ERASE: ELIMINATING REDUNDANT VISUAL TOKENS VIA ADAPTIVE TWO-STAGE TOKEN PRUNING

Anonymous authors
Paper under double-blind review

ABSTRACT

Recent advances in Vision-Language Models (VLMs) enable large language models (LLMs) to process high-resolution images, significantly improving real-world multimodal understanding. However, this capability introduces a large number of vision tokens, incurring substantial computational overhead. To mitigate this issue, various vision token compression methods have been proposed. Existing methods often estimate different sources of visual redundancy using learned representations or fixed pruning schedules. We propose ERASE, an adaptive two-stage framework that separates image-level redundancy removal from instruction-dependent token pruning. Stage 1 derives image-dependent token retention from lightweight raw-image statistics, while Stage 2 progressively removes instruction-irrelevant tokens across decoder layers. Experiments demonstrate substantial token reduction while preserving accuracy: on Qwen2.5-VL-7B, ERASE retains 95.70% of the original model's accuracy at 25% token retention.

1 INTRODUCTION

Recent VLMs augment LLMs with visual understanding capabilities by jointly processing text and image modalities (Bai et al., 2025b; Chen et al., 2024c). However, visual inputs typically generate a large number of vision tokens, significantly increasing computational cost and inference latency. Vision token compression mitigates this bottleneck by removing redundant or less relevant tokens (Wen et al., 2025; Zhang et al., 2025; Fang et al., 2026; Chen et al., 2024a; Baek et al., 2026; Zhang et al., 2026; Huang et al., 2026; Hu et al., 2026). Meanwhile, existing approaches assess visual redundancy using learned semantic features or identify instruction-relevant tokens within the LLM, often overlooking the potential of image-adaptive retention guided by raw-image statistics and layer-wise changes in text–vision alignment.

In this paper, we propose *ERASE: Eliminating Redundant visual tokens via Adaptive two-StagE token pruning*, a hierarchical vision token pruning framework that compresses vision tokens through two complementary pruning stages. ERASE decomposes vision-token redundancy into two complementary sources and removes them at different points in the VLM pipeline. Stage 1 adaptively removes image redundancy before the LLM using each image's own statistics, while Stage 2 removes instruction-irrelevant tokens within the decoder through hierarchical pruning. This separation avoids relying on expensive encoder attention maps for early redundancy removal while preserving instruction-dependent information later in the model. Extensive experiments demonstrate that ERASE consistently achieves a superior efficiency-performance trade-off over existing vision token compression methods across diverse VLM architectures and benchmarks.

2 MOTIVATION

2.1 VISION TOKEN SCALING IN MODERN VLMS

VLMs process images by partitioning them into patches and converting into visual tokens for LLM processing. Early VLMs used pretrained vision encoders, such as CLIP (Radford et al., 2021). These encoders typically operated at fixed input resolutions. For instance, the original LLaVA (Liu et al., 2023) encodes 224×224 images into 256 visual tokens using CLIP. Resizing high-resolution images to a fixed resolution can discard fine-grained details and degrade visual understanding. To address

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.

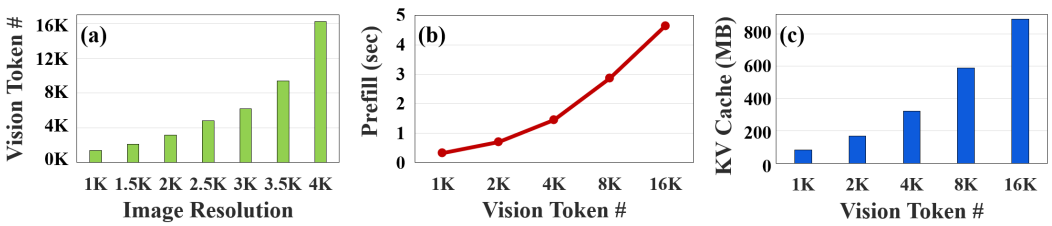
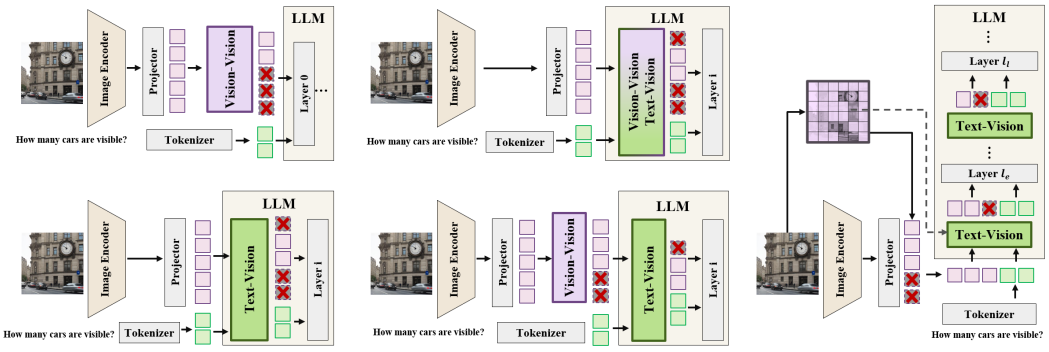


Figure 1: Scaling behavior of (a) vision token count with increasing image resolution, and (b) prefill latency and (c) KV cache size with increasing vision token count in Qwen2.5-VL-7B.



(a) Single-objective compression (b) Hybrid compression (c) Proposed ERASE

Figure 2: Comparison of vision token compression schemes. (a) (Top) vision-only compression (Fang et al., 2026; Baek et al., 2026); (Bottom) text-conditioned compression (Chen et al., 2024a; Xing et al., 2025). (b) (Top) single-stage hybrid compression (Zhang et al., 2025; Wen et al., 2025); (Bottom) multi-stage hybrid compression (Zhang et al., 2026; Huang et al., 2026; Hu et al., 2026). (c) image-adaptive two-stage hybrid compression.

this limitation, advanced VLMs adopt dynamic tiling strategies that partition high-resolution images into multiple tiles (Liu et al., 2024a; Wu et al., 2024; Zhu et al., 2025). For instance, InternVL3 (Zhu et al., 2025) selects a tile grid according to the image's aspect ratio and adds a global thumbnail to preserve global context. Recent models such as Qwen2.5-VL (Bai et al., 2025b) and Qwen3-VL (Bai et al., 2025a) further support fully configurable resolutions using dedicated image encoders.

However, recent VLMs that support high-resolution image processing substantially increase the number of vision tokens. Since images are represented as 2D spatial grids, the number of tokens scales quadratically with image resolution (Fig. 1a). For example, a 2K image generates approximately 3K tokens, while a 4K image produces around 16K tokens. Attention complexity scales quadratically with sequence length, and the token explosion significantly increases prefill latency (Fig. 1b) and KV cache memory usage (Fig. 1c). Therefore, efficient VLM inference requires effective vision token compression.

2.2 Prior Vision Token Compression Methods

As shown in Fig. 2, vision token compression methods can be categorized into single-objective and hybrid approaches. Single-objective approaches focus on either visual redundancy or text-vision relevance, whereas hybrid approaches combine both in single-stage or multi-stage frameworks.

2.2.1 Single-Objective Compression Methods

Vision-only compression Vision-only methods remove redundant tokens before the LLM using attention scores or feature similarity. Attention-based methods (Baek et al., 2026; Yang et al., 2025) primarily target CLIP-based VLMs such as LLaVA, which inherit a pretrained CLIP encoder with a `[CLS]` token trained as a global image representation aligned with text in a shared embedding space. These methods can exploit the `[CLS]`-to-patch attention already available in the pretrained CLIP encoder as an efficient pruning cue. However, unlike CLIP-based encoders, many modern VLMs use dedicated vision encoders without a `[CLS]`-like global summary token.

A direct adaptation of such attention-based methods then requires access to the full $N \times N$ self-attention map. Moreover, image encoders often process more tokens than are passed to the LLM after vision-language projector downsampling. This causes prohibitive memory overhead for high-resolution inputs, exceeding 120 GB for Qwen2.5-VL-7B (Fig. 3) and making such methods impractical. Feature-similarity-based methods instead group visually similar tokens and preserve those with lower similarity scores (Fang et al., 2026). However, visually important regions often depend on the text prompt, and ignoring text-vision relevance leaves room for further compression. Their effectiveness also depends heavily on the ability of the target model to encode semantic information into visual features.

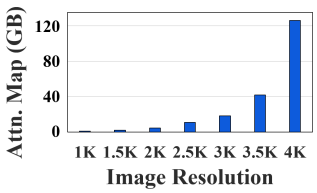


Figure 3: Attention map scaling in the image encoder.

Text-conditioned compression Text-conditioned approaches retain vision tokens that are highly relevant to the text instruction. These methods typically operate inside the LLM backbone, leveraging cross-attention to compress tokens receiving low attention weights from text tokens (Chen et al., 2024a; Xing et al., 2025). However, they overlook redundancy among similar vision tokens. As noted by DART (Wen et al., 2025), once a representative token is retained from a group of similar tokens, the importance of the remaining redundant tokens should be adjusted.

2.2.2 Hybrid compression strategies

Single-stage hybrid compression Single-stage approaches jointly address visual redundancy and text relevance, typically at or just before the LLM decoder. For instance, DART (Wen et al., 2025) employs similarity-based metrics using pivot tokens from both modalities, while CDPruner (Zhang et al., 2025) selects tokens before the LLM decoder by maximizing diversity conditioned on instruction relevance. However, these methods make compression decisions at a single point, without accounting for dynamic layer-wise evolution of vision-text relationships within the LLM backbone.

