CLASP: CLASS-ADAPTIVE LAYER FUSION AND DUAL-STAGE PRUNING FOR MULTIMODAL LARGE LANGUAGE MODELS

Anonymous authors
Paper under double-blind review

ABSTRACT

Multimodal Large Language Models (MLLMs) suffer from substantial computational overheads, driven by the massive redundancy of visual token sequences. Existing methods reduce this cost by pruning redundant visual tokens, typically using features from a single or fixed set of ViT layers together with a predefined token selection policy. However, adapting visual feature extraction and token selection to the varying information needs of different instructions remains challenging under these static designs. To address these limitations, we present *CLass-Adaptive Layer fusion and dual-Stage Pruning (CLASP)*, a plug-and-play token reduction framework that jointly adapts visual feature fusion and token pruning to each instruction. Specifically, we construct a category-specific visual representation to balance fine-grained visual detail and high-level semantics. Then we perform dual-stage pruning that allocates the token budget between attention-salient pivots (relevance) and redundancy-aware completion tokens (coverage). Mechanism analyses further clarify how category-conditioned fusion and relevance–coverage token selection support instruction-adaptive pruning. Extensive evaluations across diverse benchmarks, pruning ratios, and MLLM architectures demonstrate a favorable balance between accuracy and efficiency. Taken together, these results show that instruction-adaptive visual processing can reconcile aggressive token reduction with robust multimodal performance, offering a practical path toward MLLM deployment under constrained computational budgets.

1 INTRODUCTION

Multimodal Large Language Models (MLLMs) extend the instruction-following and reasoning capabilities of LLMs to the visual domain. These models typically align pre-trained vision encoders with autoregressive language decoders through an alignment module (Wu et al., 2023). Despite their impressive performance across various multimodal benchmarks (Liu et al., 2024d; Fu et al., 2026), MLLMs face significant challenges in practical utility. The primary bottleneck stems from the substantial computational overhead associated with processing visual tokens. In these architectures, input images are typically serialized into long, high-density sequences of patch-level tokens (Jin et al., 2025). This representation causes the overall computational cost of self-attention to scale quadratically with the input resolution. For instance, increasing the input resolution from 336×336 in LLaVA-1.5 (Liu et al., 2023) to 672×672 in LLaVA-NeXT (Liu et al., 2024b) raises the token count from 576 to a staggering $2,880$. Such a proliferation of tokens imposes excessive demands on both memory footprint and inference latency. Ultimately, these overheads create a major barrier to efficient inference and deployment (Kong et al., 2025).

To reduce the inference cost of MLLMs, recent studies (Xing et al., 2024; Bolya et al., 2022; Wen et al., 2025a; Zou et al., 2026) propose a range of token reduction approaches to prune visual tokens. Existing approaches can be broadly grouped into two primary directions. The first line of work leverages cross-modal attention scores (Chen et al., 2024a; Zhang et al., 2024b; Zhan et al., 2024; Arif et al., 2025; Zhang et al., 2025) to identify and retain instruction-relevant visual regions, primarily focusing on token relevance. The second line of work employs similarity-based pruning (Bolya et al., 2022; Xing et al., 2024; Yang et al., 2025; Alvar et al., 2025; Zhang et al., 2026) to remove redundant patches by merging or dropping low-saliency tokens, emphasizing the coverage of visual

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.

information. However, most existing methods derive visual representations from either a single ViT layer or a fixed set of layers. These static representations serve as both pruning signals and subsequent decoder inputs, yet they fail to capture task-specific nuances. Building on this static foundation, existing methods (Chen et al., 2024a; Zhang et al., 2024b; Wen et al., 2025a; Zou et al., 2026) then apply a rigid pruning strategy with predefined parameters, regardless of the instruction category. However, such methods overlook a critical nuance: different intents emphasize distinct levels of *visual abstraction* and impose varying requirements on spatial *relevance and coverage*.

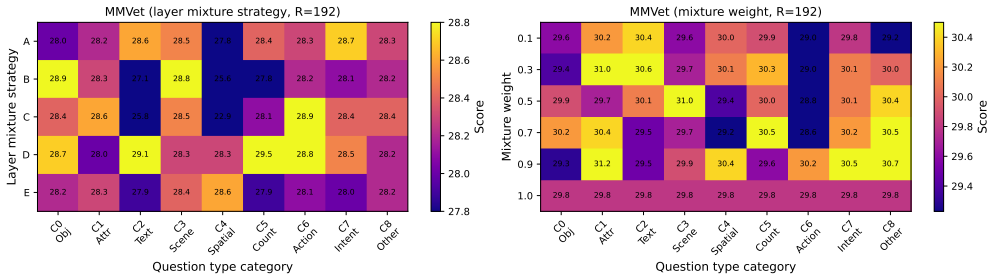

Figure 1: Visualization of class-adaptive configurations on MMVet ($R = 192$). The left panel shows layer mixtures by category, while the right panel shows relevance–coverage ratios.

Based on these insights, we contend that both visual feature extraction and token pruning should be dynamically adjusted according to input question categories. To verify this, we evaluate the sensitivity of specific instruction categories to different configurations under a fixed token budget (MM-Vet, LLaVA-v1.5-7B, $R = 192$). We first conduct a preliminary analysis on layer-mixture strategies to optimize visual feature extraction (Fig. 1, left). Specifically, we employ multiple layer-fusion schemes (A$\rightarrow$E) that shift the fusion emphasis from shallow to deep ViT layers. The results indicate that the resulting model accuracy varies significantly depending on the layer depth of the integrated features. For instance, the performance of the Count category degrades as the fusion emphasis shifts from hybrid to deep layers (29.5 *vs.*27.9). To further examine the impact of different pruning criteria, we investigate the limitations of relying on a fixed balance between attention and similarity (Fig. 1, right). We implement a hybrid strategy that dynamically switches between attention-based and similarity-based pruning. The findings show that a single, static mixing ratio fails to achieve optimal performance for every category. Specifically, text-based categories reach peak accuracy at lower ratios (≈ 0.3), whereas counting categories require higher ratios (≈ 0.7) in MMVet. These results confirm that effective visual token reduction requires class-aware adaptation of both layer-wise feature integration and the balance between relevance and coverage.

Building on these findings, we propose ***CL**ass-**A**daptive layer fusion and dual-**S**tage **P**runing (CLASP)*, a framework that dynamically fuses visual features and prunes tokens. We introduce *class-adaptive layer fusion* to optimize the visual representation. Instead of relying on a single ViT layer, this module adaptively integrates visual features from multiple layers, conditioned on the input instructions. This process preserves category-relevant local details while effectively removing redundant information. Building on these optimized representations, we implement a *class-adaptive dual-stage pruning strategy*. The pruning process consists of a relevance-preserving stage to protect key visual content. It is followed by a coverage-oriented stage to maintain the necessary context while further reducing redundancy. We evaluate our method across a comprehensive suite of eight image benchmarks and three video benchmarks. The results demonstrate that our approach consistently surpasses existing state-of-the-art token pruning methods. Notably, our method preserves approximately 94.7% of the original performance even with a nominal effective-budget reduction of 88.9%. Furthermore, after offline calibration, CLASP keeps the backbone frozen and readily integrates with diverse architectures, supporting efficient deployment in practical resource-constrained and on-device environments where visual-token redundancy increases inference costs and memory use.

Overall, our contributions can be summarized as follows: **(1)** Through empirical analysis, we identify category-specific preferences in representation depth and relevance–coverage allocation, revealing the limitations of static feature extraction and pruning policies. **(2)** We propose the CLASP framework to preserve instruction-relevant details and visual coverage through class-adaptive layer fusion and dual-stage pruning. **(3)** Extensive evaluations across multiple MLLM architectures and image and video benchmarks demonstrate consistent accuracy–efficiency improvements over strong baselines.

2 Related Work

Vision-language Models (VLMs). Multimodal large language models (MLLMs) enable visual reasoning by coupling vision encoders with LLMs through alignment interfaces, such as projectors Liu et al. (2023; 2024a) or query-based connectors (Li et al., 2023b; Dai et al., 2023). To enhance these multimodal capabilities, subsequent models (Zhu et al., 2024; Wang et al., 2024; Chen et al., 2024b; Zhu et al., 2025; Yang et al., 2026) have focused on scaling vision backbones, training data, and instruction-tuning recipes. However, as these models scale, the serialization of redundant visual inputs into long visual token sequences imposes a significant computational burden. This burden becomes particularly acute when handling high-resolution images, where the resulting explosion in sequence length leads to prohibitive inference latency (Liu et al., 2023; 2024b; An et al., 2025). Beyond efficiency, current MLLMs still struggle with fine-grained visual perception, often leading to perception gaps and hallucinations Li et al. (2023c); Yu et al. (2024); Dang et al. (2025). These perceptual shortcomings indicate a pressing need for more faithful visual representations and scalable context handling Zou et al. (2026); Wen et al. (2025b). A primary structural bottleneck is that most projector-based models extract tokens from only a single, static and often late, vision-encoder layer. By relying on such a fixed extraction point, these models fail to dynamically adapt the level of visual abstraction to the specific requirements of diverse instructions.

Visual token compression and pruning. Visual token compression and pruning have emerged as pivotal techniques to alleviate the quadratic attention cost of long sequences, particularly as MLLMs increasingly handle high-resolution inputs Huang et al. (2026); Dang et al. (2026). To achieve such efficient reduction, various methodologies have been developed, differing in importance signals and optimization goals, such as instruction relevance versus redundancy. Attention-based approaches, like FastV Chen et al. (2024a) and VTW Lin et al. (2025b), typically prune tokens using cross-modal attention patterns, while PyramidDrop Xing et al. (2024) operates on intra-modal attention inside the vision backbone. Instruction-guided methods like SparseVLM Zhang et al. (2024b) strengthen the role of textual instructions to condition these attention signals. However, attention-based criteria are often noisy and tend to retain highly duplicated tokens Qian et al. (2026). To address this, similarity-driven methods (*e.g.*, ToMe Bolya et al. (2022), VisionZip Yang et al. (2025), HiRED Arif et al. (2025), VisPruner Zhang et al. (2025),PruneSID Fang et al. (2026)) leverage intrinsic visual cues to manage redundancy and maintain representation diversity. Nonetheless, these approaches are limited by a fixed-layer visual readout, which fails to provide the adaptive level of visual abstraction required by diverse tasks. In contrast, CLASP leverages the inferred instruction category to dynamically guide both multi-layer visual feature fusion and hybrid attention-similarity pruning.

