic-resolution runs, the per-sample budget is

$$K_i = \max(1, \mathrm{round}((1-\rho)N_i)), \tag{39}$$

where N_i is the visual-token length of sample i and ρ is the target pruning ratio. Non-visual tokens, separator positions, padding outside the visual span, and text positions are not counted in N_i or K_i.

Table 5: Budget protocol used by RoRA. The table reports rules rather than distribution statistics because per-sample retention summaries were not present in the saved metadata inspected for this appendix.

Backbone	Budget type	Sample-level rule	Counted tokens	Non-visual tokens
LLaVA-1.5	Fixed K	$K \in \{192, 128, 64\}$ with $N = 576$	Projected visual tokens	Retained unchanged
LLaVA-NeXT	Fixed K in reported setting	$K = 320$ after dynamic visual encoding	Flattened visual span	Retained unchanged
Qwen2.5-VL	Ratio-based	$K_i = \mathrm{round}((1-\rho)N_i)$ for 75% and 90% pruning	Flattened image-token span	Retained unchanged
Qwen3-VL	Ratio-based	$K_i = \mathrm{round}((1-\rho)N_i)$ for 75% and 90% pruning	Runtime visual indices	Retained unchanged

Existing RefCOCO-family stage diagnostic. Saved RefCOCO-family diagnostics contain complete eight-split results for FastV, RoRA core, and RoRA core with AAR at two budgets. Table 6 reports raw localization averages over RefCOCO Val/TestA/TestB, RefCOCO+ Val/TestA/TestB, RefCOCOg Val, and RefCOCOg Test. This table summarizes completed result metadata to clarify the contribution of calibrated core protection and the AAR soft regional prior under matched budgets.

Table 6: Existing RefCOCO-family stage diagnostic. Values are raw averages over eight localization splits.

K	Pruning	FastV	RoRA Core	RoRA Core+AAR	Incremental change
58	90%	4.476	8.000	8.050	Core $+3.524$, AAR $+0.050$
144	75%	16.825	35.899	36.183	Core $+19.073$, AAR $+0.284$

AAR coverage diagnostic. For any saved run with token-level traces, AAR coverage can be computed without rerunning inference:

$$\mathrm{Cover}(\mathcal{R}_{\mathrm{AAR}}) = \frac{|\mathcal{R}_{\mathrm{AAR}}|}{N}. \tag{40}$$

This diagnostic measures the area of the soft regional prior. Coverage statistics are tracked as a reproducibility item and are not included as a numerical claim in this version.

Efficiency protocol. We distinguish the standard accuracy configuration, denoted RoRA-Full, from any speed-aligned configuration, denoted RoRA-Core. RoRA-Full uses the complete core,

AAR/context, redundancy filtering, and detail repair specified in the implementation tables. RoRA-Core refers to a reduced configuration used only when a speed table explicitly states it; its disabled or reduced components must be listed next to that table. Reliable timing should use warm-up runs followed by at least five repeated measurements and should report selector overhead, end-to-end latency, total evaluation time, and decoding length. Repeated timing statistics are tracked as a reproducibility item and are not added as numerical claims in this version.

H TARGETED DIAGNOSTIC MECHANISM ABLATION

Implementation and provenance notes. This targeted addendum contrasts D2-style debiased global selection with RoRA's role-oriented allocation. RoRA uses the weak object-aware prior as a soft cue for protected semantic-core allocation, AAR as a soft regional prior, and tail replacement as a detail-repair step. The prior is not a detector mask, and the diagnostic does not tune settings separately for each benchmark.

Table 7: Compact mechanism ablation on LLaVA-1.5-7B. Rows are compared only within the same retained-token budget.