Multi-stage hybrid compression Multi-stage approaches generally combine both objectives by first reducing visual redundancy before the LLM decoder layers and then applying context-aware compression within it (Zhang et al., 2026; Huang et al., 2026; Hu et al., 2026). Separating these objectives can mitigate attention dilution among similar visual tokens before relevance filtering. Despite this advantage, the methods adopted at each stage still present limitations. They often rely on attention in the image encoder to remove redundancy, introducing the same memory overhead discussed for vision-only methods. VScan (Zhang et al., 2026) further exacerbates this issue as it requires two attention maps inside the image encoder, effectively doubling the severe memory overhead shown in Fig. 3. iLLaVA (Hu et al., 2026) mitigates this overhead by accessing the softmax normalizer values that FlashAttention-2 (Dao, 2024) computes, instead of materializing the full attention map. However, their redundancy estimates depend on learned visual representations, while stage-wise budgets remain fixed across images. This motivates model-independent early pruning with image-adaptive retention followed by context-aware pruning.

3 Proposed method

3.1 Overview of ERASE

ERASE derives image-dependent token retention from raw-image statistics to reduce visual redundancy, followed by instruction-aware pruning (Fig. 4). Stage 1 jointly determines which visual tokens to retain and how many using image-relative redundancy cues. Stage 2 progressively removes instruction-irrelevant tokens at shallow and deep layers, reusing Stage 1 visual cues to complement shallow attention while relying on instruction relevance alone for deep pruning.

3.2 Stage 1: Image-level vision token pruning

Stage 1 exploits low-level image structure to identify visually redundant tokens, motivated by the well-established observation that structural information is concentrated around intensity transitions and high-frequency regions, while smooth regions are more spatially redundant.

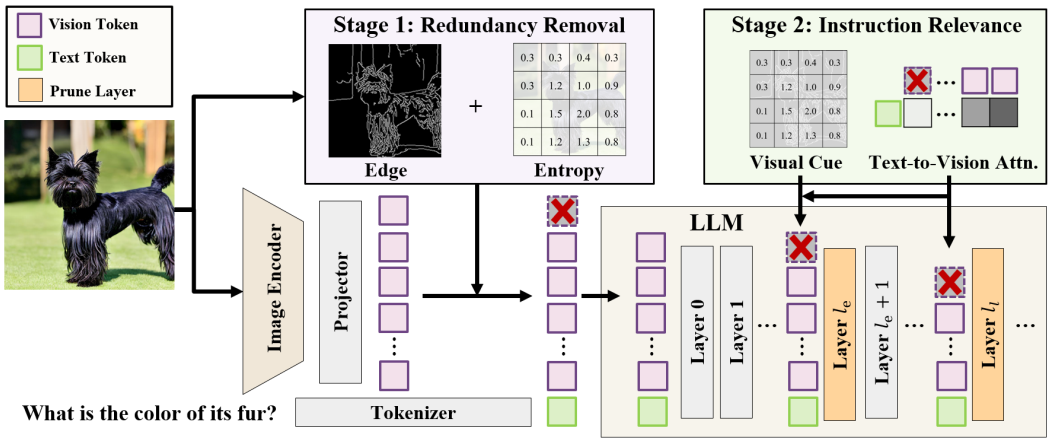


Figure 4: Overview of ERASE.

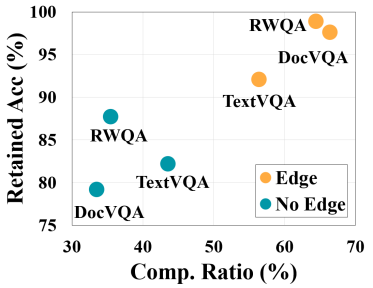


Figure 5: Accuracy–compression of edge and non-edge patches.

We verify this for vision-token compression by separately retaining patches with and without edges. As shown in Fig. 5, edge-containing patches better preserve accuracy under compression, supporting edges as informative cues. Based on this observation, ERASE uses edges as the primary cue for informative regions. To reduce sensitivity to color variation and local noise, the input is converted to luminance and smoothed by average pooling and Gaussian filtering, yielding $\tilde{Y}$, which is used only to derive pruning cues while the original image is passed unchanged to the image encoder. Edges from $\tilde{Y}$ capture structural boundaries but may miss informative regions with subtle variations. ERASE therefore complements them with patch-wise entropy to capture local information diversity. Together, the two cues identify informative patches and prune smooth, low-information regions.

Structural information estimation via edges To detect edges, we compute the horizontal and vertical gradients of $\tilde{Y}$ using Sobel filters (Sobel & Feldman, 1968),

$$G_x = S_x * \tilde{Y}, \qquad G_y = S_y * \tilde{Y}, \tag{1}$$

$$S_x = \begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix}, \qquad S_y = \begin{bmatrix} -1 & -2 & -1 \\ 0 & 0 & 0 \\ 1 & 2 & 1 \end{bmatrix}, \tag{2}$$

from which the gradient magnitude at pixel (x, y) is computed as

$$M(x, y) = \sqrt{G_x(x, y)^2 + G_y(x, y)^2}. \tag{3}$$

Following the Sobel gradient computation, we adopt Canny-inspired thresholding (Canny, 1986). Canny uses both high and low thresholds to preserve edge continuity. In contrast, we use a single adaptive threshold derived from the image median, since our goal is to identify informative regions rather than precisely trace edges. With $\tau \in [0, 1]$ controlling the edge sensitivity, we define

$$\tau_E = (1 - \tau)\,\mathrm{median}(\tilde{Y}), \qquad e(x, y) = \mathbf{1}[M(x, y) > \tau_E]\,. \tag{4}$$

A token is retained if its corresponding patch contains any detected edge pixel. Otherwise, the patch is treated as a candidate redundant region. Because gradient magnitude can miss informative low-contrast structures, faint handwriting on a bright board is almost entirely pruned in Fig. 6b (top). We address this limitation using the complementary entropy-based criterion described next.

Local information estimation via entropy To complement the edge-based criterion, we measure patch-wise intensity entropy from the preprocessed luminance map $\tilde{Y}$. While edge information captures spatial intensity changes, entropy captures the diversity of intensity values within each



(a) Original patches (b) Edge patches (c) High-entropy patches (d) Combined patches

Figure 6: Retained regions selected by the edge, entropy, and combined criteria.

patch, providing a complementary cue for informative regions without strong boundaries. For each patch p of size $p_h \times p_w$, we construct a 256-bin histogram of its 8-bit luminance values and compute

$$H_p = -\sum_{v=0}^{255} \hat{P}_p(v) \ln \hat{P}_p(v), \qquad \hat{P}_p(v) = \frac{n_p(v)}{p_h \cdot p_w}, \tag{5}$$

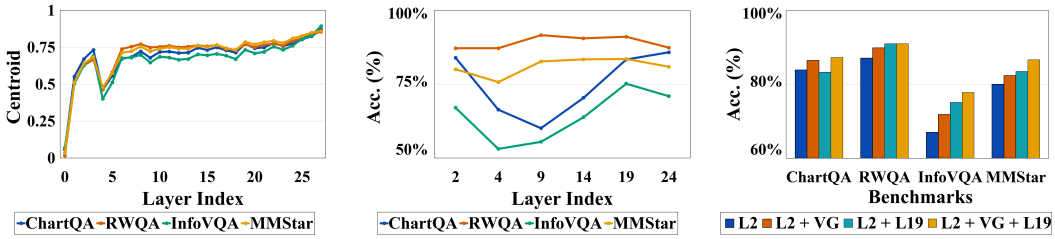
where $n_p(v)$ is the number of pixels with luminance value v. Uniform patches concentrate their values in a few bins and thus have low entropy, whereas patches with diverse intensity values produce broader distributions and higher entropy. Since entropy distributions vary across images, ERASE uses the mean entropy across all patches, $\bar{H}$, as an image-adaptive threshold and retains a patch when $H_p > \bar{H}$. Unlike gradient magnitude, H_p depends on the distribution of intensity values rather than their spatial differences, allowing it to remain high even when local variations are too subtle to form strong edges. As shown in Fig. 6c (top), this enables the entropy criterion to retain faint handwriting that is discarded by the edge criterion. However, entropy measures intensity diversity without accounting for spatial structure, so informative but locally uniform patches may fall below $\bar{H}$, such as the simple arrow preserved by edges in Fig. 6 (bottom). To preserve regions identified by either cue, ERASE prunes a token only when its patch contains no detected edge and $H_p \leq \bar{H}$ (Fig. 6d). This complementary rule preserves structural boundaries and local intensity diversity while adapting retention to each image's statistics rather than a prescribed token count.

3.3 Stage 2: Context-aware vision token pruning

Layer-wise vision–text dynamics Prior studies suggest hierarchical multimodal processing across decoder layers (Wu et al., 2025; Shi et al., 2025). DyVTE (Wu et al., 2025) identifies early fusion, intra-modality modeling, and multimodal reasoning, with cross-modal interaction decreasing after early fusion and increasing again in deeper layers. Likewise, Shi et al. (2025) associate shallow layers with basic visual recognition and deeper layers with fine-grained visual-semantic tasks. To characterize the layer-wise interaction between the two modalities, we use a centroid-based proxy for vision–text alignment, defined as the cosine similarity between their mean representations at each decoder-layer input. Let $X_t^{(\ell)} \in \mathbb{R}^{n_t \times d}$ and $X_v^{(\ell)} \in \mathbb{R}^{n_v \times d}$ denote the text and vision representations entering layer ℓ, respectively. Their centroids and vision–text centroid similarity are defined as

$$\bar{x}_t^{(\ell)} = \frac{1}{n_t} \sum_{i=1}^{n_t} X_{t,i}^{(\ell)}, \qquad \bar{x}_v^{(\ell)} = \frac{1}{n_v} \sum_{j=1}^{n_v} X_{v,j}^{(\ell)}, \qquad S^{(\ell)} = \frac{\bar{x}_t^{(\ell)} \cdot \bar{x}_v^{(\ell)}}{\|\bar{x}_t^{(\ell)}\|_2 \|\bar{x}_v^{(\ell)}\|_2}. \tag{6}$$

Throughout this section, we use samples correctly answered by the base model and define retained accuracy as the fraction remaining correct after compression. Fig. 7a shows an early rise, subsequent drop, and eventual recovery in this proxy, broadly consistent with the hierarchical processing described above. This trend also appears across model sizes and architectures (Appendix A.4). This pattern motivates examining an early pruning point before the drop and a later point after recovery.