3 Preliminary

Architecture of MLLMs. We consider a multimodal LLM consisting of (i) a vision encoder with L layers, (ii) a lightweight vision–text interface (*e.g.*, a projector), and (iii) a causal text decoder. Specifically, the vision encoder extracts visual features from input images, which are then aligned with the text embedding space to guide the generation process. For sample n, let $\mathcal{V}_n$ denote the set of visual token indices. The vision encoder produces layer-wise *patch token representations* $\{\mathbf{Z}_n^{(l)}\}_{l=1}^{L}$, where $\mathbf{z}_{n,t}^{(l)} \in \mathbb{R}^{d_v}$ lies in the vision-encoder embedding space:

$$\mathbf{Z}_n^{(l)} \triangleq \{\mathbf{z}_{n,t}^{(l)}\}_{t\in\mathcal{V}_n}, \qquad l = 1, \ldots, L. \tag{1}$$

The interface module (*e.g.*, a projector) maps the selected/fused visual representations into the decoder embedding space, yielding *decoder-space visual tokens* $\tilde{\mathbf{Z}}_n$, which are concatenated with the text embeddings of the prompt x_n. The decoder generates $\mathbf{y}_n$ autoregressively:

$$p(\mathbf{y}_n \mid x_n, \tilde{\mathbf{Z}}_n) = \prod_i p\Big(y_{n,i} \mid \mathbf{y}_{n,<i}, x_n, \tilde{\mathbf{Z}}_n\Big). \tag{2}$$

where $y_{n,i}$ denotes the i-th response token and $\mathbf{y}_{n,<i}$ denotes the generated response prefix.

Attention mechanism for token relevance. Attention weights provide a training-free proxy for visual-token relevance (Zhang et al., 2024b; Wen et al., 2025b; Zhang et al., 2025). For an attention block over the multimodal sequence, let $\mathbf{Q}$, $\mathbf{K}$, and $\mathbf{V}$ denote the query, key, and value projections,

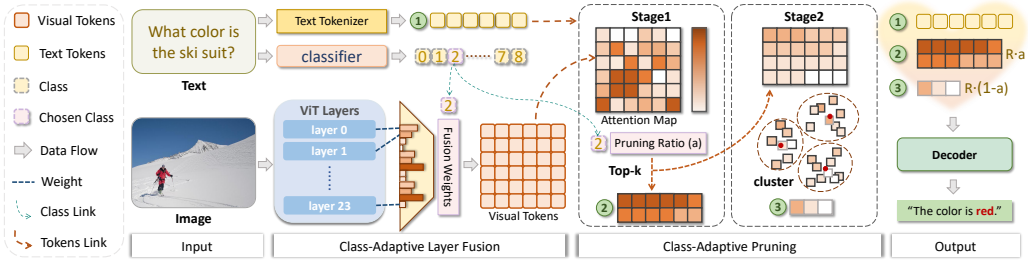

Figure 2: **Overview of CLASP.** A prompt-to-class router conditions visual processing on the instruction category. (i) **Class-Adaptive Layer Fusion:** Multi-layer ViT features are combined using class-specific mixture weights to capture category-appropriate visual abstractions. (ii) **Class-Adaptive Pruning:** At a given progressive pruning stage, the class-specific ratio a splits the local retention budget R for projected visual tokens between attention-based selection (Stage I) for instruction relevance and similarity-based clustering (Stage II) for visual coverage.

respectively, and let d_k denote the key dimension. The scaled dot-product attention matrix is $\mathbf{A}_n = \mathrm{softmax}(\mathbf{Q}\mathbf{K}^\top/\sqrt{d_k})$, and the corresponding output is $\mathbf{A}_n\mathbf{V}$. We average $\mathbf{A}_n$ over attention heads, where $\mathbf{A}_{n,i,t}$ denotes the attention from query token i to visual token t. We score each visual token $t \in \mathcal{V}_n$ by averaging the attention it receives from a reference set $\mathcal{S}$ (*e.g.*, instruction):

$$\phi_{n,t} = \frac{1}{|\mathcal{S}|}\sum_{i\in\mathcal{S}} \mathbf{A}_{n,i,t}, \qquad t \in \mathcal{V}_n. \tag{3}$$

Given a retention budget K, attention-based selection keeps the top-K highest-scoring tokens:

$$\mathcal{P}_n = \mathrm{Top}_K\big(\{\phi_{n,t}\}_{t\in\mathcal{V}_n}\big), \tag{4}$$

where $\mathrm{Top}_K(\cdot)$ returns indices of the K largest values.

Similarity mechanism for coverage. Attention is a strong relevance cue, yet attention-only pruning may retain many near-duplicate tokens and can be unstable under attention shift/dispersion (Zhang et al., 2025; Wen et al., 2025b). Therefore, several approaches incorporate similarity-based signals to ensure broader coverage (Zhang et al., 2025). Given token vectors $\{\tilde{\mathbf{z}}_{n,t}\}_{t\in\mathcal{V}_n}$, define normalized features $\mathbf{u}_{n,t} = \tilde{\mathbf{z}}_{n,t}/\|\tilde{\mathbf{z}}_{n,t}\|_2$ so that $\mathrm{sim}(t,t') = \mathbf{u}_{n,t}^\top\mathbf{u}_{n,t'}$ is cosine similarity. Let $\mathcal{P}_n$ be the attention-selected pivot set in Eq. (4). For any candidate token $t \in \mathcal{V}_n \setminus \mathcal{P}_n$, we define its *worst-case* redundancy with respect to the pivots as:

$$\rho_{n,t} = \max_{j\in\mathcal{P}_n} \mathbf{u}_{n,t}^\top\mathbf{u}_{n,j}. \tag{5}$$

A smaller $\rho_{n,t}$ indicates that token t is weakly covered by the pivot set (low cosine overlap) and thus provides more complementary information, which is preferred for enhancing contextual coverage.

4 METHOD

Overview. Given a prompt x_n and an input image $\mathbf{I}_n$, our goal is to reduce the *effective* number of visual tokens processed by the multimodal decoder to a target budget R, while preserving instruction-relevant content and maintaining token coverage. Following SparseVLM (Zhang et al., 2024b), we adopt a three-stage progressive pruning schedule and sparsify the visual token set at intermediate decoder layers $\{2, 6, 15\}$. The overall schedule is indexed by the effective budget R, with actual retention budgets B_1, B_2, B_3 at decoder layers 2, 6, and 15, respectively; the schedules are listed in Appendix C. Figure 2 illustrates the dual-stage pruning operation at a single pruning location. For simplicity, the local retention budget is denoted by R in the figure and its caption; it corresponds to B_s at progressive pruning stage s in the formulation below. Our method introduces two class-adaptive components driven by a prompt-to-class router: (i) *class-adaptive layer fusion*, which forms a category-conditioned mixture of multi-layer vision representations, and (ii) *class-adaptive pruning*, which allocates each stage budget between attention-salient pivots (relevance) and low-redundancy completion tokens (coverage). The detailed pseudocode is provided in Algorithm 1 in Appendix A.1.

4.1 CLASS-ADAPTIVE LAYER FUSION

Motivation and Design Principle. Vision backbones in MLLMs (*e.g.*, ViT-style encoders) typically encode visual information at different levels of abstraction across layers: shallower layers preserve local texture, edges, and fine spatial layouts, while deeper layers become increasingly semantic and invariant. Our study (Sec. 1, Fig. 1) indicates that *the optimal feature granularity for pruning depends on the instruction category*: detail-centric instructions (*e.g.*, OCR, counting) benefit from retaining mid/shallow cues, whereas holistic instructions (*e.g.*, scene understanding) prefer deeper abstractions. Therefore, instead of extracting visual representations from a fixed depth, we construct a prompt-conditioned depth mixture that adaptively balances fine-grained and semantic representations.

Prompt-to-class Routing. A text taxonomy defines $\mathcal{C} = \{0, 1, \ldots, C-1\}$, and a text-only router maps each prompt x_n to $c_n = \mathrm{Route}(x_n) \in \mathcal{C}$. For benchmark evaluation, categories are precomputed before MLLM inference. Table 5 reports runtime and peak GPU memory over the full workflow, including category preprocessing; Table 23 provides component timings.

Category-conditioned Layer Mixture. We encode category-specific layer preferences with a score matrix $\mathbf{W} \in \mathbb{R}^{C \times L}$, where the c-th row $\mathbf{w}_c$ assigns unnormalized importance to each vision layer (Appendix A). The category-wise configurations $(\mathbf{W}, \mathbf{a})$ are selected by offline discrete search on a labeled calibration set and fixed during evaluation. The MLLM backbone remains frozen. This avoids backbone finetuning but still requires calibration data and offline search. For sample n, we convert $\mathbf{w}_{c_n}$ to mixture weights via a temperature-controlled softmax $\boldsymbol{\alpha}_n = \mathrm{softmax}(\tau\, \mathbf{w}_{c_n})$:

$$\boldsymbol{\alpha}_n = \{\alpha_{n,l}\}_{l=1}^{L},\ \sum_{l=1}^{L} \alpha_{n,l} = 1, \tag{6}$$

where $\tau > 0$ controls the sharpness of the mixture (small τ: uniform averaging, large τ: near one-hot selection). This formulation strictly generalizes fixed-layer heuristics: choosing a single layer corresponds to a one-hot $\boldsymbol{\alpha}_n$.

Token-wise Fusion. Let $\mathbf{Z}_n^{(l)} = \{\mathbf{z}_{n,t}^{(l)}\}_{t \in \mathcal{V}_n}$ be layer-wise *patch representations* as in Eq. (1). Directly mixing intermediate activations across layers can be sensitive to scale differences. We then fuse multi-layer *features* via adaptive weights by a convex combination:

$$\bar{\mathbf{Z}}_n \triangleq \{\bar{\mathbf{z}}_{n,t}\}_{t \in \mathcal{V}_n}, \qquad \bar{\mathbf{z}}_{n,t} = \sum_{l=1}^{L} \alpha_{n,l}\, \mathbf{z}_{n,t}^{(l)}. \tag{7}$$

Convexity is deliberate, providing stability and interpretability (see Appendix B for formal properties). Finally, the frozen MLP projector maps fused features into the decoder embedding space:

$$\tilde{\mathbf{Z}}_n = f_{\text{proj}}(\bar{\mathbf{Z}}_n), \qquad \tilde{\mathbf{Z}}_n = \{\tilde{\mathbf{z}}_{n,t}\}_{t \in \mathcal{V}_n},\ \tilde{\mathbf{z}}_{n,t} \in \mathbb{R}^d. \tag{8}$$

While we assume ViT-style token alignment, hierarchical backbones are supported by mapping tokens to a common grid, and fusion adds $O(L\,|\mathcal{V}_n|\,d_v)$ overhead (Appendices A.3 and B.5).