Method	K	Deb.	Obj.	AAR	Detail	POPE	TextVQA	VizWiz	Overall
FastV	128	N	N	N	N	81.89	58.06	51.37	96.71
RoRA-Core	128	Y	Y	N	N	85.06	58.24	51.63	98.07
RoRA-Core+AAR	128	Y	Y	Y	N	84.93	58.14	51.61	98.28
RoRA-Full	128	Y	Y	Y	tail	84.90	58.18	51.71	98.33
D2Pruner-style Base	192	Y	N	N	N	85.24	58.62	33.63	99.47
RoRA-Core	192	Y	Y	N	N	85.42	58.64	33.71	99.56
RoRA-Core+AAR	192	Y	Y	Y	N	85.39	58.70	33.72	99.62
RoRA-Full	192	Y	Y	Y	tail	85.46	58.66	33.79	99.66

Deb. denotes position-debiased scoring; Obj. denotes the weak object-aware prior. FastV and the D2Pruner-style baseline use global filtering, while RoRA rows use core-referenced filtering.

Protocol. The deadline addendum uses RTX PRO 6000 for targeted short reruns with LLaVA-1.5-7B, $N = 576$, retained budgets $K = 128$ and $K = 192$, pruning ratios 77.8% and 66.7%, and pruning layer 2. All added numbers are read from CSV/JSON/log artifacts.

Limitations of additional evidence. Because these experiments were added under a limited compute budget, Table 7 is a diagnostic mechanism check rather than a replacement for the full benchmark comparisons in the main paper. The main paper remains the source for broad benchmark claims and the reported 24.6% reduction in end-to-end inference time, corresponding to a $1.33\times$ speedup over unpruned inference.

I Full Implementation Settings

Tables 8–10 list the anonymized RoRA settings used for the reported backbone–budget pairs. The values were read from final launcher records, result-alignment configs, and saved metadata. To avoid a dense many-column layout, each row groups the same implementation parameters into budget/core, AAR/context, redundancy, and detail-repair fields. The table intentionally uses anonymous record labels only.

Table 8: Full LLaVA-family RoRA implementation settings. N is the visual-token length before pruning, A is the anchor count, and r_a is the AAR radius.

Setting	Budget and core	AAR and context	Redundancy guard	Detail repair
LLaVA-1.5, $K = 192$; record L15-K192	$N = 576$; $\ell_p = 2$; $\rho_p = .650$; $K_p^\star = 124$.	$A = 24$; $r_a = 2$; $\lambda_{\text{aar}} = .050$; $v_{\text{in}}/v_{\text{out}} = .25/1.0$; $\lambda_{\text{obj}} = .0025$.	Graph-safe on; $\tau = .8$; spatial weight $w_s = 0$.	$K_d = 4$; profile v1; gate .20; candidate pool all; tail weight n/a; tail-replacement mode.
LLaVA-1.5, $K = 128$; record L15-K128	$N = 576$; $\ell_p = 2$; $\rho_p = .690$; $K_p^\star = 88$.	$A = 20$; $r_a = 2$; $\lambda_{\text{aar}} = .000$; $v_{\text{in}}/v_{\text{out}} = .25/1.0$; $\lambda_{\text{obj}} = .0025$.	Graph-safe on; $\tau = .8$; spatial weight $w_s = 0$.	$K_d = 6$; profile v1; gate .005; candidate pool all; tail weight n/a; tail-replacement mode.
LLaVA-1.5, $K = 64$; record L15-K64	$N = 576$; $\ell_p = 2$; $\rho_p = .550$; $K_p^\star = 35$.	$A = 16$; $r_a = 1$; $\lambda_{\text{aar}} = .005$; $v_{\text{in}}/v_{\text{out}} = .25/1.0$; $\lambda_{\text{obj}} = .0005$.	Graph-safe on; $\tau = .8$; spatial weight $w_s = 0$.	$K_d = 1$; profile v1; gate .10; candidate pool all; tail weight n/a; tail-replacement mode.
LLaVA-NeXT, $K = 320$; record Next-K320	N dynamic; $\ell_p = 2$; $\rho_p = .700$; $K_p^\star = 224$.	$A = 32$; $r_a = 2$; $\lambda_{\text{aar}} = .050$; $v_{\text{in}}/v_{\text{out}} = .25/1.0$; $\lambda_{\text{obj}} = .0025$.	Graph-safe on; $\tau = .8$; spatial weight $w_s = 0$.	$K_d = 4$; profile v1; gate .20; candidate pool all; tail weight n/a; tail-replacement mode.