(a) Vision–text centroid similarity (b) Deep-layer selection (c) Stage 2 ablation

Figure 7: Layer-wise analysis on Qwen2.5-VL-7B: (a) vision–text centroid similarity, (b) retained accuracy across deep pruning layers under a 25% layer-averaged token budget, and (c) Stage 2 component analysis, where L2 denotes shallow instruction-aware pruning, VG adds Stage 1 visual guidance, and L19 denotes additional deep pruning at layer 19. Stage 1 pruning is disabled in (b)/(c).

Hierarchical context-aware pruning Guided by this observation, we evaluate shallow and deep pruning points. We fix the shallow pruning point at layer 2, which lies before the observed similarity drop, and sweep the deep pruning point under a 25% layer-averaged budget to evaluate candidate deep pruning layers (Fig. 7b). Accuracy decreases when the second pruning point lies in the post-drop region and generally recovers as it moves deeper. However, pruning too late can reduce accuracy because meeting the same budget requires more aggressive late-stage pruning. Given these pruning points, ERASE ranks vision tokens by instruction relevance at each stage. For compatibility with FlashAttention-2, we separately compute the instruction-to-vision attention. The relevance of vision token v is

$$a_v = \frac{1}{N_h} \sum_{h=1}^{N_h} \sum_{t \in \mathcal{T}_{\text{inst}}} A_{t,v}^{(h)}, \tag{7}$$

where N_h is the number of attention heads and $\mathcal{T}_{\text{inst}}$ denotes the instruction tokens. At the shallow pruning stage, ERASE combines instruction relevance with the visual cues from Stage 1. Since this is the first stage where instruction relevance is introduced, retaining visual informativeness helps avoid overly aggressive instruction-driven pruning at an early layer. Specifically, shallow pruning reuses the Stage 1 edge density $\rho_v = \frac{1}{p_h p_w} \sum_{(x,y) \in p(v)} e(x,y)$ and patch entropy $H_{p(v)}$, where $p(v)$ denotes the corresponding Stage 1 patch. The visual cues are combined with attention as

$$g_v = \frac{\rho_v}{\text{std}(\rho)} + \frac{H_{p(v)}}{\text{std}(H)}, \qquad z_v = a_v + w \, \frac{\text{std}(a)}{\text{std}(g)} g_v. \tag{8}$$

To account for per-image scale differences, we normalize each visual cue by its standard deviation, which also makes the result independent of the units of ρ and H, and rescale their combined guidance to match the spread of attention scores, allowing w to control its relative strength. Tokens are ranked by z_v at the shallow layer and a_v at the deep layer. After shallow pruning has jointly considered visual informativeness and instruction relevance, deep pruning focuses on instruction relevance alone. Fig. 7c analyzes the components of Stage 2. Adding visual guidance to shallow pruning improves average accuracy by 3.14 points across four datasets. Hierarchical shallow–deep pruning further improves accuracy under the same layer-averaged budget. Combining visually guided shallow pruning with deep instruction-aware pruning achieves the highest overall accuracy, supporting the proposed hierarchical design.

4 Experiments

4.1 Experiment Settings

Models and benchmarks ERASE is evaluated on Qwen2.5-VL-7B (Bai et al., 2025b) and Qwen3-VL-8B (Bai et al., 2025a) with native-resolution image processing, and InternVL3-8B (Zhu et al., 2025) with dynamic image tiling. We evaluate using VLMEvalKit (Duan et al., 2024): text recognition (TextVQA (Singh et al., 2019)), document/chart understanding (ChartQA (Masry et al., 2022), DocVQA (Mathew et al., 2021), InfoVQA (Mathew et al., 2022)), visual reasoning (MathVista (Lu et al., 2024)), and general visual question answering (RealWorldQA (xAI Corp., 2024),

MMStar (Chen et al., 2024b), MMBench v1.1 (Liu et al., 2024b)). Additional results on smaller models are provided in Appendix A.3.

Implementation details ERASE prunes at layer 2 and at approximately two-thirds of the decoder depth, using the same relative-depth rule across architectures: layer 19 of 28 for Qwen2.5-VL-7B and InternVL3-8B, and layer 24 of 36 for Qwen3-VL-8B. We set $\tau = 0.45$ and $w = 0.2$ for all models. We compare against PDrop (Xing et al., 2025), FastV (Chen et al., 2024a), DART (Wen et al., 2025), CDPruner (Zhang et al., 2025), PruneSID (Fang et al., 2026), iLLaVA (Hu et al., 2026), AgilePruner (Baek et al., 2026), VScan (Zhang et al., 2026), and Nüwa (Huang et al., 2026) on InternVL3-8B. AgilePruner, VScan, and Nüwa are evaluated only on InternVL3-8B because they incur OOM on the Qwen models due to encoder attention maps. Retain denotes the layer-averaged fraction of visual tokens processed during prefill relative to the base model, while Rel. denotes the average accuracy relative to the base model. PDrop follows its reported configuration because its pruning schedule cannot match our retention ratios. Appendix A.1 provides further details.

4.2 Accuracy evaluation

ERASE achieves the highest average accuracy at all three matched retention ratios on Qwen2.5-VL-7B and Qwen3-VL-8B (Tables 1 and 2). At 25% retention, ERASE outperforms the strongest baseline on each model, iLLaVA and PruneSID, by 3.95 and 4.68 points, respectively, while preserving 95.70% and 86.71% of the base model accuracy. The gains are particularly pronounced on

Table 1: Results on Qwen2.5-VL-7B.

Retain↓	Method	TextVQA	ChartQA	DocVQA	InfoVQA	MathVista	$MMB_{v1.1}$	RWQA	MMStar	Avg.↑	Rel. (%)↑
100%	Base	85.30	86.00	94.82	82.31	65.10	82.51	70.20	65.53	78.97	100
47%	PDrop	84.19	82.12	92.55	71.34	56.40	77.48	64.97	58.87	73.49	93.06
45%	FastV	84.40	82.92	93.19	74.90	63.20	81.58	68.50	62.00	76.34	96.66
	DART	80.38	77.04	85.35	60.57	62.00	80.96	69.15	62.53	72.25	91.49
	CDPruner	80.96	66.04	85.41	61.91	56.70	80.88	67.84	63.07	70.35	89.08
	PruneSID	80.46	80.56	90.32	64.95	61.30	81.11	69.15	62.73	73.82	93.48
	iLLaVA	84.16	82.28	93.09	76.43	64.30	81.81	68.76	62.27	<u>76.64</u>	<u>97.04</u>
	ERASE	84.52	83.24	94.22	79.27	64.20	81.27	70.20	64.13	**77.63**	**98.30**
35%	FastV	83.75	79.04	90.27	68.13	62.30	80.80	67.97	60.20	74.06	93.78
	DART	76.91	70.40	78.17	53.10	57.70	79.88	67.19	61.33	68.09	86.21
	CDPruner	79.19	59.24	81.57	57.56	55.40	80.42	67.19	62.00	67.82	85.88
	PruneSID	77.04	77.60	86.07	57.49	60.40	80.50	67.84	62.20	71.14	90.09
	iLLaVA	83.24	79.08	90.91	72.11	60.70	81.81	69.02	62.40	<u>74.91</u>	<u>94.86</u>
	ERASE	84.26	83.04	94.12	76.89	63.70	81.66	69.80	63.93	**77.18**	**97.73**
25%	FastV	82.01	71.24	81.09	55.84	58.20	78.72	66.41	56.73	68.78	87.09
	DART	69.56	59.16	66.25	41.83	54.00	78.56	64.44	57.07	61.36	77.70
	CDPruner	75.92	50.52	75.65	51.09	53.30	79.80	65.49	60.53	64.04	81.09
	PruneSID	70.65	70.84	77.59	50.13	59.20	79.49	65.62	60.80	66.79	84.58
	iLLaVA	80.64	70.76	86.26	64.74	58.80	81.66	68.50	61.60	<u>71.62</u>	<u>90.69</u>
	ERASE	83.50	80.20	93.35	71.83	62.20	81.50	69.41	62.60	**75.57**	**95.70**

Table 2: Results on Qwen3-VL-8B.

Retain↓	Method	TextVQA	ChartQA	DocVQA	InfoVQA	MathVista	$MMB_{v1.1}$	RWQA	MMStar	Avg.↑	Rel. (%)↑
100%	Base	82.98	83.16	95.75	83.21	69.10	84.52	71.37	66.60	79.59	100
47%	PDrop	75.05	61.64	84.94	59.46	48.80	77.94	68.76	54.93	66.44	83.48
45%	FastV	75.18	53.96	81.92	55.22	49.00	76.63	68.24	55.07	64.40	80.92
	DART	71.53	69.32	82.69	59.65	56.10	79.41	67.84	59.53	68.26	85.77
	CDPruner	74.24	56.00	72.63	52.28	42.60	77.55	63.40	52.93	61.45	77.22
	PruneSID	78.18	76.92	93.46	73.16	59.90	78.87	69.15	60.40	<u>73.76</u>	<u>92.67</u>
	iLLaVA	77.40	65.64	93.02	76.57	45.40	81.35	70.85	60.53	71.35	89.64
	ERASE	80.99	77.76	94.77	77.74	54.70	81.19	70.33	61.53	**74.88**	**94.08**
35%	FastV	69.77	42.00	70.07	45.61	43.60	74.61	65.75	49.00	57.55	72.31
	DART	65.18	60.36	73.30	50.36	49.80	78.25	68.76	55.07	62.64	78.70
	CDPruner	69.94	46.72	63.57	47.70	38.00	73.14	61.44	49.33	56.23	70.65
	PruneSID	73.54	65.64	90.46	66.90	52.80	75.62	68.24	60.33	<u>69.19</u>	<u>86.94</u>
	iLLaVA	74.18	62.28	90.74	70.59	44.30	80.19	70.33	57.93	68.82	86.47
	ERASE	79.99	75.04	94.29	72.82	51.00	79.49	70.85	59.07	**72.82**	**91.50**
25%	FastV	61.48	27.76	51.42	35.66	37.80	68.96	63.40	45.80	49.04	61.61
	DART	57.26	50.20	57.46	40.48	47.60	74.46	64.71	51.87	55.51	69.74
	CDPruner	62.26	37.92	52.07	41.39	38.40	68.96	58.43	46.60	50.75	63.77
	PruneSID	69.01	58.12	83.66	57.26	50.40	72.76	68.37	55.07	<u>64.33</u>	<u>80.83</u>
	iLLaVA	70.61	51.04	85.81	58.54	42.50	78.02	67.58	55.13	63.65	79.98
	ERASE	77.75	68.08	92.50	65.13	47.10	76.63	69.28	55.60	**69.01**	**86.71**

Table 3: Results on InternVL3-8B.