4.2 CLASS-ADAPTIVE PRUNING

At progressive pruning stage s, we retain B_s visual tokens from the current candidate set. To simplify the single-stage description below, we write $B = B_s$ and suppress the stage index on the token sets and selection variables. B is an actual retention count, whereas R indexes the overall effective-budget setting. Conceptually, we want a subset that (i) preserves instruction-critical evidence (*relevance*) and (ii) avoids wasting budget on redundant patches (*coverage*). A single scalar scoring rule cannot reliably satisfy both criteria across heterogeneous instruction types. We therefore instantiate pruning as an explicit *two-stage* procedure, and make the relevance–coverage budget split category-dependent.

Category-dependent Split Ratio. We maintain a class-wise ratio vector $\mathbf{a} \in [0,1]^C$ and set $a_n \triangleq a_{c_n}$. Intuitively, large a_n allocates more slots to attention-salient tokens (useful for detail-centric prompts), while small a_n allocates more slots to coverage completion (useful for holistic prompts). Given the current stage budget B, we compute $K_1 = \lfloor a_n B \rfloor$ and $K_2 = B - K_1$.

Stage I: Attention Pivots for Relevance Preservation. We select K_1 *pivots* maximizing relevance:

$$\mathcal{P}_n = \mathrm{Top}_{K_1}\big(\{\phi_{n,t}\}_{t \in \mathcal{V}_n}\big). \tag{9}$$

This stage is intentionally aggressive in preserving query- or summary-attended evidence and acts as an anchor set that protects small but critical local visual regions (*e.g.*, text glyphs, counting targets). Appendix B formalizes the optimality of Top_{K_1} for additive relevance.

Stage II: Coverage Completion via Redundancy-aware Clustering. Attention pivots alone can be redundant: multiple high-attention tokens may correspond to near-identical patches. To complement them, we allocate K_2 slots to a clustering-based completion stage that explicitly improves coverage, consistent with the overview depicted in Fig. 2. Let us define $\mathcal{U}_n = \mathcal{V}_n \setminus \mathcal{P}_n$ and the normalized features $\mathbf{u}_{n,t} = \tilde{\mathbf{z}}_{n,t}/\|\tilde{\mathbf{z}}_{n,t}\|_2$. We first measure redundancy of a candidate token in the pivot set as:

$$\rho_{n,t}(\mathcal{P}_n) = \max_{j \in \mathcal{P}_n} \mathbf{u}_{n,t}^\top \mathbf{u}_{n,j}, \qquad t \in \mathcal{U}_n. \tag{10}$$

Instead of directly taking Bottom_{K_2}, we use $\rho_{n,t}$ to construct a deterministic, redundancy-minimizing initialization for the clustering phase: we choose the K_2 least redundant tokens as seeds and set their features as initial centers. We then run T iterations of spherical K-means (Hornik et al., 2012) on $\{\mathbf{u}_{n,t}\}_{t \in \mathcal{U}_n}$ computed with cosine similarity:

$$s_{n,t}^{(r)} = \arg \max_{k \in \{1,\dots,K_2\}} \mathbf{u}_{n,t}^\top \boldsymbol{\mu}_{n,k}^{(r-1)}. \tag{11}$$

Equation (11) assigns each candidate token to the current cluster center with the highest cosine similarity. Given these assignments, each cluster center is updated as the ℓ_2-normalized mean of its assigned token features. We alternate assignment and center update for T iterations. The exact update is given in Equation. (35) of Appendix B.

Finally, we select one representative (medoid) token per resulting cluster:

$$q_{n,k} = \arg \max_{t : s_{n,t}^{(T)} = k} \mathbf{u}_{n,t}^\top \boldsymbol{\mu}_{n,k}^{(T)}. \tag{12}$$

Equation (12) chooses the token closest to each final cluster center, so the output remains a set of discrete visual tokens rather than continuous centroids. The selected medoids form the completion set $\mathcal{Q}_n = \{q_{n,k}\}_{k=1}^{K_2}$, containing one representative token per cluster. Geometrically, the redundancy-aware seeding favors angular separation from the pivot set at initialization (low $\rho_{n,t}$), while the clustering refinement aims to reduce mutual duplication among completion tokens by encouraging them to cover multiple modes on the unit sphere. We use T=5 iterations by default, as supported by the reported clustering ablation in Table 18 across multiple benchmarks.

4.3 Retained Tokens and Decoder Inference.

At each pruning stage s, we combine the pivots $\mathcal{P}_n$ and medoids $\mathcal{Q}_n$ to retain B_s visual tokens:

$$\mathcal{V}_n \leftarrow \mathcal{P}_n \cup \mathcal{Q}_n, \qquad |\mathcal{V}_n| = B_s. \tag{13}$$

The retained visual hidden states continue through subsequent decoder layers under the progressive pruning schedule. The decoder generates the response $\mathbf{y}_n$ autoregressively:

$$p(\mathbf{y}_n \mid x_n, \mathbf{I}_n) = \prod_i p\Big(y_{n,i} \mid \mathbf{y}_{n,<i}, x_n, \mathbf{I}_n\Big). \tag{14}$$

This hybrid set balances instruction *relevance* through $\mathcal{P}_n$ and visual *coverage* through $\mathcal{Q}_n$.

5 Experiment

Benchmarks. We evaluate multimodal capability on ten established benchmarks spanning general visual understanding, compositional reasoning, OCR-centric reasoning, real-world robustness, object hallucination, and video comprehension. For image understanding, we use GQA (Hudson & Manning, 2019), MMBench (Liu et al., 2024d), MME (Fu et al., 2026), POPE (Li et al., 2023c), VQAv2 (Goyal et al., 2017), ScienceQA (Lu et al., 2022), and TextVQA (Singh et al., 2019). To assess temporal reasoning and video capability, we further include TGIF-QA (Jang et al., 2017), MSVD-QA, and MSRVTT-QA (Xu et al., 2017) for video question answering (abbreviated as TGIF, MSVD, and MSRVTT, respectively). We follow the official dataset splits and evaluation protocols,reporting the standard metrics for each benchmark to ensure reproducibility and fair comparison.

Table 1: Performance comparison on LLaVA-v1.5-7B under three effective visual-token budgets R.

Methods	GQA	MMB	MME	POPE	SQA	VQA-V2	TextVQA	Average
Upper Bound, 576 Tokens	61.9	64.7	1862	85.9	69.5	78.4	58.2	100.0%
LLaVA-1.5 7B, Effective budget $R = 192$ ($\downarrow$ **66.7%**)								
ToMe (ICLR23)(Bolya et al., 2022)	54.3	60.5	1563	72.4	65.2	68.0	52.1	88.5%
FastV (ECCV24)(Chen et al., 2024a)	52.7	61.2	1612	64.8	67.3	67.1	52.5	87.8%
MustDrop (2024.11)(Liu et al., 2024c)	58.2	62.3	1787	82.6	69.2	76.0	56.5	96.6%
LLaVA-PruMerge (ICCV25)(Shang et al., 2025)	54.3	59.6	1632	71.3	67.9	70.6	54.3	90.2%
PDrop (CVPR25)(Xing et al., 2024)	57.1	63.2	1766	82.3	68.8	75.1	56.1	96.0%
FiCoCo-V (2025.03)(Han et al., 2026)	58.5	62.3	1732	82.5	67.8	74.4	55.7	95.4%
HiRED (AAAI25)(Arif et al., 2025)	58.7	62.8	1737	82.8	68.4	74.9	47.4	93.9%
VisionZip (CVPR25)(Yang et al., 2025)	59.3	64.5	1767	86.4	68.9	76.8	57.3	98.1%
SparseVLM (ICML25)(Zhang et al., 2024b)	57.6	62.5	1721	83.6	69.1	75.6	56.1	95.9%
DART (EMNLP25)(Wen et al., 2025b)	58.9	63.6	1856	82.8	69.8	76.7	57.4	98.1%
HiPrune (ACL Findings 26)(Liu et al., 2025)	59.2	62.8	1814	86.1	68.9	76.7	57.6	98.0%
PruneSID (ICLR26)(Fang et al., 2026)	60.1	63.7	1791	86.9	68.5	76.8	56.7	98.1%
CLASP (ours)	60.4	61.3	1848	85.6	69.6	77.1	57.6	**98.4%**
LLaVA-1.5 7B, Effective budget $R = 128$ ($\downarrow$ **77.8%**)								
ToMe (ICLR23)(Bolya et al., 2022)	52.4	53.3	1343	62.8	59.6	63.0	49.1	80.4%
FastV (ECCV24)(Chen et al., 2024a)	49.6	56.1	1490	59.6	60.2	61.8	50.6	81.2%
MustDrop (2024.11)(Liu et al., 2024c)	56.9	61.1	1745	78.7	68.5	74.6	56.3	94.6%
LLaVA-PruMerge (ICCV25)(Shang et al., 2025)	53.3	58.1	1554	67.2	67.1	68.8	54.3	87.9%
PDrop (CVPR25)(Xing et al., 2024)	56.0	61.1	1644	82.3	68.3	72.9	55.1	93.6%
FiCoCo-V (2025.03)(Han et al., 2026)	57.6	61.1	1711	82.2	68.3	73.1	55.6	94.6%
HiRED (AAAI25)(Arif et al., 2025)	57.2	61.5	1710	79.8	68.1	73.4	46.1	91.9%
VisionZip (CVPR25)(Yang et al., 2025)	57.6	63.4	1768	84.7	68.8	75.6	56.8	96.8%
SparseVLM (ICML25)(Zhang et al., 2024b)	56.0	60.0	1696	80.5	67.1	73.8	54.9	93.3%
DART (EMNLP25)(Wen et al., 2025b)	57.9	63.2	1845	80.1	69.1	75.9	56.4	96.7%
HiPrune (ACL Findings 26)(Liu et al., 2025)	57.3	62.2	1782	82.8	68.3	74.9	56.6	96.0%
PruneSID (ICLR26)(Fang et al., 2026)	58.8	62.1	1749	86.5	68.3	75.3	54.7	96.3%
CLASP (ours)	58.9	60.7	1790	85.2	69.0	76.7	56.7	**97.0%**
LLaVA-1.5 7B, Effective budget $R = 64$ ($\downarrow$ **88.9%**)								
ToMe (ICLR23)(Bolya et al., 2022)	48.6	43.7	1138	52.5	50.0	57.1	45.3	70.1%
FastV (ECCV24)(Chen et al., 2024a)	46.1	48.0	1256	48.0	51.1	55.0	47.8	71.1%
MustDrop (2024.11)(Liu et al., 2024c)	53.1	60.0	1612	68.0	63.4	69.3	54.2	88.1%
LLaVA-PruMerge (ICCV25)(Shang et al., 2025)	51.9	55.3	1549	65.3	68.1	67.4	54.0	86.5%
PDrop (CVPR25)(Xing et al., 2024)	41.9	33.3	1092	55.9	68.6	69.2	45.9	72.7%
FiCoCo-V (2025.03)(Han et al., 2026)	52.4	60.3	1591	76.0	68.1	71.3	53.6	90.4%
HiRED (AAAI25)(Arif et al., 2025)	54.6	60.2	1599	73.6	68.2	69.7	44.2	88.0%
VisionZip (CVPR25)(Yang et al., 2025)	55.1	60.1	1690	77.0	69.0	72.4	55.5	92.8%
SparseVLM (ICML25)(Zhang et al., 2024b)	52.7	56.2	1505	75.1	62.2	68.2	51.8	86.5%
DART (EMNLP25)(Wen et al., 2025b)	55.9	60.6	1765	73.9	69.8	72.4	54.4	93.0%
HiPrune (ACL Findings 26)(Liu et al., 2025)	53.6	59.5	1646	73.0	68.9	69.2	54.9	90.5%
PruneSID (ICLR26)(Fang et al., 2026)	57.1	58.8	1733	83.8	67.8	73.7	54.2	94.1%
CLASP (ours)	57.0	59.1	1709	82.8	69.8	75.2	55.2	**94.7%**