Table 9: Full Qwen2.5-VL RoRA implementation settings. Each pruning ratio uses one fixed RoRA configuration across the corresponding benchmark suite, with $K = \text{round}((1 - \rho)N)$.

Setting	Budget and core	AAR and context	Redundancy guard	Detail repair
Qwen2.5-VL, 75% pruning; record Q25-R75	$\ell_p = 2$; $K = .25N$; $\rho_p = .970$.	$A = 16$; $r_a = 3$; $\lambda_{\text{aar}} = .000350$; $v_{\text{in}}/v_{\text{out}} = .25/1.0$; $\lambda_{\text{obj}} = .000600$.	Graph-safe on; $\tau = .800$; spatial weight $w_s = .000060$.	$K_d = 7$; profile v1; gate 4.60; candidate pool 40; tail weight .000060; tail-replacement mode.
Qwen2.5-VL, 90% pruning; record Q25-R90	$\ell_p = 2$; $K = .10N$; $\rho_p = .795$.	$A = 13$; $r_a = 1$; $\lambda_{\text{aar}} = .000250$; $v_{\text{in}}/v_{\text{out}} = .25/1.0$; $\lambda_{\text{obj}} = .000350$.	Graph-safe on; $\tau = .796$; spatial weight $w_s = .000025$.	$K_d = 5$; profile v1; gate 4.00; candidate pool 32; tail weight .000020; tail-replacement mode.

Table 10: Full Qwen3-VL RoRA implementation settings from anonymized saved metadata. The metadata records multiple frozen RoRA labels by task family, so the appendix reports label-level provenance rather than collapsing mixed metadata into one artificial configuration.

Setting	Budget and core	AAR and context	Redundancy guard	Detail repair
Q3-Mix, 75%/90% pruning; multi-task label	$\ell_p = 2$; $K = (1-\rho)N$; $\rho_p = .970$.	$A = 14$; $r_a = 2$; $\lambda_{\text{aar}} = .000150$; $v_{\text{in}}/v_{\text{out}} = .25/1.0$; $\lambda_{\text{obj}} = .000600$.	Graph-safe on; $\tau = .800$; spatial weight $w_s = .000040$.	$K_d = 4$; profile v1; gate 3.60; candidate pool 32; tail weight .000040; tail-replacement mode.
Q3-Strong, 75% pruning; 75% subset label	$\ell_p = 2$; $K = .25N$; $\rho_p = .970$.	$A = 16$; $r_a = 3$; $\lambda_{\text{aar}} = .000350$; $v_{\text{in}}/v_{\text{out}} = .25/1.0$; $\lambda_{\text{obj}} = .000600$.	Graph-safe on; $\tau = .800$; spatial weight $w_s = .000060$.	$K_d = 7$; profile v1; gate 4.60; candidate pool 40; tail weight .000060; tail-replacement mode.
Q3-Safe, 90% pruning; 90% subset label	$\ell_p = 2$; $K = .10N$; $\rho_p = .950$.	$A = 12$; $r_a = 2$; $\lambda_{\text{aar}} = .000080$; $v_{\text{in}}/v_{\text{out}} = .25/1.0$; $\lambda_{\text{obj}} = .000500$.	Graph-safe on; $\tau = .850$; spatial weight $w_s = .000025$.	$K_d = 2$; profile v1; gate 3.00; candidate pool 20; tail weight .000020; tail-replacement mode.
Q3-P95, 75% pruning; 75% subset label	$\ell_p = 2$; $K = .25N$; $\rho_p = .950$.	$A = 14$; $r_a = 2$; $\lambda_{\text{aar}} = .000150$; $v_{\text{in}}/v_{\text{out}} = .25/1.0$; $\lambda_{\text{obj}} = .000600$.	Graph-safe on; $\tau = .800$; spatial weight $w_s = .000040$.	$K_d = 4$; profile v1; gate 3.60; candidate pool 32; tail weight .000040; tail-replacement mode.

For all rows, non-visual tokens are retained unchanged, visual indices are selected within the visual span, and the final index list is sorted before subsequent LLM layers. No benchmark-specific score discussion is included in this supplement.