Retain↓	Method	TextVQA	ChartQA	DocVQA	InfoVQA	MathVista	$MMB_{v1.1}$	RWQA	MMStar	Avg.↑	Rel. (%)↑
100%	Base	82.11	86.08	91.98	75.49	68.80	85.76	71.24	68.73	78.77	100
47%	PDrop	81.46	82.56	88.69	67.52	58.50	83.90	70.59	65.27	74.81	94.97
45%	FastV	81.05	82.52	82.17	64.37	64.00	84.98	71.24	66.00	74.54	94.63
	DART	74.19	78.72	77.59	57.33	66.30	83.90	70.33	66.00	71.80	91.14
	CDPruner	70.78	64.80	66.87	53.99	57.80	82.04	66.27	60.80	65.42	83.05
	PruneSID	72.96	77.76	76.02	58.93	66.10	82.82	70.59	64.93	71.26	90.47
	iLLaVA	66.63	59.64	68.65	54.53	58.40	81.81	68.37	62.60	65.08	82.61
	AgilePruner	79.33	79.60	83.83	66.82	63.40	83.90	69.15	64.73	73.85	93.74
	VScan	80.89	81.04	88.73	71.44	60.90	83.98	71.24	65.40	75.45	95.78
	Nüwa	80.54	82.96	87.13	70.34	65.90	84.06	71.37	66.27	<u>76.07</u>	<u>96.57</u>
	ERASE	80.47	82.64	87.95	70.24	66.40	84.75	71.37	67.13	**76.37**	**96.95**
35%	FastV	79.88	79.84	73.80	54.69	64.30	84.83	70.33	65.47	71.64	90.95
	DART	70.55	73.28	68.95	48.99	65.10	83.36	68.63	63.60	67.81	86.08
	CDPruner	65.28	55.96	55.42	46.92	52.10	80.57	63.27	57.60	59.64	75.71
	PruneSID	67.42	72.28	62.56	49.84	65.10	81.97	68.89	64.33	66.55	84.48
	iLLaVA	63.61	56.08	62.77	48.68	55.90	79.72	66.53	61.13	61.80	78.46
	AgilePruner	76.71	76.72	77.61	61.33	60.10	83.44	67.19	62.13	70.65	89.69
	VScan	79.71	79.28	85.06	68.28	57.30	83.13	70.07	64.07	73.36	93.13
	Nüwa	79.33	79.84	82.57	66.77	64.00	83.51	70.46	65.20	<u>73.96</u>	<u>93.89</u>
	ERASE	79.92	80.72	86.53	65.30	65.80	84.21	71.37	66.73	**75.07**	**95.30**
25%	FastV	76.68	74.92	63.86	45.21	61.50	83.82	69.67	62.93	67.32	85.46
	DART	64.08	62.08	54.94	40.18	62.40	80.88	66.41	60.33	61.41	77.96
	CDPruner	58.21	46.76	42.60	39.46	42.50	78.25	60.39	55.93	53.01	67.30
	PruneSID	57.10	61.88	46.75	40.19	58.20	78.87	67.84	61.87	59.09	75.01
	iLLaVA	58.59	50.92	56.64	44.21	53.40	79.64	66.27	60.00	58.71	74.53
	AgilePruner	71.13	67.28	66.77	52.76	57.60	83.44	64.58	59.40	65.37	82.98
	VScan	76.35	72.60	75.02	59.64	53.60	81.58	66.54	61.73	68.38	86.81
	Nüwa	75.94	74.92	73.19	59.76	57.90	82.66	68.76	62.93	<u>69.51</u>	<u>88.24</u>
	ERASE	78.23	78.12	80.66	57.22	63.30	84.37	70.98	66.20	**72.39**	**91.89**

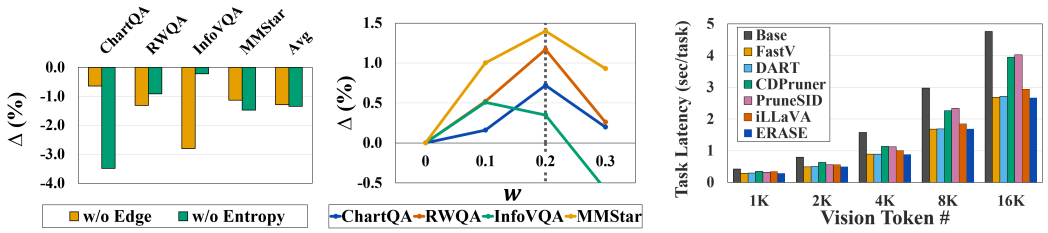
Table 4: Vision-only token compression comparison. † denotes Stage 1 only.

Method	Budget	TextVQA	ChartQA	DocVQA	InfoVQA	MathVista	$MMB_{v1.1}$	RWQA	MMStar	Avg.↑	Rel. (%)↑
Base	–	85.30	86.00	94.82	82.31	65.10	82.51	70.20	65.53	78.97	100
PruneSID	Fixed	84.38	82.48	89.28	73.50	62.90	81.66	70.20	64.07	76.06	96.31
iLLaVA†	Fixed	84.66	83.44	91.06	79.62	61.70	82.28	68.50	63.87	76.89	97.37
ERASE†	Fixed	84.81	83.56	93.58	79.31	64.40	82.59	70.07	63.27	<u>77.70</u>	<u>98.39</u>
ERASE†	Adaptive	84.89	83.76	94.32	80.31	65.00	81.73	71.11	64.60	**78.22**	**99.04**
Retain (%)		69.11	52.51	41.28	60.77	47.22	63.84	64.73	57.24	57.09	–

OCR-intensive tasks, where removing even a small text region can alter the recognized content. Table 3 evaluates ERASE on InternVL3-8B, which uses dynamic tiling and thus tests its applicability to a different visual input architecture. Its encoder `[CLS]` token also enables comparison with AgilePruner, VScan, and Nüwa without OOM. VScan and Nüwa are the strongest baselines at matched retention ratios, yet ERASE achieves the highest average accuracy at all three ratios. At 25% retention, ERASE preserves 91.89% of the base-model accuracy and exceeds AgilePruner, VScan, and Nüwa by 7.02, 4.01, and 2.88 points in average accuracy, respectively. Appendix A.3.2 additionally evaluates AgilePruner, VScan, and Nüwa on a Qwen model using downscaled images.

4.3 Ablation and efficiency analysis

Effectiveness of Stage 1 pruning We compare ERASE's Stage 1 with vision-only compression methods, PruneSID and Stage 1 of iLLaVA at the benchmark-wise average retention ratios obtained by ERASE's adaptive Stage 1 (Table 4). Fixed ERASE matches these ratios by removing or adding patches to its Stage 1 selection using the visual score in Eq. 8. It achieves an average accuracy of 77.70%, outperforming iLLaVA (76.89%) and PruneSID (76.06%). Adapting retention to individual images using image-relative statistics further improves average accuracy to 78.22%. These results support the effectiveness of ERASE's visual selection criteria and the additional benefit of image-adaptive retention. To assess the contribution of each cue at Stage 1, we compare the combined edge-entropy criterion with edge-only and entropy-only variants in Stage 1 (Fig. 8a). Either cue alone yields lower accuracy than both cues together across all benchmarks, with their relative contributions varying by benchmark. These results support retaining regions identified by either cue.



(a) Stage 1 visual cues (b) Stage 2 guidance weight (c) Task latency

Figure 8: Ablation and efficiency analysis on Qwen2.5-VL-7B. (a) Stage 1 accuracy loss using either visual cue alone. (b) Stage 2 shallow-layer visual-guidance accuracy gains over $w = 0$. Accuracies at $w = 0$ are 79.48, 68.24, 71.48, and 61.20 for ChartQA, RWQA, InfoVQA, and MMStar, respectively. (c) End-to-end latency across different visual token counts.

Table 5: Efficiency comparison of pruning methods on 4K images with 16K vision tokens.

Method	**KV Cache** (MB) ↓	**Prefill Latency** (ms) ↓	**Decode Latency** (ms/token) ↓	**Task Latency** (ms/task) ↓
Base	890.74	4647	37.34	4762
FastV	224.59 (×3.97)	2582 (×1.80)	33.47 (×1.12)	2689 (×1.77)
DART	224.75 (×3.96)	2606 (×1.78)	35.24 (×1.06)	2715 (×1.75)
CDPruner	224.61 (×3.97)	3848 (×1.21)	33.64 (×1.11)	3954 (×1.20)
PruneSID	224.62 (×3.97)	3926 (×1.18)	31.78 (×1.17)	4029 (×1.18)
iLLaVA	224.61 (×3.97)	2839 (×1.64)	33.54 (×1.11)	2941 (×1.62)
ERASE (Ours)	224.60 (×3.97)	2569 (×1.81)	32.51 (×1.15)	2671 (×1.78)

Impact of visual guidance at Stage 2 Fig. 8b varies w at the shallow pruning layer under 25% retention with Stage 1 enabled. Moderate visual guidance generally improves accuracy, with the best average performance at $w = 0.2$, whereas stronger guidance degrades performance across most benchmarks. This suggests that visual cues complement instruction relevance at the shallow layer by preserving useful visual context, while excessive visual guidance can dominate token selection and weaken instruction-aware pruning. We therefore set $w = 0.2$ to balance visual guidance and instruction relevance.