Models and Comparison Methods. We apply our method to various MLLM architectures, including the LLaVA series: LLaVA-1.5 (Liu et al., 2024a) for image understanding, LLaVA-NeXT (Liu et al., 2024b) for high-resolution inputs, and Video-LLaVA (Lin et al., 2024) for video understanding, as well as the open-source model Qwen2.5-VL (Bai et al., 2025). For efficiency-oriented comparisons, we benchmark against state-of-the-art token reduction methods, including ToMe (Bolya et al., 2022), LLaVA-PruMerge (Shang et al., 2025), FastV (Chen et al., 2024a), HiRED (Arif et al., 2025), PDrop (Xing et al., 2024), Multi-Stage Vision Token Dropping (Liu et al., 2024c), SparseVLM (Zhang et al., 2024b), VisionZip (Yang et al., 2025), DART (Wen et al., 2025b), HiPrune Liu et al. (2025) and PruneSID Fang et al. (2026). More details regarding the model architectures and experimental results are provided in Appendix C, including benchmark descriptions and evaluation protocols.

Main Results. Table 1 compares CLASP with token merging and dropping methods on LLaVA-1.5-7B across seven diverse image-understanding benchmarks. We report raw task scores and compute the normalized average by averaging per-benchmark score ratios relative to the unpruned 576-token model. At effective budgets $R = 192, 128, 64$ (nominal reductions of 66.7%, 77.8%, and 88.9%), CLASP achieves the highest normalized averages among the compared methods: 98.4%, 97.0%, and 94.7%, respectively. CLASP leads on GQA at $R = 192$ (60.4) and scores 85.2 on POPE at $R = 128$. At $R = 64$, it exceeds the strongest baseline, PruneSID (94.1%), by 0.6 percentage points and DART (93.0%) by 1.7 points. Across the three budgets, gains over HiPrune are 0.4/1.0/4.2 percentage points, compared with 0.3/0.7/0.6 over PruneSID. These results indicate consistent improvements in average performance retention across the evaluated budget settings. As detailed in Appendix C, R denotes an effective-budget setting, matching budget labels do not imply identical FLOPs across methods.

Table 2: LLaVA-NeXT-7B performance at an effective budget of $R = 320$. Scores cover seven benchmarks. The average is normalized to the unpruned model. Best pruning results are bold blue.

Methods	GQA	MMB	MME	POPE	SQA	VQA_{V2}	VQA_{Text}	Average
Upper Bound, 2880 Tokens	64.2	67.4	1851	86.5	70.1	81.8	64.9	100.0%
LLaVA-NeXT 7B	*Effective budget* $R = 320$ (↓ 88.9%)							
FastV (ECCV24)(Chen et al., 2024a)	55.9	61.6	1661	71.7	62.8	71.9	55.7	87.8%
LLaVA-PruMerge (ICCV25)(Shang et al., 2025)	53.6	61.3	1534	60.8	66.4	69.7	50.6	83.6%
PDrop (CVPR25)(Xing et al., 2024)	56.4	63.4	1663	77.6	67.5	73.5	54.4	90.2%
MustDrop (2024.11)(Liu et al., 2024c)	57.3	62.8	1641	82.1	68.0	73.7	59.9	92.2%
FasterVLM(ICCV25)(Zhang et al., 2024a)	56.9	61.6	1701	83.6	66.5	74.0	56.5	91.6%
HiRED (AAAI25)(Arif et al., 2025)	59.3	64.2	1690	83.3	66.7	75.7	58.8	93.4%
SparseVLM (ICML25)(Zhang et al., 2024b)	56.1	60.6	1533	82.4	66.1	71.5	58.4	89.6%
DART (EMNLP25)(Wen et al., 2025b)	61.7	**65.3**	1710	84.1	**68.4**	**79.1**	58.7	95.3%
CLASP (ours)	**62.7**	63.2	**1723**	**85.8**	67.0	78.0	**61.7**	**95.7%**

Main Results on Higher-Resolution Inputs. Table 2 reports high-resolution evaluation results on LLaVA-NeXT-7B at an effective budget of $R = 320$, corresponding to an 88.9% nominal reduction from 2,880 visual tokens. Following the same evaluation protocol as Table 1, we report both raw benchmark scores and normalized averages for cross-benchmark comparison, with the unpruned 2,880-token model set to 100%. Under this aggressive visual-token budget, CLASP achieves the best normalized average of 95.7% among the compared methods, surpassing DART and HiRED by 0.4 and 2.3 percentage points, respectively. It also achieves the highest scores on four of the seven evaluated benchmarks: GQA (62.7), MME (1723), POPE (85.8), and TextVQA (61.7). Together, these gains on GQA, MME, POPE, and TextVQA indicate that CLASP preserves both global visual evidence and fine-grained textual details under high-resolution compression. Overall, the average and per-benchmark results support CLASP's effectiveness in preserving task performance across diverse evaluation tasks under aggressive high-resolution token pruning.

Results on Qwen Architecture. Beyond LLaVA, we evaluate CLASP on Qwen2.5-VL-7B (Bai et al., 2025) at pruning rates of 66.7%, 77.8%, and 88.9%. As shown in Table 3, CLASP achieves normalized averages of 96.5%, 94.4%, and 89.0%, exceeding SparseVLM (Zhang et al., 2024b) by 2.4, 3.6, and 6.1 percentage points, respectively. Notably, these gains are consistent across all five benchmarks and all three pruning rates, highlighting strong overall stability. The widening margin supports CLASP's cross-architecture robustness under aggressive visual-token pruning.

Table 3: Qwen2.5-VL-7B results at three pruning rates. Best scores are bolded.

Methods	MME	POPE	SQA	VQA_{Text}	MMB	Avg.
Upper Bound	2308	86.1	78.0	77.8	82.2	100%
Qwen2.5-VL-7B	*Token Pruning Rate = 66.7%*					
SparseVLM	2165	81.7	73.8	71.5	78.4	94.1%
CLASP (ours)	**2197**	**85.1**	**75.4**	**74.7**	**78.6**	**96.5%**
Qwen2.5-VL-7B	*Token Pruning Rate = 77.8%*					
SparseVLM	2086	77.7	72.8	68.7	75.6	90.8%
CLASP (ours)	**2154**	**83.0**	**74.5**	**73.1**	**76.2**	**94.4%**
Qwen2.5-VL-7B	*Token Pruning Rate = 88.9%*					
SparseVLM	1813	66.7	72.0	61.2	72.1	82.9%
CLASP (ours)	**2010**	**76.6**	**72.5**	**68.0**	**72.7**	**89.0%**

Main Results on Video Benchmarks. We further evaluate CLASP with Video-LLaVA to test whether its relevance–coverage strategy generalizes beyond static images. Table 4 compares CLASP with SparseVLM and the full-token upper bound on TGIF, MSVD, and MSRVTT. CLASP achieves the highest average score of 52.95, exceeding SparseVLM (51.22) and the unpruned model (52.23), with particularly clear gains on TGIF (45.65 vs. 43.47) and MSRVTT (51.58 vs. 51.28). These improvements suggest that removing repetitive spatial and temporal tokens helps the model focus on informative events, allowing the proposed pruning strategy to transfer effectively to video understanding across diverse temporal reasoning and question-answering settings.

Table 4: Video-LLaVA-7B results on three video benchmarks.

Method	TGIF	MSVD	MSRVTT	Average
Upper Bound	43.47	**61.93**	51.28	52.23
SparseVLM	44.67	59.29	49.69	51.22
CLASP (ours)	**45.65**	61.62	**51.58**	**52.95**

Efficiency Analysis. Table 5 compares CLASP with SparseVLM (Zhang et al., 2024b) and the full-token upper bound on POPE. With $R = 192$, CLASP provides a $1.5\times$ workflow-level speedup while retaining 99.6% of the upper-bound accuracy, compared with a $1.2\times$ speedup and 97.3% accuracy for SparseVLM. Under the more aggressive $R = 58$ setting, CLASP reaches a $2.1\times$ speedup and preserves 95.4% accuracy, whereas SparseVLM drops to 92.3%. The speedup mainly comes from progressive early pruning: once visual tokens are removed at decoder layers 2, 6, and 15, subsequent blocks process shorter multimodal sequences, reducing both self-attention and feed-forward computation. As the retained budget decreases, these downstream savings increasingly outweigh the routing and clustering overhead, explaining the larger speedup at $R = 58$.