J Normalization References and Timing Boundaries

J.1 Reference scores for the normalized averages

The archived evaluation records contain two full-token reference vectors: a literature reference and a locally evaluated reference. The Vanilla rows displayed in Table 1 contain the literature reference scores. The local evaluation records instead use the local full-token scores listed in Table 11. In the LLaVA-1.5 records, the aggregate column is explicitly designated as an average relative to the local full-token reference; the recorded evaluation scope includes MMB static-dev, relaxed TextVQA evaluation, and VizWiz test-self. The LLaVA-NeXT records likewise retain separate literature and local full-token reference rows.

Table 11: Full-token reference scores retained in the archived evaluation records. The literature-reference columns match the Vanilla rows displayed in Table 1; the local-reference columns reproduce its displayed RoRA normalized averages as detailed in Table 12.

	LLaVA-1.5-7B		LLaVA-NeXT-7B	
Benchmark	Literature	Local	Literature	Local
GQA	61.9	61.98	64.2	64.18
MMB	64.7	64.09	67.4	68.30
MMB-CN	58.1	55.58	60.6	46.91
MME	1862	1869.70	1851	1849.30
POPE	85.9	86.97	86.5	88.76
ScienceQA	69.5	69.46	70.1	69.86
VQAv2	78.4	76.71	81.8	81.06
TextVQA	58.2	59.70	64.9	65.25
VizWiz	50.0	50.54	57.6	57.17

To make the denominator distinction explicit, let $R^{\mathrm{disp}}_{m,b}$ be a benchmark score as displayed in Table 1. For reference type $q \in \{\mathrm{local}, \mathrm{lit}\}$, the corresponding reconstruction is

$$\mathrm{NAvg}^{\mathrm{disp}}_{q}(m) = \frac{100}{9} \sum_{b \in \mathcal{B}} \frac{R^{\mathrm{disp}}_{m,b}}{F^{q}_{b}}, \tag{41}$$

where $\mathcal{B}$ contains the nine benchmarks in Table 11. Table 12 applies this calculation to all four RoRA rows. Using the local reference gives 99.8175%, 99.1305%, 96.5166%, and 95.4560%, respectively; rounding to one decimal place reproduces the reported 99.8%, 99.1%, 96.5%, and 95.5%. In particular, the NeXT average is 93.4527% when the displayed literature reference is used instead. The discrepancy therefore cannot be attributed solely to rounding the displayed literature reference: the choice of denominator is material.

Table 12: Arithmetic reconstruction of the RoRA averages in Table 1 from its displayed benchmark scores. Both reconstructed columns use identical numerators and differ only in the full-token reference. These are consistency calculations, not additional experimental runs.

Model	K	Reported	Local reference	Literature reference
LLaVA-1.5-7B	192	99.8%	99.8175%	99.5773%
LLaVA-1.5-7B	128	99.1%	99.1305%	98.9055%
LLaVA-1.5-7B	64	96.5%	96.5166%	96.3015%
LLaVA-NeXT-7B	320	95.5%	95.4560%	93.4527%

These reconstructions explain the displayed aggregate values but do not establish the exact unrounded scores of the final experimental runs. The single Vanilla row in Table 1 must therefore not be treated as a universal denominator for every aggregate entry. Comparisons of normalized averages require a common reference vector and matched evaluation protocols; aggregates inherited from different reference conventions are not automatically comparable. This clarification records the reference provenance without replacing the benchmark scores in the main table.

J.2 Interpretation of the Prefill and Total columns

The RoRA latency entries in Table 4 are based on a latency profiler over the first 100 POPE samples, with batch size 1 and `max_new_tokens=1`. The first five samples are excluded as warm-up, leaving 95 samples in each reported arithmetic mean. These are 95 sample timings, rather than 95 independent repetitions of the complete experiment.

For each sample, the profiler first times a standalone model forward call with caching enabled to obtain the Prefill measurement. It then separately times a call to `model.generate` with one new token to obtain the Total measurement. The key/value cache returned by the standalone forward is not passed into the generation call. Each call is timed using a wall-clock timer with CUDA synchronization immediately before and after execution. Input loading and transfer to the GPU occur outside these timed regions, as do output text decoding and file writing. Thus, Total denotes the generation-call latency under this protocol, not full data-processing pipeline latency.