Inference efficiency We profile Qwen2.5-VL-7B on a single NVIDIA RTX 5880 Ada Generation GPU with batch size 1 and FlashAttention-2 (Dao, 2024), using a randomly sampled 10% subset of InfoVQA covering image resolutions corresponding to 1K–16K visual tokens. The target retention ratio is set to 25%. As the number of visual tokens increases, ERASE achieves increasingly larger task-latency reductions over the base model (Fig. 8c). Similar retention ratios do not necessarily result in similar latency because pruning overhead varies across methods: CDPruner performs iterative subset selection in embedding space, while PruneSID relies on pairwise token similarity and non-maximum suppression, causing their overhead to grow with the visual token count. In contrast, ERASE and FastV use lightweight token-selection procedures and scale more efficiently. At 16K visual tokens (Table 5), ERASE reduces the KV cache by 3.97× and achieves 1.81× prefill and 1.78× end-to-end speedups with low decoding overhead. Its task latency remains comparable to the most efficient baselines while achieving substantially higher accuracy (Table 1). Appendix A.3.5 provides additional results averaged over different resolutions.

5 Conclusion

We proposed ERASE, an adaptive two-stage framework for efficient VLM inference. Stage 1 jointly determines visual token selection and image-dependent retention from lightweight raw-image statistics, while Stage 2 progressively removes instruction-irrelevant tokens through hierarchical instruction-aware pruning. Across diverse VLM architectures and benchmarks, ERASE achieves better efficiency–accuracy trade-offs than prior methods, preserving higher accuracy under aggressive compression while substantially reducing end-to-end latency for high-resolution inputs.

AI USE STATEMENT

In this work, we did not use generative AI tools for literature retrieval or discovery, drafting paper sections, developing theoretical models or conceptual frameworks, proposing or refining hypotheses, designing or providing feedback on research methodology or experiments, implementing methods, translation, or interpreting results. Generating synthetic datasets, formulating or proving mathematical claims, cleaning or reformatting datasets, and qualitative or thematic data analysis are not applicable to this work. Additionally, we used generative AI tools to polish author-written text for grammar, clarity, and readability. We reviewed all AI-assisted edits to ensure that they preserved the intended meaning and accurately reflected our methods and results. We take responsibility for the final content of this work, including text, claims, or artifacts produced with the aid of generative AI.

REPRODUCIBILITY STATEMENT

For reproducibility, we provide detailed descriptions of the experimental protocol and implementation settings used in this work. Section 4.1 and Appendix A.1 document the model and evaluation configurations, pruning-layer selection, retention settings, and hyperparameters required to reproduce our experiments. Algorithm 1 describes the complete end-to-end procedure of ERASE, covering both Stage 1 image-level pruning and Stage 2 hierarchical instruction-aware pruning. We additionally provide our source code in the supplementary material.

REFERENCES

Changwoo Baek, Jouwon Song, Sohyeon Kim, and Kyeongbo Kong. AgilePruner: An empirical study of attention and diversity for adaptive visual token pruning in large vision-language models. In *International Conference on Learning Representations*, 2026.

Jinze Bai, Shuai Bai, Yunfei Chu, Zeyu Cui, Kai Dang, Xiaodong Deng, Yang Fan, Wenbin Ge, Yu Han, Fei Huang, et al. Qwen technical report. *arXiv preprint arXiv:2309.16609*, 2023.

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, et al. Qwen2.5-VL technical report. *arXiv preprint arXiv:2502.13923*, 2025b.

John Canny. A computational approach to edge detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, PAMI-8(6):679–698, 1986.

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*, 2024a.

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

Tri Dao. FlashAttention-2: Faster attention with better parallelism and work partitioning. In *International Conference on Learning Representations*, 2024.

Haodong Duan, Junming Yang, Yuxuan Qiao, Xinyu Fang, Lin Chen, Yuan Liu, Xiaoyi Dong, Yuhang Zang, Pan Zhang, Jiaqi Wang, et al. VLMEvalKit: An open-source toolkit for evaluating large multi-modality models. In *Proceedings of the 32nd ACM International Conference on Multimedia*, 2024.

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. In *International Conference on Learning Representations*, 2026.

Lianyu Hu, Liqing Gao, Fanhua Shang, Liang Wan, and Wei Feng. iLLaVA: An image is worth fewer than 1/3 input tokens in large multimodal models. In *International Conference on Learning Representations*, 2026.

Yihong Huang, Fei Ma, Yihua Shao, Jingcai Guo, Zitong Yu, Laizhong Cui, and Qi Tian. Nüwa: Mending the spatial integrity torn by VLM token pruning. In *International Conference on Learning Representations*, 2026.

Haotian Liu, Chunyuan Li, Qingyang Wu, and Yong Jae Lee. Visual instruction tuning. In *Advances in Neural Information Processing Systems*, 2023.

Haotian Liu, Chunyuan Li, Yuheng Li, Bo Li, Yuanhan Zhang, Sheng Shen, and Yong Jae Lee. LLaVA-NeXT: Improved reasoning, OCR, and world knowledge, 2024a. 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*, 2024b.

Pan Lu, Hritik Bansal, Tony Xia, Jiacheng Liu, Chunyuan Li, Hannaneh Hajishirzi, Hao Cheng, Kai-Wei Chang, Michel Galley, and Jianfeng Gao. MathVista: Evaluating mathematical reasoning of foundation models in visual contexts. In *International Conference on Learning Representations*, 2024.

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*, 2022.

Minesh Mathew, Dimosthenis Karatzas, and CV Jawahar. DocVQA: A dataset for VQA on document images. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, 2021.

Minesh Mathew, Viraj Bagal, Rubèn Tito, Dimosthenis Karatzas, Ernest Valveny, and CV Jawahar. InfographicVQA. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, 2022.

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*, 2021.

Cheng Shi, Yizhou Yu, and Sibei Yang. Vision function layer in multimodal LLMs. In *Advances in Neural Information Processing Systems*, 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*, 2019.

Irwin Sobel and Gary Feldman. A 3×3 isotropic gradient operator for image processing. Presented at a talk at the Stanford Artificial Intelligence Project (SAIL), 1968.

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*, 2025.

Qiong Wu, Wenhao Lin, Yiyi Zhou, Weihao Ye, Zhanpeng Zeng, Xiaoshuai Sun, and Rongrong Ji. Accelerating multimodal large language models via dynamic visual-token exit and the empirical findings. In *Advances in Neural Information Processing Systems*, 2025.

Zhiyu Wu, Xiaokang Chen, Zizheng Pan, Xingchao Liu, Wen Liu, Damai Dai, Huazuo Gao, Yiyang Ma, Chengyue Wu, Bingxuan Wang, et al. DeepSeek-VL2: Mixture-of-experts vision-language models for advanced multimodal understanding. *arXiv preprint arXiv:2412.10302*, 2024.

xAI Corp. Grok-1.5 vision preview: Connecting the digital and physical worlds with our first multimodal model, 2024. URL https://x.ai/blog/grok-1.5v.

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*, 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*, 2025.

Ce Zhang, Kaixin Ma, Tianqing Fang, Wenhao Yu, Hongming Zhang, Zhisong Zhang, Haitao Mi, and Dong Yu. VScan: Rethinking visual token reduction for efficient large vision-language models. *Transactions on Machine Learning Research*, 2026.

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*, 2025.

Jinguo Zhu, Weiyun Wang, Zhe Chen, Zhaoyang Liu, Shenglong Ye, Lixin Gu, Hao Tian, Yuchen Duan, Weijie Su, Jie Shao, et al. InternVL3: Exploring advanced training and test-time recipes for open-source multimodal models. *arXiv preprint arXiv:2504.10479*, 2025.

A APPENDIX

A.1 DETAILED EXPERIMENTAL SETTINGS

Evaluation and baseline settings We configure all baselines following their official implementations. CDPruner uses text tokens projected through the LLM embedding layer. DART and FastV prune at layer 2, with keep ratios adjusted to match the target layer-averaged retention. For iLLaVA, we use the official code setting for Qwen3-VL and the paper configuration for other model series. For Qwen3-VL, 40%, 60%, and 80% of the pruning budget are allocated to Stage 1 at target retention ratios of 45%, 35%, and 25%, respectively, with the remainder assigned to the LLM to achieve the target ratios. For VScan and Nüwa, Stage 2 is placed at the middle decoder layer. VScan uses $R_2 = 33.3\%$ with R_1 adjusted to the target, while Nüwa uses Stage 1 retention ratios of 55%, 45%, and 35% for targets of 45%, 35%, and 25%, respectively. PDrop cannot match the targets as its four-stage schedule begins pruning only after the first quarter of the decoder. We therefore use its original four-stage schedule, retaining 100%, 50%, 25%, and 12.5% of the vision tokens in successive stages. PruneSID and AgilePruner follow their official implementations, with pruning parameters adjusted to the target retention. For benchmarks where VLMEvalKit recommends LLM-as-a-judge evaluation, we use Qwen1.5-1.8B-Chat (Bai et al., 2023).

Configuration of ERASE For Stage 1, ERASE converts each input image to the luminance channel and applies 4×4 average pooling followed by a 3×3 Gaussian filter before computing the entropy and edge statistics. For Qwen-series models, which preserve the native image resolution, these statistics are computed directly from the input image. InternVL3 instead represents an image using a thumbnail together with multiple local tiles. Since the thumbnail and local tiles have different information densities, we compute their adaptive thresholds separately: the thumbnail uses its own mean entropy and median intensity, whereas all local tiles share the statistics computed over the tile set.