Table 5: End-to-end POPE efficiency comparison at two token budgets.

Method	Time	Latency	Mem.	Acc.	Δ
	Retain 192 Tokens (↓ 66.7%)				
Upper Bound	49:41	0.335s	19.0G	100.0%	–
SparseVLM	40:51	0.275s	**15.8G**	97.3%	1.2×
CLASP (ours)	**33:08**	**0.223s**	17.6G	**99.6%**	**1.5×**
	Retain 58 Tokens (↓ 89.9%)				
Upper Bound	49:41	0.335s	19.0G	100.0%	–
SparseVLM	31:28	0.212s	**14.6G**	92.3%	1.6×
CLASP (ours)	**24:10**	**0.163s**	17.6G	**95.4%**	**2.1×**

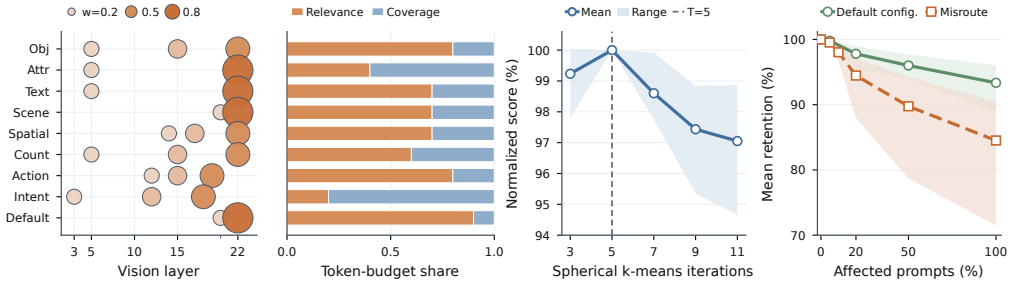

Figure 3: **Mechanism analysis of CLASP.** From left to right: category-specific layer weights, relevance–coverage budget splits, clustering iterations, and robustness to routing perturbations.

Mechanism Analysis. To verify that CLASP adapts both representation and pruning to instruction type, we visualize in Figure 3 the category-wise configurations selected on the balanced 18k-sample calibration set. Scene and default categories place 0.8 fusion weight on the deep layer L_{22}, whereas action and intention shift weight toward intermediate layers. Relevance-budget shares likewise range from 0.2 for intention to 0.9 for default. To determine the appropriate coverage-refinement depth, we vary the spherical k-means iterations over $T \in \{3, 5, 7, 9, 11\}$ at $R = 192$. Performance peaks at $T = 5$ and declines with further refinement. To assess routing robustness, we perturb increasing fractions of prompts and compare default fallback with random non-default misrouting across five benchmarks. When all prompts are affected, default fallback retains 93.4% of performance, versus 84.5% under random misrouting, supporting conservative fallback under uncertainty. Additional ablation results and detailed per-benchmark comparisons are provided in Appendix D.2.

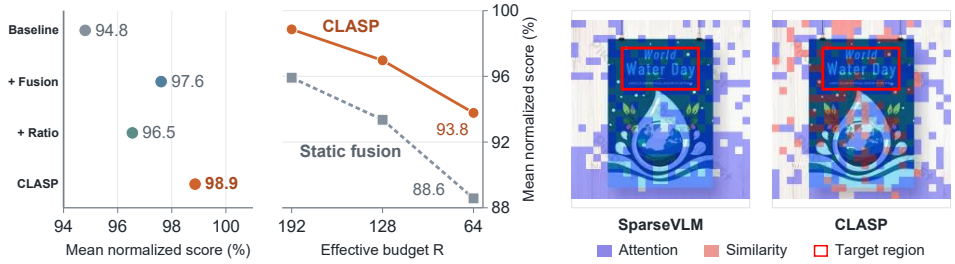

Figure 4: **Component analysis and token selection.** From left to right: component ablations, CLASP versus static fusion, and SparseVLM versus CLASP token selection on TextVQA.

Component Analysis and Token Selection. Figure 4 presents three complementary analyses on LLaVA-v1.5-7B. In the left panel, component ablations at $R = 192$ show that layer fusion and category-dependent budget splitting raise the normalized score from 94.8 to 97.6 and 96.5, while their combination reaches 98.9. In the middle panel, CLASP consistently outperforms the best static-fusion baseline, with the gap increasing from 2.9 to 5.2 percentage points as R decreases from 192 to 64. These scores are normalized to the corresponding unpruned model and averaged equally across MME, TextVQA, GQA, and POPE. In the right panels, the TextVQA example shows that CLASP concentrates attention-selected tokens on the target text and uses similarity-selected tokens to preserve complementary context, illustrating how the two stages balance relevance and coverage.

6 Conclusion

CLASP combines class-adaptive layer fusion with dual-stage pruning to preserve token relevance and spatial coverage. Evaluations across diverse multimodal benchmarks establish a favorable accuracy–efficiency trade-off. The framework reduces visual-token computation without finetuning the multimodal backbone. Its modular design supports existing MLLM architectures across practical deployment pipelines. Future work targets end-to-end learnable routing in place of offline-calibrated category configurations. Hardware-aware co-design offers another path toward efficient deployment on resource-constrained devices. These extensions could support practical multimodal inference for edge computing, embodied systems, and other limited-resource settings.

REPRODUCIBILITY STATEMENT

Our experiments use pretrained multimodal models with frozen parameters and follow official dataset splits and standard evaluation metrics. Class-specific layer-fusion weights and pruning ratios are selected through labeled offline calibration over a predefined search space and remain fixed during evaluation. At inference time, a text-only router assigns each prompt to a category that determines the fusion weights and relevance–coverage split. Token pruning follows a progressive schedule, with effective-budget settings distinguished from actual stage-wise retention counts. The experimental setup specifies the model configurations, evaluation toolkits, hardware, and pruning schedules needed to reproduce the evaluations.

AI USE STATEMENT

OpenAI Codex was used as an assistive tool during manuscript preparation. Its use was limited to language editing, improving the organization and clarity of technical descriptions, revising captions and appendix exposition, and assisting with LaTeX formatting and layout. It was not used to conduct experiments or generate the reported numerical measurements. All AI-assisted material was reviewed and, where necessary, corrected by the authors; the authors verified the technical claims, equations, citations, tables, and conclusions and take full responsibility for the final content of the paper.

REFERENCES

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

Xiang An, Yin Xie, Kaicheng Yang, Wenkang Zhang, Xiuwei Zhao, Zheng Cheng, Yirui Wang, Songcen Xu, Changrui Chen, Didi Zhu, et al. Llava-onevision-1.5: Fully open framework for democratized multimodal training. *arXiv preprint arXiv:2509.23661*, 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, 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*, 2025.

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.

David Chen and William B Dolan. Collecting highly parallel data for paraphrase evaluation. In *Proceedings of the 49th annual meeting of the association for computational linguistics: human language technologies*, pp. 190–200, 2011.

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

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

Wenliang Dai, Junnan Li, Dongxu Li, Anthony Tiong, Junqi Zhao, Weisheng Wang, Boyang Li, Pascale N Fung, and Steven Hoi. Instructblip: Towards general-purpose vision-language models with instruction tuning. *Advances in neural information processing systems*, 36:49250–49267, 2023.

Yunkai Dang, Donghao Wang, Jiacheng Yang, Yifan Jiang, Meiyi Zhu, Yuekun Yang, Cong Wang, Qi Fan, Wenbin Li, and Yang Gao. Fuse-rsvlm: Feature fusion vision-language model for remote sensing. *arXiv preprint arXiv:2512.24022*, 2025.

Yunkai Dang, Minxin Dai, Yuekun Yang, Zhangnan Li, Wenbin Li, Feng Miao, and Yang Gao. Uhr-bat: Budget-aware token compression vision-language model for ultra-high-resolution remote sensing. In *Forty-third International Conference on Machine Learning*, 2026.

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

Chaoyou Fu, Peixian Chen, Yunhang Shen, Yulei Qin, Mengdan Zhang, Xu Lin, Jinrui Yang, Xiawu Zheng, Ke Li, Xing Sun, et al. Mme: A comprehensive evaluation benchmark for multimodal large language models. *Advances in Neural Information Processing Systems*, 38, 2026.

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.

Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Alex Vaughan, et al. The llama 3 herd of models. *arXiv preprint arXiv:2407.21783*, 2024.

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 *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 3608–3617. IEEE, 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.

Kurt Hornik, Ingo Feinerer, Martin Kober, and Christian Buchta. Spherical k-means clustering. *Journal of statistical software*, 50:1–22, 2012.

Yihong Huang, Fei Ma, Yihua Shao, Jingcai Guo, Zitong Yu, Laizhong Cui, and Qi Tian. N\" uwa: Mending the spatial integrity torn by vlm token pruning. *arXiv preprint arXiv:2602.02951*, 2026.

Drew A Hudson and Christopher D Manning. Gqa: A new dataset for real-world visual reasoning and compositional question answering. In *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 6693–6702. IEEE, 2019.

Yunseok Jang, Yale Song, Youngjae Yu, Youngjin Kim, and Gunhee Kim. Tgif-qa: Toward spatio-temporal reasoning in visual question answering. In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 1359–1367. IEEE, 2017.

Yizhang Jin, Jian Li, Tianjun Gu, Yexin Liu, Bo Zhao, Jinxiang Lai, Zhenye Gan, Yabiao Wang, Chengjie Wang, Xin Tan, et al. Efficient multimodal large language models: A survey. *Visual Intelligence*, 3(1):27, 2025.

Zhenglun Kong, Yize Li, Fanhu Zeng, Lei Xin, Shvat Messica, Xue Lin, Pu Zhao, Manolis Kellis, Hao Tang, and Marinka Zitnik. Token reduction should go beyond efficiency in generative models–from vision, language to multimodality. *arXiv preprint arXiv:2505.18227*, 2025.