The two columns are consequently measurements of separate calls, not nested stages recorded during a single execution. Their difference must not be interpreted as decoding time, and the inequality $T_{\mathrm{Total}} \geq T_{\mathrm{Prefill}}$ is not a required relation between these separately measured means. The saved RoRA summaries contain Prefill/Total means of 108.0887/106.7600 ms and 66.4893/65.6589 ms for the two retained-token budgets, which round to the entries in Table 4. These records verify the origin of the displayed numbers; they do not identify the specific cause of the small ordering difference between the two call paths. Each speedup is computed against the corresponding unpruned timing in the same column.

The profiler summaries also retain medians and 90th percentiles, but no standard deviations or independent-run uncertainty estimates are available in those summaries. The FLOP and KV-cache entries in Table 4 are budget-matched estimates retained from the D^2Pruner comparison setup, rather than additional measurements produced by this latency profiler. This protocol is distinct from the full 9,000-question POPE evaluation used for the LLaVA-1.5 timing comparison in Table 3.

K ISOLATING THE OBJECT PRIOR IN CORE SELECTION

We conduct a separate diagnostic to isolate the contribution of the object prior beyond positional debiasing. We use LLaVA-1.5-7B with pruning at layer 2, allocate the entire retained-token budget to core top-K selection, and disable context, detail repair, and graph-based redundancy filtering. Within each paired comparison, the model, samples, retained budget, positional debiasing, score normalization, position indices, and decoding settings are identical; only the object-prior coefficient changes from zero to its selected value. Decoding is greedy with at most 32 new tokens.

Selection and evaluation protocol. For each benchmark, we search $K \in \{64, 128, 192\}$ and $\lambda_{\text{obj}} \in \{0, .01, .03, .1, .3, .6\}$ on 60 examples. Among nonzero-coefficient candidates, we first restrict selection to configurations with a positive search-set gain when available, then rank by object-prior accuracy, gain, smaller coefficient, and larger budget. If none improves over its matched baseline, gain takes precedence over accuracy. The selected configuration is frozen before evaluation on 160 further examples. Search and evaluation samples are disjoint by image-file content hash, and images used in earlier diagnostic rounds are excluded. Budgets and coefficients are selected separately for each benchmark; this diagnostic therefore uses task-specific tuning rather than the main benchmark configuration.

Table 13: Per-benchmark core-only object-prior diagnostic on LLaVA-1.5-7B. Each column reports scores (%) on 160 held-out question examples, image-disjoint from the 60 search examples used to select the budget and coefficient. The two score rows use the same budget and differ only in the object-prior coefficient; the coefficient row gives the nonzero value used by Core + obj. These are diagnostic-subset results with task-specific tuning, not full benchmark results.

Setting	GQA	MMBench EN	VQAv2	ScienceQA	VizWiz	TextVQA
K	192	128	192	128	64	64
Core, no obj	64.38	74.38	75.50	71.88	25.75	39.31
Core + obj	66.88	75.62	75.69	71.88	25.56	41.00

GQA uses normalized exact match, MMBench EN and the image subset of ScienceQA use option-letter accuracy, and VQAv2, VizWiz, and TextVQA use ten-annotator consensus scoring with EvalAI answer normalization. Scores in Table 13 are rounded to two decimal places. The object prior improves four diagnostic subsets, leaves ScienceQA unchanged, and slightly reduces VizWiz performance. The largest gains are 2.50 percentage points on GQA and 1.69 points on TextVQA, computed before rounding.

Scope of the evidence. These small, internally held-out subsets probe the object prior under the selected settings. They do not establish a stable improvement on full benchmarks or under a shared coefficient. Image-clustered paired 95% bootstrap intervals from 3,000 resamples include or touch zero for all six differences and are not adjusted for multiple comparisons. We therefore treat the observed gains as diagnostic evidence rather than a claim of statistically significant improvement.