Because the Stage 1 retention ratio varies across images, ERASE dynamically selects one of three Stage 2 schedules to respect the target layer-averaged token budget. If Stage 1 already reaches or falls below the target retention, Stage 2 is skipped. In this case, the achieved retention can be lower than the target, so the reported retention ratios serve as upper bounds for ERASE. If using the default deep-layer keep ratio would require an early keep ratio greater than one, shallow pruning is skipped and the deep-layer keep ratio is adjusted instead. Otherwise, ERASE performs progressive pruning: the deep layer retains 30% of the vision tokens reaching it, while the shallow-layer keep ratio is adjusted per sample to meet the target layer-averaged retention.

A.2 ALGORITHM OF ERASE

Algorithm 1 summarizes the overall ERASE procedure. Stage 1 first converts the image to luminance and applies average pooling followed by Gaussian smoothing. It then identifies structurally informative regions using Sobel-based edges and locally informative regions using patch-wise entropy. A vision token is pruned only when its corresponding patch contains neither a detected edge nor above-average entropy. After Stage 1, ERASE constructs a per-sample Stage 2 pruning schedule from the Stage 1 retention r_1 and target layer-averaged retention $\bar{r}$. Depending on r_1, Stage 2 is either skipped, applied only at the deep pruning point, or applied progressively at both shallow and deep pruning points. Shallow pruning combines instruction-to-vision attention with the visual cues obtained in Stage 1, whereas deep pruning relies on attention alone.

For instance, consider a 28-layer model with target layer-averaged retention $\bar{r} = 25\%$ and layers 2 and 19 as the shallow and deep pruning points, respectively. If Stage 1 retains 30% of the original vision tokens, the progressive solution would require $r_e > 1$. ERASE therefore skips shallow pruning and uses only the deep pruning point, with $r_l \approx 48\%$. Consequently, layers 0–18 operate with 30% of the original vision tokens, while layers 19–27 operate with approximately $30\% \times 48\% \approx 14.4\%$. If Stage 1 retains 50%, ERASE performs progressive pruning with $r_e \approx 61\%$ at layer 2 and the default $r_l^0 = 30\%$ at layer 19. Thus, layers 0–1 operate with 50% of the original vision tokens, layers 2–18 with approximately $50\% \times 61\% \approx 30.5\%$, and layers 19–27 with approximately $50\% \times 61\% \times 30\% \approx 9.2\%$. In both cases, the vision-token retention averaged across all 28 decoder layers is approximately 25%.

Algorithm 1 ERASE: Adaptive Two-Stage Vision Token Pruning

Require: Image I, vision tokens $\mathbf{V} \in \mathbb{R}^{N \times d}$, instruction tokens $\mathbf{T} \in \mathbb{R}^{N_t \times d}$, target retention $\bar{r}$
Require: Pooling factor f, edge sensitivity τ, number of decoder layers L, pruning layers (l_e, l_l), default late keep ratio r_l^0, guidance weight w
Ensure: Pruned multimodal tokens $\mathbf{X}_{\text{out}}$

// Stage 1: Image-level pruning

$\tilde{Y} \leftarrow \text{Gaussian}\big(\text{AvgPool}_{f\times f}(\text{Luminance}(I))\big)$

$H_p \leftarrow -\sum_v \hat{P}_p(v) \ln \hat{P}_p(v), \quad \bar{H} \leftarrow \frac{1}{N}\sum_{p=1}^{N} H_p$

$M \leftarrow \|\nabla_{\text{Sobel}} \tilde{Y}\|_2, \quad e(x,y) \leftarrow \mathbf{1}[M(x,y) > (1-\tau)\,\text{median}(\tilde{Y})]$

$\rho_v \leftarrow \frac{1}{p_h p_w}\sum_{(x,y)\in p(v)} e(x,y)$

$\mathcal{I}_1 \leftarrow \{v \mid H_{p(v)} > \bar{H} \ \vee \ \rho_v > 0\}, \quad \mathbf{V}_1 \leftarrow \mathbf{V}[\mathcal{I}_1, :], \quad r_1 \leftarrow |\mathcal{I}_1|/N$

// Stage 2: Adaptive compression schedule

$\hat{r}_e \leftarrow \dfrac{L\bar{r}/r_1 - l_e}{(l_l - l_e) + r_l^0(L - l_l)}$ ▷ only if $r_1 > \bar{r}$

$$(r_e, r_l) \leftarrow \begin{cases} (1,1), & r_1 \le \bar{r} \quad \text{(Stage 2 skipped)}, \\ \left(1, \frac{L\bar{r}/r_1 - l_l}{L - l_l}\right), & \hat{r}_e \ge 1 \quad \text{(late only)}, \\ (\hat{r}_e, r_l^0), & \text{otherwise} \quad \text{(progressive)}. \end{cases}$$

// Hierarchical context-aware pruning

$\mathbf{V}^{(0)} \leftarrow \mathbf{V}_1, \quad \mathbf{T}^{(0)} \leftarrow \mathbf{T}$

for $\ell = 0, \ldots, L-1$ **do**

  $[\mathbf{V}^{(\ell+1)}; \mathbf{T}^{(\ell+1)}] \leftarrow \text{Decoder}_\ell\big([\mathbf{V}^{(\ell)}; \mathbf{T}^{(\ell)}]\big)$

  if $\ell + 1 \in \{l_e, l_l\}$ **then**

    $a_v \leftarrow \dfrac{1}{N_h}\displaystyle\sum_{h=1}^{N_h}\sum_{t=1}^{N_t} A_{t,v}^{(h)}$ ▷ N_v: current number of vision tokens

  end if

  if $\ell + 1 = l_e$ and $r_e < 1$ **then**

    $g_v \leftarrow \dfrac{\rho_v}{\text{std}(\rho)} + \dfrac{H_{p(v)}}{\text{std}(H)}, \quad z_v \leftarrow a_v + w\,\dfrac{\text{std}(a)}{\text{std}(g)}\,g_v, \quad \mathcal{I}_e \leftarrow \text{TopK}(z, \lfloor r_e N_v \rfloor)$

    $\mathbf{V}^{(\ell+1)} \leftarrow \mathbf{V}^{(\ell+1)}[\mathcal{I}_e, :]$

  else if $\ell + 1 = l_l$ and $r_l < 1$ **then**

    $\mathcal{I}_l \leftarrow \text{TopK}(a, \lfloor r_l N_v \rfloor), \quad \mathbf{V}^{(\ell+1)} \leftarrow \mathbf{V}^{(\ell+1)}[\mathcal{I}_l, :]$

  end if

end for

return $\mathbf{X}_{\text{out}} \leftarrow [\mathbf{V}^{(L)}; \mathbf{T}^{(L)}]$

A.3 Extended evaluation results

A.3.1 Evaluation on different model sizes

ERASE achieves the highest average accuracy at all three matched retention ratios on Qwen2.5-VL-3B and Qwen3-VL-4B, supporting robustness across model sizes (Tables 6 and 7). On Qwen2.5-VL-3B, its gains over the strongest baseline are 0.66, 2.39, and 6.31 points at 45%, 35%, and 25% retention, respectively. At 25% retention, ERASE preserves 92.85% of the base-model accuracy, compared with 84.33% for FastV. On Qwen3-VL-4B, the corresponding average-accuracy gains are 2.42, 4.49, and 3.58 points.

FastV and PDrop primarily use instruction relevance, while PruneSID relies on visual information. CDPruner and DART combine both sources but select tokens at a single point. iLLaVA distributes compression across stages, yet uses fixed stage-wise retention settings. ERASE instead jointly determines early token selection and image-dependent retention from raw-image statistics, then adjusts subsequent pruning to the resulting retention. Shallow pruning reuses these statistics alongside instruction relevance to preserve visual context, while deep pruning focuses on instruction relevance alone. These results demonstrate ERASE's effectiveness across diverse VLM architectures and model sizes.

Table 6: Results on Qwen2.5-VL-3B.

Retain↓	Method	TextVQA	ChartQA	DocVQA	InfoVQA	MathVista	$MMB_{v1.1}$	RWQA	MMStar	Avg.↑	Rel. (%)↑
100%	Base	79.06	83.96	93.07	75.95	60.30	77.48	66.41	55.93	74.02	100
47%	PDrop	76.94	70.08	80.92	58.20	51.60	73.92	60.13	52.33	65.52	88.51
45%	FastV	77.53	80.08	89.86	67.50	58.30	76.47	66.27	53.00	71.13	96.09
	DART	73.45	79.80	85.69	53.96	55.20	75.23	64.58	54.20	67.76	91.55
	CDPruner	77.68	73.52	78.24	53.22	54.90	76.08	63.92	53.53	66.39	89.69
	PruneSID	69.83	79.56	86.20	58.62	57.20	75.15	63.01	55.80	68.17	92.10
	iLLaVA	77.09	75.64	89.09	67.39	58.10	76.32	65.10	55.13	70.48	95.22
	ERASE	77.47	80.48	91.41	72.56	56.20	76.39	64.58	55.20	**71.79**	**96.98**
35%	FastV	76.21	76.16	85.35	61.95	55.40	75.77	64.58	52.40	68.48	92.51
	DART	70.12	75.36	78.54	44.59	52.80	75.23	61.70	53.60	63.99	86.45
	CDPruner	77.21	69.40	79.81	52.18	51.20	75.46	62.48	51.40	64.89	87.67
	PruneSID	63.44	75.60	81.54	51.96	56.30	73.30	64.31	54.67	65.14	88.00
	iLLaVA	75.42	68.16	84.77	60.69	56.70	76.24	63.79	54.87	67.58	91.30
	ERASE	76.52	78.76	90.99	70.22	55.60	75.93	64.84	54.13	**70.87**	**95.75**
25%	FastV	72.93	65.52	73.39	50.74	51.60	72.68	61.70	50.80	62.42	84.33
	DART	63.89	67.36	66.58	34.86	50.20	72.99	59.87	50.67	58.30	78.77
	CDPruner	75.61	62.28	75.13	48.49	47.50	74.30	59.87	50.47	61.71	83.36
	PruneSID	57.29	70.08	72.20	44.19	53.00	72.60	62.61	52.27	60.53	81.78
	iLLaVA	71.01	50.68	72.69	51.89	53.40	75.23	63.14	53.33	61.42	82.98
	ERASE	75.32	75.52	88.45	64.46	53.10	75.62	64.58	52.80	**68.73**	**92.85**

Table 7: Results on Qwen3-VL-4B.