Bohao Li, Rui Wang, Guangzhi Wang, Yuying Ge, Yixiao Ge, and Ying Shan. Seed-bench: Benchmarking multimodal llms with generative comprehension. *arXiv preprint arXiv:2307.16125*, 2023a.

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, 2023b.

Kaiyuan Li, Xiaoyue Chen, Chen Gao, Yong Li, and Xinlei Chen. Balanced token pruning: Accelerating vision language models beyond local optimization. *arXiv preprint arXiv:2505.22038*, 2025.

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, 2023c.

Bin Lin, Yang Ye, Bin Zhu, Jiaxi Cui, Munan Ning, Peng Jin, and Li Yuan. Video-llava: Learning united visual representation by alignment before projection. In *Proceedings of the 2024 conference on empirical methods in natural language processing*, pp. 5971–5984, 2024.

Junyan Lin, Haoran Chen, Yue Fan, Yingqi Fan, Xin Jin, Hui Su, Jinlan Fu, and Xiaoyu Shen. Multi-layer visual feature fusion in multimodal llms: Methods, analysis, and best practices. In *2025 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 4156–4166. IEEE, 2025a.

Zhihang Lin, Mingbao Lin, Luxi Lin, and Rongrong Ji. Boosting multimodal large language models with visual tokens withdrawal for rapid inference. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 39, pp. 5334–5342, 2025b.

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

Haotian Liu, Chunyuan Li, Yuheng Li, and Yong Jae Lee. Improved baselines with visual instruction tuning. In *2024 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 26286–26296. IEEE, 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.

Jizhihui Liu, Feiyi Du, Guangdao Zhu, Niu Lian, Jun Li, and Bin Chen. Hiprune: Training-free visual token pruning via hierarchical attention in vision-language models. *arXiv e-prints*, pp. arXiv–2508, 2025.

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.

Taoyu Qian, Qi Wang, Daqian Shi, Yuanhao Jiang, Shang Gao, and Hualong Yu. E2s-pruner: Progressive two-stage evidence fusion for visual token pruning in vision-language models. *arXiv preprint arXiv:2608.23253*, 2026.

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

Amanpreet Singh, Vivek Natarajan, Meet Shah, Yu Jiang, Xinlei Chen, Dhruv Batra, Devi Parikh, and Marcus Rohrbach. Towards vqa models that can read. In *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 8309–8318. IEEE, 2019.

Peng Wang, Shuai Bai, Sinan Tan, Shijie Wang, Zhihao Fan, Jinze Bai, Keqin Chen, Xuejing Liu, Jialin Wang, Wenbin Ge, et al. Qwen2-vl: Enhancing vision-language model's perception of the world at any resolution. *arXiv preprint arXiv:2409.12191*, 2024.

Zichen Wen, Yifeng Gao, Weijia Li, Conghui He, and Linfeng Zhang. Token pruning in multimodal large language models: Are we solving the right problem? In *Findings of the Association for Computational Linguistics: ACL 2025*, pp. 15537–15549, 2025a.

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, 2025b.

Jiayang Wu, Wensheng Gan, Zefeng Chen, Shicheng Wan, and Philip S Yu. Multimodal large language models: A survey. In *2023 IEEE International Conference on Big Data (BigData)*, pp. 2247–2256. IEEE, 2023.

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

Dejing Xu, Zhou Zhao, Jun Xiao, Fei Wu, Hanwang Zhang, Xiangnan He, and Yueting Zhuang. Video question answering via gradually refined attention over appearance and motion. In *Proceedings of the 25th ACM international conference on Multimedia*, pp. 1645–1653, 2017.

Jun Xu, Tao Mei, Ting Yao, and Yong Rui. Msr-vtt: A large video description dataset for bridging video and language. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 5288–5296, 2016.

Jiacheng Yang, Anqi Chen, Yunkai Dang, Qi Fan, Cong Wang, Wenbin Li, Feng Miao, and Yang Gao. Annotation-free visual reasoning for high-resolution large multimodal models via reinforcement learning. *arXiv e-prints*, pp. arXiv–2602, 2026.

Senqiao Yang, Yukang Chen, Zhuotao Tian, Chengyao Wang, Jingyao Li, Bei Yu, and Jiaya Jia. Visionzip: Longer is better but not necessary in vision language models. In *2025 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 19792–19802. IEEE, 2025.

Tianyu Yu, Haoye Zhang, Yuan Yao, Yunkai Dang, Da Chen, Xiaoman Lu, Ganqu Cui, Taiwen He, Zhiyuan Liu, Tat-Seng Chua, et al. Rlaif-v: Aligning mllms through open-source ai feedback for super gpt-4v trustworthiness. *arXiv preprint arXiv:2405.17220*, 3(4):7, 2024.

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

Zheng Zhan, Zhenglun Kong, Yifan Gong, Yushu Wu, Zichong Meng, Hangyu Zheng, Xuan Shen, Stratis Ioannidis, Wei Niu, Pu Zhao, et al. Exploring token pruning in vision state space models. *Advances in Neural Information Processing Systems*, 37:50952–50971, 2024.

Qizhe Zhang, Aosong Cheng, Ming Lu, Zhiyong Zhuo, Minqi Wang, Jiajun Cao, Shaobo Guo, Qi She, and Shanghang Zhang. cls] attention is all you need for training-free visual token pruning: Make vlm inference faster. *arXiv preprint arXiv:2412.01818*, 2(3):4, 2024a.

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

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. *Advances in Neural Information Processing Systems*, 38:25438–25468, 2026.

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*, 2024b.

Deyao Zhu, Xiaoqian Shen, Xiang Li, Mohamed Elhoseiny, et al. Minigpt-4: Enhancing vision-language understanding with advanced large language models. In *International Conference on Learning Representations*, volume 2024, pp. 18378–18394, 2024.

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.

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.

APPENDIX CONTENTS

A Additional Method Details **16**

A.1 Detailed Algorithm 16

A.2 Calibration of $\mathbf{W}$ and $\mathbf{a}$ 16

A.3 Fusion with hierarchical backbones 17

B Additional Theoretical Analysis **19**

B.1 Notation and setup 19

B.2 Properties of class-adaptive layer fusion 19

B.3 Relevance–Coverage Optimization 21

B.4 Geometric Diversity Guarantees 23

B.5 Computational complexity 23

C Detailed Experiment Settings **24**

D Additional Experiments **30**

D.1 Calibration and Routing Evaluations 30

D.2 Additional Results and Ablations 31

D.3 Qualitative Case Studies 43

A Additional Method Details

A.1 Detailed Algorithm

Algorithm 1 summarizes the complete execution procedure of CLASP. The text-only router first assigns the prompt to a category, which determines both the layer-mixture weights and the relevance–coverage split ratio. The corresponding multi-layer visual features are fused and projected into the decoder space. At each progressive pruning layer, Stage I retains the most attention-salient pivot tokens, while Stage II initializes spherical K-means with low-redundancy candidates, refines the clusters, and selects one medoid from each cluster. The pivot and medoid sets are merged at each stage and reduced according to the actual retention counts B_1, B_2, B_3. The final set has B_3 tokens. R denotes the effective-budget setting associated with this schedule.

Algorithm 1 Class-adaptive layer fusion and pruning.

Require: prompt x_n, layer-wise tokens $\{\mathbf{Z}_n^{(l)}\}_{l=1}^{L}$ with indices $\mathcal{V}_n$, effective budget R with actual stage budgets (B_1, B_2, B_3), router $\mathrm{Route}(\cdot)$, layer-score matrix $\mathbf{W}$, split ratios $\mathbf{a}$, temperature τ, projector f_{proj}, reference set $\mathcal{S}$

1: $c_n \leftarrow \mathrm{Route}(x_n)$
2: $\boldsymbol{\alpha}_n \leftarrow \mathrm{softmax}(\tau\, \mathbf{w}_{c_n})$
3: $\tilde{\mathbf{Z}}_n \leftarrow f_{\text{proj}}\Big(\sum_{l=1}^{L} \alpha_{n,l}\, \mathbf{Z}_n^{(l)}\Big)$
4: $\mathcal{V} \leftarrow \mathcal{V}_n$
5: **for** $(l_s, B_s) \in \{(2, B_1), (6, B_2), (15, B_3)\}$ **do**
6: Compute relevance $\{\phi_{n,t}\}_{t\in\mathcal{V}}$ at layer l_s
7: $K_1 \leftarrow \lfloor a_{c_n} B_s \rfloor, \quad K_2 \leftarrow B_s - K_1$
8: $\mathcal{P} \leftarrow \mathrm{Top}_{K_1}(\{\phi_{n,t}\}_{t\in\mathcal{V}})$ (Eq. (9))
9: Initialize centers using $\mathrm{Bottom}_{K_2}(\{\rho_{n,t}(\mathcal{P})\})$ (Eq. (10))
10: Refine centers via spherical K-means for T iterations
11: $\mathcal{Q} \leftarrow$ Select cluster medoids (Eq. (36))
12: $\mathcal{V} \leftarrow \mathcal{P} \cup \mathcal{Q}$
13: **end for**
14: **return** $\{\tilde{\mathbf{z}}_{n,t}\}_{t\in\mathcal{V}}$

Algorithm 1 follows the boundary conventions in Appendix B.3. When $K_1 = 0 < K_2$, line 9 uses deterministic farthest-first initialization instead of pivot-relative Bottom-K_2 seeding. When $K_2 = 0$, set $\mathcal{Q} = \varnothing$ and skip lines 9–11.

A.2 Calibration of $\mathbf{W}$ and $\mathbf{a}$

The adaptive parameters in our method are low dimensional ($C \times L$ layer scores and C split ratios) and can be set without finetuning any parameters of the underlying MLLM (vision encoder, projector, or LLM). We use offline calibration to select category-specific configurations without updating the backbone. Unless an ablation specifies otherwise, calibration uses 18,000 labeled samples balanced across nine categories (2,000 per category), as described in Appendix C.

Data and objective. The offline search uses a labeled calibration set $\mathcal{D}_{\text{cal}} = \{(x_n, \mathrm{img}_n, y_n^\star, c_n)\}_{n=1}^{N_{\text{cal}}}$ that contains (i) prompts x_n, (ii) images img_n, (iii) reference outputs $y_n^\star$ (*e.g.*, gold answers for QA / multiple-choice labels), and (iv) category labels $c_n \in \mathcal{N}_c$. Category labels follow the Qwen3-8B intent-classification and heuristic-mapping procedure described in Appendix C. Reference outputs are used to score candidate configurations during offline calibration, with the MLLM backbone frozen.