Retain↓	Method	TextVQA	ChartQA	DocVQA	InfoVQA	MathVista	$MMB_{v1.1}$	RWQA	MMStar	Avg.↑	Rel. (%)↑
100%	Base	81.56	82.68	94.79	79.94	65.30	82.89	71.24	65.53	77.99	100
47%	PDrop	70.23	55.76	80.71	55.32	46.60	74.77	67.06	52.07	62.82	80.54
45%	FastV	78.96	61.64	85.57	62.87	46.30	76.32	66.93	55.20	66.72	85.55
	DART	71.54	73.00	83.08	57.84	56.10	79.41	68.89	56.73	68.32	87.60
	CDPruner	76.45	68.00	78.88	59.73	46.90	78.02	66.53	56.40	66.36	85.09
	PruneSID	74.51	76.80	92.44	70.92	53.80	77.48	70.59	57.67	71.78	92.03
	iLLaVA	76.38	73.00	91.42	71.31	48.40	78.72	72.68	58.47	71.30	91.42
	ERASE	79.74	77.92	94.09	75.78	54.50	80.26	70.07	61.27	**74.20**	**95.14**
35%	FastV	75.24	50.64	76.13	53.80	42.50	71.36	64.31	47.60	60.20	77.18
	DART	65.67	64.80	73.55	48.69	50.40	76.93	66.67	54.07	62.60	80.26
	CDPruner	73.19	60.04	70.81	53.62	43.20	75.62	66.27	50.80	61.69	79.10
	PruneSID	69.39	68.92	89.52	64.03	50.10	74.69	69.02	54.53	67.53	86.58
	iLLaVA	73.79	60.16	89.72	66.63	45.70	78.17	70.33	57.67	67.77	86.90
	ERASE	78.36	74.28	93.38	71.59	52.90	79.49	68.24	59.80	**72.26**	**92.65**
25%	FastV	67.60	34.60	61.34	41.05	37.70	62.69	58.82	43.60	50.93	65.30
	DART	57.79	53.72	57.84	37.48	45.30	74.54	64.18	48.60	54.93	70.43
	CDPruner	68.01	50.12	59.97	47.12	41.00	73.53	62.88	48.00	56.33	72.22
	PruneSID	64.77	61.20	84.31	55.46	45.90	71.52	66.41	51.13	62.59	80.25
	iLLaVA	72.84	52.96	87.96	54.44	45.20	77.17	69.93	56.13	64.58	82.80
	ERASE	76.48	67.76	90.66	63.59	47.50	76.47	67.71	55.07	**68.16**	**87.39**

A.3.2 Evaluation with downscaled inputs

AgilePruner, VScan, and Nüwa encounter OOM when processing inputs up to Qwen2.5-VL's default maximum of 16K visual tokens per image. Without an encoder `[CLS]` token for efficient attention-based scoring, their evaluated implementations require full attention maps, incurring substantial memory overhead. We therefore downscale their inputs to fit within the same GPU memory capacity.

Table 8: Accuracy and peak GPU memory on Qwen2.5-VL-7B.

Retain↓	Method	TextVQA	ChartQA	DocVQA	InfoVQA	MathVista	$MMB_{v1.1}$	RWQA	MMStar	Avg.↑	Peak. (GB)↓
100%	Base	85.30	86.00	94.82	82.31	65.10	82.51	70.20	65.53	78.97	19.37
45%	AgilePruner	83.60	84.56	94.15	75.40	63.80	81.58	67.32	64.20	76.83	38.81
	VScan	84.84	83.76	93.66	78.02	56.20	78.02	65.23	58.93	74.83	37.38
	Nüwa	83.75	84.60	94.36	77.51	65.60	81.58	69.54	63.73	77.58	39.59
	ERASE	84.52	83.24	94.22	79.27	64.20	81.27	70.20	64.13	**77.63**	**19.15**
35%	AgilePruner	82.26	80.68	92.27	70.19	63.10	80.65	67.45	62.87	74.93	38.81
	VScan	84.05	82.64	92.83	74.35	56.00	77.63	66.27	57.40	73.90	37.38
	Nüwa	82.82	83.60	93.68	75.02	64.50	81.04	70.07	62.73	76.68	39.59
	ERASE	84.26	83.04	94.12	76.89	63.70	81.66	69.80	63.93	**77.18**	**19.15**
25%	AgilePruner	80.12	78.20	88.10	64.42	57.60	80.80	67.19	60.93	72.17	38.81
	VScan	83.23	79.60	89.64	68.29	53.70	76.78	64.18	55.13	71.32	37.38
	Nüwa	80.51	81.60	92.22	70.04	61.40	81.27	69.80	61.93	74.85	39.59
	ERASE	83.50	80.20	93.35	71.83	62.20	81.50	69.41	62.60	**75.57**	**19.15**

Table 8 presents these results. AgilePruner and Nüwa accommodate approximately 4.7K visual tokens per image, whereas VScan, which requires two attention maps, accommodates approximately 2.1K. Despite downscaling, their peak memory reaches 37.38–39.59 GB, roughly twice that of the base model. In contrast, ERASE achieves the highest average accuracy at every evaluated retention ratio using lightweight image statistics for early redundancy removal. It handles up to 16K visual tokens per image while using only 19.15 GB of peak memory, approximately half that of these baselines.

A.3.3 Impact of τ for edge detection

As defined in Eq. 4, τ controls the threshold for edge detection. Increasing τ lowers the threshold and consequently retains more edge-containing patches, which generally improves accuracy at the cost of a lower compression ratio. To test the effect of this parameter, we evaluate $\tau \in \{0.35, 0.40, 0.45, 0.50\}$ using only the edge-based criterion in Stage 1. We report the average accuracy and compression ratio over the four benchmarks used in our main ablation study: ChartQA, RealWorldQA, InfoVQA, and MMStar.

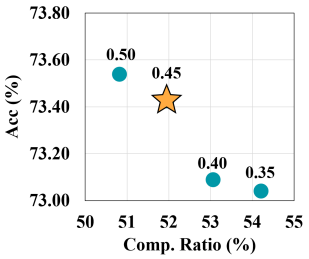


Figure 9: Accuracy and compression trade-off for the edge-threshold parameter τ.

As shown in Fig. 9, $\tau = 0.45$ provides a favorable operating point in the accuracy–compression trade-off. Increasing τ from 0.40 to 0.45 improves average accuracy by 0.34 points while reducing the compression ratio by 1.11 points. In contrast, a further increase from 0.45 to 0.50 reduces the compression ratio by a comparable 1.13 points but provides only a 0.11 point accuracy improvement. This diminishing return beyond $\tau = 0.45$ motivates our choice of $\tau = 0.45$ for all experiments.

A.3.4 End-to-end component ablation

Complementing the analysis in Fig. 7c, which measures retained accuracy on samples correctly answered by the base model, Table 9 reports task accuracy on the full evaluation sets of the same four benchmarks. All configurations use the same target layer-averaged visual token retention of 25%. VG denotes shallow-layer visual guidance, and ✓ and × indicate enabled and disabled components, respectively. Shallow instruction-aware pruning is enabled throughout. Disabling Stage 1 removes only its token pruning step, while the image statistics are still computed when visual guidance is enabled.

Stage 1 improves average accuracy across all four Stage 2 configurations. This improvement might be attributed to reduced attention dilution among similar visual tokens, as Stage 1 removes visually redundant tokens before instruction-aware selection. Adding visual guidance also improves average accuracy both with and without Stage 1, supporting the use of visual cues alongside instruction relevance at the shallow layer to preserve useful visual context. Finally, adding deep instruction-aware pruning improves average accuracy across all configurations under the same overall token budget. This supports hierarchical selection in Stage 2, where shallow pruning preserves visual context and subsequent deep pruning further narrows the retained tokens to instruction-relevant visual content. The complete configuration achieves the highest average accuracy of 71.01%.

Table 9: End-to-end component ablation on Qwen2.5-VL-7B.

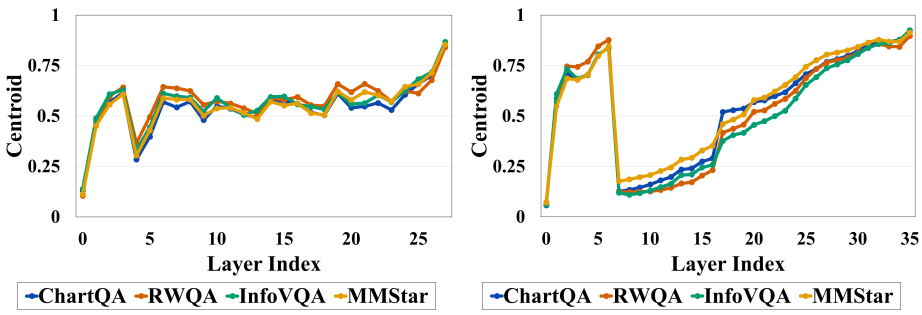
Stage 1	Shallow	VG	Deep	ChartQA	InfoVQA	RWQA	MMStar	Avg.
×	✓	×	×	73.84	58.53	65.23	56.60	63.55
×	✓	✓	×	75.80	61.74	66.80	58.33	65.67
×	✓	×	✓	73.64	65.23	67.71	58.00	66.15
×	✓	✓	✓	76.64	67.23	67.71	61.87	68.36
✓	✓	×	×	79.68	67.22	67.97	60.00	68.72
✓	✓	✓	×	79.68	68.08	67.45	61.00	69.05
✓	✓	×	✓	79.48	71.48	68.24	61.20	70.10
✓	✓	✓	✓	80.20	71.83	69.41	62.60	**71.01**

Table 10: Average efficiency across inputs with varying image resolutions.

Method	**KV Cache** (MB) ↓	**Prefill Latency** (ms) ↓	**Decode Latency** (ms/token) ↓	**Task Latency** (ms/task) ↓
Base	409.43	1990	37.82	2105
FastV	104.32 (×3.92)	1099 (×1.81)	34.17 (×1.11)	1206 (×1.75)
DART	104.48 (×3.92)	1113 (×1.79)	35.37 (×1.07)	1219 (×1.73)
CDPruner	104.33 (×3.92)	1560 (×1.28)	34.17 (×1.11)	1665 (×1.26)
PruneSID	104.34 (×3.92)	1568 (×1.27)	33.15 (×1.14)	1671 (×1.26)
iLLaVA	104.34 (×3.92)	1237 (×1.61)	33.45 (×1.13)	1341 (×1.57)
ERASE (Ours)	104.32 (×3.92)	1099 (×1.81)	34.62 (×1.09)	1205 (×1.75)

A.3.5 Efficiency across image resolutions

While Table 5 focuses on high-resolution inputs with 16K vision tokens, Table 10 reports efficiency averaged over inputs spanning a broad range of image resolutions. At lower resolutions, the additional overhead of CDPruner and PruneSID becomes less dominant, resulting in higher average speedups than in the 16K-token setting. Nevertheless, both remain slower than other lightweight methods. CDPruner incurs repeated embedding-space subset selection, while PruneSID additionally materializes an $N \times N$ similarity matrix outside the accelerator. iLLaVA also introduces repeated token-merging operations across its compression stages. In contrast, FastV and DART achieve low latency through a single lightweight selection, limiting pruning overhead. Although ERASE applies multiple pruning stages, Stage 1 computes lightweight visual cues directly on a single-channel luminance image, and Stage 2 performs instruction-aware selection on this reduced token set. ERASE consequently achieves an average task latency of 1205 ms, comparable to FastV and DART. Together with the higher average accuracy reported in Table 1, these results show that ERASE's adaptive, hierarchical pruning preserves accuracy without sacrificing the latency advantage of lightweight selection.



(a) InternVL3-8B (b) Qwen3-VL-8B

(c) Qwen3-VL-4B (d) Qwen2.5-VL-3B

Figure 10: Vision–text centroid similarity across decoder layers.

A.4 VISION–TEXT CENTROID SIMILARITY

To examine whether the layer-wise pattern observed on Qwen2.5-VL-7B extends to other models, we analyze the vision–text centroid similarity defined in Eq. 6 on InternVL3-8B, Qwen3-VL-8B, Qwen3-VL-4B, and Qwen2.5-VL-3B across four benchmarks (Fig. 10). Across these models, this alignment proxy exhibits a common pattern: an early rise, a subsequent drop, and recovery toward deeper layers, where its values are generally higher and more stable. Within each model, the benchmark curves closely follow one another, suggesting a largely consistent layer-wise pattern across the evaluated datasets.

Despite this shared pattern, the recovery trajectory varies across models. Qwen2.5-VL-7B reaches a mid-layer plateau with mild fluctuations before increasing further in deeper layers (Fig. 7a). InternVL3-8B shows a fluctuating recovery, both Qwen3-VL models exhibit a distinct upward transition after the drop, and Qwen2.5-VL-3B recovers more gradually with some mid-layer fluctuations. These observations extend the qualitative pattern motivating shallow and deep pruning candidates in Section 3.3, while showing that the recovery trajectory is model-specific.

A.5 VISUALIZATION OF RETAINED TOKENS AT EACH STAGE

Fig. 11 visualizes the image regions corresponding to retained visual tokens after each pruning step. Stage 1 removes visually redundant regions while preserving informative structures throughout the image. The shallow and deep pruning steps of Stage 2 progressively focus the retained tokens on instruction-relevant content. Across these examples, answer-relevant details within the red boxes remain visible as surrounding regions are increasingly pruned. These qualitative results illustrate how the two stages jointly reduce visual redundancy and preserve information needed to answer the instruction.

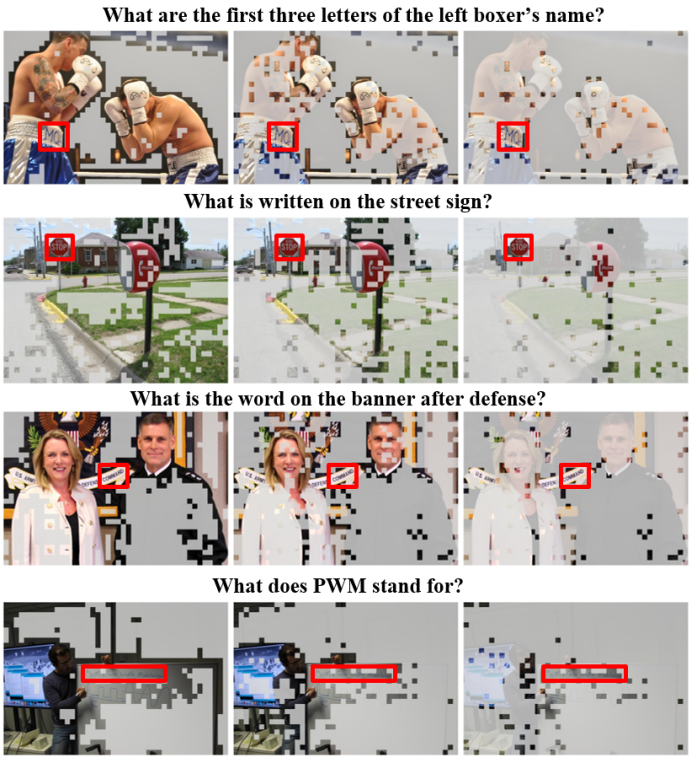


Figure 11: Retained visual tokens after Stage 1 (left), shallow Stage 2 pruning (middle), and deep Stage 2 pruning (right). Instructions appear above each row. Red boxes mark answer-relevant regions, and faded patches indicate pruned tokens.

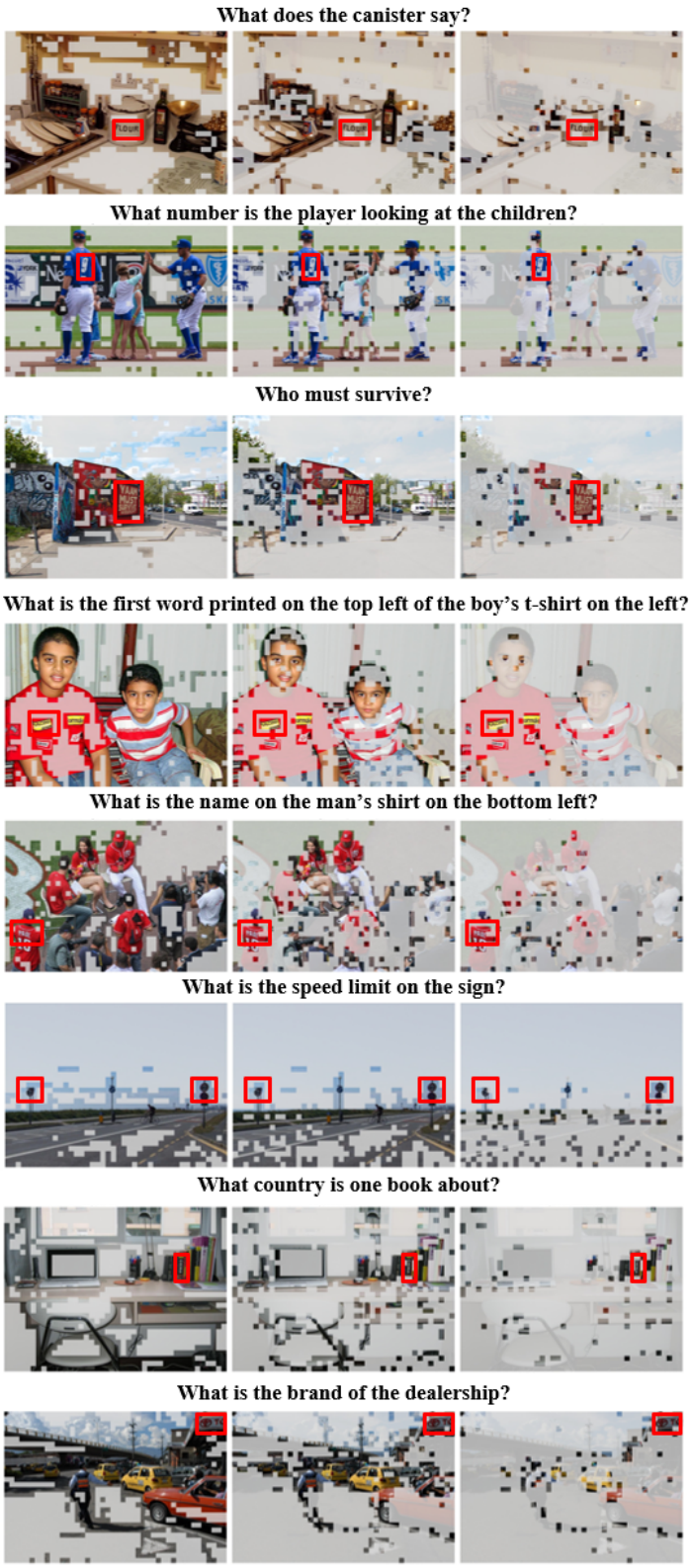


Figure 11: Retained visual token examples (continued).