For a fixed effective token budget R, we select $(\mathbf{W}, \mathbf{a})$ to maximize the task score under pruning:

$$\max_{\mathbf{W},\mathbf{a}} \ \mathrm{Score}(\mathcal{D}_{\text{cal}}.R, \mathbf{W}, \mathbf{a}) \quad \text{s.t. } a_c \in [0,1],\ \forall c. \tag{15}$$

This objective selects category-conditioned layer weights and split ratios under a fixed effective budget. Only these lightweight configurations are searched, leaving the vision encoder, projector, and language decoder frozen throughout calibration.

Here $\mathrm{Score}(\cdot)$ uses the same decoding and evaluation protocol as in the main experiments, but restricted to $\mathcal{D}_{\text{cal}}$. Concretely, we write

$$\mathrm{Score}(\mathcal{D}_{\text{cal}}.R, \mathbf{W}, \mathbf{a}) = \frac{1}{|\mathcal{D}_{\text{cal}}|} \sum_{(x_n, \mathrm{img}_n, y_n^\star, c_n) \in \mathcal{D}_{\text{cal}}} \mathrm{Eval}(\hat{y}_n(R, \mathbf{W}, \mathbf{a}),\ y_n^\star)\,, \tag{16}$$

where $\hat{y}_n(R, \mathbf{W}, \mathbf{a})$ is the model prediction produced when applying our class-adaptive layer fusion and dual-stage pruning under budget R, and $\mathrm{Eval}(\cdot)$ is the benchmark metric (*e.g.*, exact-match accuracy for QA, multiple-choice accuracy for MMBench-style evaluation, or any dataset-specific scorer).

Why calibration does not require MLLM finetuning. The search variables $(\mathbf{W}, \mathbf{a})$ specify layer mixtures and budget splits rather than updates to the vision encoder, projector, or language decoder. We evaluate discrete candidates with forward passes and select their configurations using reference outputs from the calibration set. The candidate prototypes and intent-mapping rules are predefined, whereas the category-wise choice of fusion configuration and split ratio is data-driven. Once selected, these configurations remain fixed during evaluation. Only the prompt-dependent category selection changes. CLASP combines labeled offline configuration selection with prompt-to-class prediction while keeping the MLLM backbone frozen. For benchmark evaluation, prompt categories are computed in a preprocessing pass and subsequently used for MLLM answer generation.

Per-class decomposition. Since c_n selects the row $\mathbf{w}_{c_n}$ and ratio a_{c_n}, the calibration naturally decomposes over categories: each class c can be calibrated on the subset $\mathcal{D}_{\text{cal}}^{(c)} = \{(x_n, \mathrm{img}_n, y_n^\star, c_n) \in \mathcal{D}_{\text{cal}} :\ c_n = c\}$. independently, which improves stability and reduces search complexity. Algorithm 2 summarizes a simple per-class procedure.

Efficient evaluation in practice. Calibration can be further accelerated by: (i) caching the frozen vision-encoder layer outputs $\{Z_n^{(l)}\}_{l=1}^{L}$ for each (x_n, img_n) once, so re-evaluating a candidate $\mathbf{w}_c$ only requires a cheap weighted sum (fusion) plus the standard forward decode; (ii) using a small, balanced $\mathcal{D}_{\text{cal}}$ per category (hundreds to a few thousand samples typically suffice); and (iii) optionally using early rejection (evaluate candidates on a small subset first, and fully score only the top few).

Structured parameterization for fast search. To avoid overfitting and to reduce search complexity, we recommend a structured parameterization: (i) restrict $\mathbf{w}_c$ to have support on a small candidate layer set (*e.g.*, 3–5 layers spanning shallow/mid/deep), and (ii) restrict a_c to a coarse grid (*e.g.*, $\{0.2, 0.4, 0.6, 0.8\}$ or a slightly denser grid when needed). This yields a compact search space while preserving the key adaptivity signals revealed by our motivation study.

Optional continuous refinement (if desired). If one wants to refine beyond a discrete candidate set, one can keep the same constraints but allow non-uniform weights on the selected layers (still normalized by softmax), and use derivative-free optimizers (*e.g.*, coordinate search / random search) on the low-dimensional parameters. This is optional and not required for stable gains.

A.3 Fusion with hierarchical backbones

Our exposition assumes token indices align across layers (a standard property of ViT-style encoders). For hierarchical encoders with resolution changes (*e.g.*, feature pyramids), fusion can be supported by mapping each layer to a common token set before applying Eq. (7).

Token alignment via spatial resampling. Let layer l output a grid of patch tokens with spatial resolution $H_l \times W_l$ (excluding any special token), and let the target resolution be $H_\star \times W_\star$ (typically the final layer resolution). We define an alignment operator $\mathrm{Align}_l(\cdot)$ that maps $Z_n^{(l)} \in \mathbb{R}^{(H_l W_l) \times d_v}$ to $\hat{Z}_n^{(l)} \in \mathbb{R}^{(H_\star W_\star) \times d_v}$:

$$\hat{Z}_n^{(l)} = \mathrm{Align}_l\Big(Z_n^{(l)}\Big), \qquad \mathrm{Align}_l = \begin{cases} \text{bilinear/nearest interpolation (upsample)} & \text{if } H_l W_l < H_\star W_\star, \\ \text{average pooling / strided pooling (downsample)} & \text{if } H_l W_l > H_\star W_\star, \\ \text{identity} & \text{if } H_l W_l = H_\star W_\star. \end{cases} \tag{17}$$

In practice, we reshape tokens back to a feature map of size $H_l \times W_l \times d_v$, apply standard 2D resampling, and flatten back to a sequence. If the backbone includes a special token (*e.g.*, [CLS]), we either (i) keep it separate and fuse it with the same mixture weights, or (ii) drop it when it is not used by the downstream projector.

Algorithm 2 Discrete Subspace Search for Optimal Layer Prototypes and Pruning Ratios.

Require: calibration set $\mathcal{D}_{\text{cal}}$ with labels c_n; candidate layer sets $\{\mathcal{L}^{(m)}\}$; candidate ratios $\mathcal{A}$; effective budget R and its associated progressive schedule

for $c \in \mathcal{N}_c$ **do**
  Initialize best score $s^\star \leftarrow -\infty$
  for layer candidate $\mathcal{L}^{(m)}$ **do**
    for $a \in \mathcal{A}$ **do**
      Define $\mathbf{w}_c$ supported on $\mathcal{L}^{(m)}$ (*e.g.*, uniform over $\mathcal{L}^{(m)}$, $-\infty$ elsewhere)
      Evaluate Score on $\mathcal{D}_{\text{cal}}^{(c)}$ using $(\mathbf{w}_c, a)$
      if score $> s^\star$ **then**
        Update $(\mathbf{w}_c^\star, a_c^\star) \leftarrow (\mathbf{w}_c, a)$, and $s^\star \leftarrow$ score
      end if
    end for
  end for
end for
Return $\mathbf{W} = [\mathbf{w}_0^\star; \ldots; \mathbf{w}_{C-1}^\star]$, $\mathbf{a} = [a_0^\star, \ldots, a_{C-1}^\star]$

After alignment, layer fusion proceeds exactly as in Eq. (7) by mixing the aligned tokens $\{\hat{Z}_n^{(l)}\}_{l=1}^{L}$. The computational overhead of fusion remains linear in the number of visual tokens (and linear in the number of fused layers); see Appendix B.5 for a step-wise complexity breakdown.

B Additional Theoretical Analysis

B.1 Notation and setup

For sample n, let $M_n \triangleq |\mathcal{V}_n|$ denote the number of visual patch tokens. Layer-wise tokens are $\mathbf{z}_{n,t}^{(l)} \in \mathbb{R}^{d_v}$ for $l \in \{1, \dots, L\}$ and $t \in \mathcal{V}_n$. Class-adaptive fusion produces $\bar{\mathbf{z}}_{n,t} \in \mathbb{R}^{d_v}$ and aligned tokens $\tilde{\mathbf{z}}_{n,t} = f_{\text{proj}}(\bar{\mathbf{z}}_{n,t}) \in \mathbb{R}^d$.

Selection operators. For a collection of real-valued scores $\{s_t\}_{t\in\mathcal{I}}$ indexed by $\mathcal{I}$, we denote by $\text{Top}_K(\{s_t\}_{t\in\mathcal{I}})$ the index set of the K largest scores, and by $\text{Bottom}_K(\{s_t\}_{t\in\mathcal{I}})$ the index set of the K smallest scores. When ties occur, any deterministic tie-breaking rule suffices and does not affect the statements below.

Simplex notation. We denote the probability simplex by

$$\Delta^{L-1} \triangleq \Big\{\boldsymbol{\gamma} \in \mathbb{R}^L \ : \ \gamma_l \geq 0 \ (\forall l), \ \sum_{l=1}^{L} \gamma_l = 1\Big\}. \tag{18}$$

Unit-normalization for cosine similarity. Whenever cosine similarity is used, we work with ℓ_2-normalized features

$$\mathbf{u}_{n,t} \triangleq \frac{\tilde{\mathbf{z}}_{n,t}}{\|\tilde{\mathbf{z}}_{n,t}\|_2} \in \mathbb{R}^d. \tag{19}$$

This normalization isolates *directional* information and makes inner products $\mathbf{u}_{n,t}^\top \mathbf{u}_{n,j}$ equal cosine similarity, which is the natural notion of redundancy/novelty on the unit sphere. Consequently, Euclidean distance on this manifold becomes strictly monotonic with respect to angular separation, simplifying the geometric proofs in Section B.4.

B.2 Properties of class-adaptive layer fusion

Fusion equations in execution order. For clarity, we restate the fusion module as a short sequence of equations.

1. **Routing and mixture weights.** Given the prompt x_n, a text-only router predicts a class c_n, which selects a row $\mathbf{w}_{c_n} \in \mathbb{R}^L$ from the layer-score matrix $\mathbf{W}$. We then convert it into a probability distribution over layers:

$$c_n \leftarrow \text{Route}(x_n), \qquad \boldsymbol{\alpha}_n \triangleq \text{softmax}(\tau\, \mathbf{w}_{c_n}) \in \Delta^{L-1}. \tag{20}$$

   Here $\tau > 0$ is a temperature that controls how "peaky" the layer preference is.

2. **Token-wise convex fusion across layers.** For each token index $t \in \mathcal{V}_n$, we fuse its layer-wise representations by a weighted sum. This dynamically integrates features from varying levels of visual abstraction:

$$\bar{\mathbf{z}}_{n,t} = \sum_{l=1}^{L} \alpha_{n,l} \mathbf{z}_{n,t}^{(l)}. \tag{21}$$

3. **Projection into the decoder space.** The fused token is mapped to the decoder embedding space through f_{proj}:

$$\tilde{\mathbf{z}}_{n,t} = f_{\text{proj}}(\bar{\mathbf{z}}_{n,t}) \in \mathbb{R}^d. \tag{22}$$

Lemma B.1 (Convexity of fused features) *For any sample n and token index $t \in \mathcal{V}_n$, the fused token in Eq.* (21) *lies in the convex hull of $\{\mathbf{z}_{n,t}^{(l)}\}_{l=1}^L$.*

Proof B.1 *By Eq.* (20), $\boldsymbol{\alpha}_n = \text{softmax}(\tau \mathbf{w}_{c_n})$ *satisfies* $\alpha_{n,l} \geq 0$ *and* $\sum_{l=1}^L \alpha_{n,l} = 1$, *hence Eq.* (21) *is a convex combination. Geometrically, the fused vector lies inside the layer-wise feature polytope.*

Interpretation. Eq. (21) keeps each fused token within the convex hull of its layer-wise representations. By the triangle inequality, the fused token satisfies $\|\bar{z}_{n,t}\|_2 \le \max_l \|z_{n,t}^{(l)}\|_2$.

Lemma B.2 (Limits of temperature-scaled softmax) *Let $\boldsymbol{\alpha}(\tau) = \mathrm{softmax}(\tau \mathbf{w})$ for any fixed $\mathbf{w} \in \mathbb{R}^L$. (i) As $\tau \to 0$, $\boldsymbol{\alpha}(\tau) \to (1/L)\mathbf{1}$. (ii) As $\tau \to \infty$, $\boldsymbol{\alpha}(\tau)$ converges to a one-hot distribution supported on $\arg\max_l w_l$ when the maximizer is unique.*

Proof B.2 *(i) $\mathrm{softmax}(0 \cdot \mathbf{w})$ is uniform. Continuity gives the limit. (ii) For large τ, $\exp(\tau w_l)$ is dominated by the largest w_l, yielding the standard softmax one-hot asymptotics, effectively selecting the single maximal layer.*

Assumption 1 (Bounded layer-wise token norms) *There exists $B > 0$ such that for all n, t, l, $\|\mathbf{z}_{n,t}^{(l)}\|_2 \le B$, uniformly bounding the feature magnitude.*

Lemma B.3 (Softmax Lipschitz continuity) *Let $\boldsymbol{\alpha}(\mathbf{u}) = \mathrm{softmax}(\mathbf{u})$. Then for any $\mathbf{u}, \mathbf{v} \in \mathbb{R}^L$, representing arbitrary input logit vectors,*

$$\|\boldsymbol{\alpha}(\mathbf{u}) - \boldsymbol{\alpha}(\mathbf{v})\|_1 \le \frac{1}{2}\|\mathbf{u} - \mathbf{v}\|_1. \tag{23}$$

Consequently, with $\boldsymbol{\alpha}(\tau\mathbf{w}) = \mathrm{softmax}(\tau\mathbf{w})$, applied to the temperature-scaled inputs,

$$\|\boldsymbol{\alpha}(\tau\mathbf{w}) - \boldsymbol{\alpha}(\tau\mathbf{w}')\|_1 \le \frac{\tau}{2}\|\mathbf{w} - \mathbf{w}'\|_1. \tag{24}$$

Proof B.3 *We make the dependence explicit in three steps, mirroring the execution logic.*

(a) ***Jacobian form.*** *Let $J(\mathbf{u}) = \nabla_{\mathbf{u}}\boldsymbol{\alpha}(\mathbf{u})$ with entries $J_{ij} = \alpha_i(\delta_{ij} - \alpha_j)$.*

(b) ***Column-wise ℓ_1 bound.*** *For any column j, summing the absolute values of entries,*

$$\sum_i |J_{ij}| = |\alpha_j(1-\alpha_j)| + \sum_{i \ne j} |-\alpha_i\alpha_j| = \alpha_j(1-\alpha_j) + \alpha_j \sum_{i \ne j} \alpha_i = 2\alpha_j(1-\alpha_j) \le \tfrac{1}{2},$$

since $\max_{x \in [0,1]} 2x(1-x) = 1/2$.

(c) ***Mean value theorem.*** *The induced operator norm satisfies $\|J(\mathbf{u})\|_{1\to 1} \le 1/2$ for all $\mathbf{u}$. Therefore,*

$$\|\boldsymbol{\alpha}(\mathbf{u}) - \boldsymbol{\alpha}(\mathbf{v})\|_1 \le \sup_{\xi} \|J(\xi)\|_{1\to 1} \|\mathbf{u} - \mathbf{v}\|_1 \le \tfrac{1}{2}\|\mathbf{u} - \mathbf{v}\|_1.$$

Eq. (24) follows by substituting $\mathbf{u} = \tau\mathbf{w}$ and $\mathbf{v} = \tau\mathbf{w}'$.

Proposition 1 (Stability under routing perturbations) *Under Assumption 1, for any two layer-score vectors $\mathbf{w}, \mathbf{w}' \in \mathbb{R}^L$ and fixed τ,*

$$\|\bar{\mathbf{z}}_{n,t}(\mathbf{w}) - \bar{\mathbf{z}}_{n,t}(\mathbf{w}')\|_2 \le \frac{B\tau}{2}\|\mathbf{w} - \mathbf{w}'\|_1. \tag{25}$$

If f_{proj} is L_{proj}-Lipschitz, then $\|\tilde{\mathbf{z}}_{n,t}(\mathbf{w}) - \tilde{\mathbf{z}}_{n,t}(\mathbf{w}')\|_2 \le \frac{B\tau L_{\mathrm{proj}}}{2}\|\mathbf{w} - \mathbf{w}'\|_1$.

Proof B.4 *By applying the triangle inequality with the norm bound B and Lemma B.3, we have:*

$$\|\bar{\mathbf{z}}(\mathbf{w}) - \bar{\mathbf{z}}(\mathbf{w}')\|_2 \le B\|\boldsymbol{\alpha}(\tau\mathbf{w}) - \boldsymbol{\alpha}(\tau\mathbf{w}')\|_1 \le \frac{B\tau}{2}\|\mathbf{w} - \mathbf{w}'\|_1.$$

The projector bound follows from Lipschitzness of f_{proj}, scaling the error by the constant.

Practical reading. Eq. (25) makes the role of τ explicit: larger τ yields sharper (more class-distinct) depth preferences (Lemma B.2), but also linearly amplifies sensitivity to score perturbations. This justifies conservative τ (or annealing) to effectively dampen the amplification of routing noise when router uncertainty is non-negligible.

Corollary 1 (Misrouting sensitivity) *Let c and c' be two classes with score vectors $\mathbf{w}_c$ and $\mathbf{w}_{c'}$. Under Assumption 1, the fused-token drift caused by misrouting $c \to c'$, which corresponds to the feature shift induced by the incorrect selection, satisfies*

$$\|\bar{\mathbf{z}}_{n,t}(\mathbf{w}_c) - \bar{\mathbf{z}}_{n,t}(\mathbf{w}_{c'})\|_2 \le \frac{B\tau}{2}\|\mathbf{w}_c - \mathbf{w}_{c'}\|_1. \tag{26}$$

Proof B.5 *Apply Proposition 1 by substituting the specific class vectors $\mathbf{w} = \mathbf{w}_c$ and $\mathbf{w}' = \mathbf{w}_{c'}$.*

B.3 Relevance–Coverage Optimization

We analyze one pruning stage with actual budget $B = B_s$, split into $K_1 = \lfloor a_n B \rfloor$ and $K_2 = B - K_1$, where $a_n \triangleq a_{c_n}$. Here B is a retained-token count, not the effective-budget label R for the full progressive schedule. Stage I selects *relevance pivots* using saliency scores $\phi_{n,t}$, and Stage II selects *diversity completion* tokens from the non-pivot pool $\mathcal{U}_n = \mathcal{V}_n \setminus \mathcal{P}_n$, designed to capture semantic information missed by the saliency-based selection.

Boundary conventions. Each stage uses an integer budget $0 \le B_s \le |\mathcal{V}_n|$. Selections return distinct token indices; ties are resolved by increasing token or cluster index, and zero-size selections return $\varnothing$. When $K_2 = 0$, set $\mathcal{Q}_n = \varnothing$ and skip Stage II. For similarity computations only, zero token features are mapped to the fixed unit vector $e_1 = (1, 0, \dots, 0)^\top$.

For $K_1 > 0$ and $K_2 > 0$, retain Bottom-K_2 seeding based on redundancy with respect to the fixed pivot set. When $K_1 = 0 < K_2$, use deterministic farthest-first initialization on $\mathcal{U}_n = \mathcal{V}_n$. Set $c_1 = \min \mathcal{U}_n$ and $\mathcal{C}_1 = \{c_1\}$. For $m = 2, \dots, K_2$, select

$$c_m = \arg \min_{t \in \mathcal{U}_n \setminus \mathcal{C}_{m-1}} \max_{j \in \mathcal{C}_{m-1}} u_{n,t}^\top u_{n,j}, \qquad \mathcal{C}_m = \mathcal{C}_{m-1} \cup \{c_m\}.$$

Set $\mathcal{C}_n^{(0)} = \mathcal{C}_{K_2}$ and $\mu_{n,m}^{(0)} = u_{n,c_m}$ for $m = 1, \dots, K_2$. This branch bypasses pivot-relative redundancy computation and Bottom-K_2 seeding.

For $K_2 > 0$ and integer $T \ge 1$, after each assignment step, fill empty clusters in increasing cluster-index order by moving the smallest-index eligible token from a cluster containing at least two tokens, setting the receiving center to that token's unit feature. Use the repaired assignments for center updates and medoid selection. If a cluster's feature sum is zero, retain its previous unit center. Since $|\mathcal{U}_n| \ge K_2$, repair always succeeds and yields K_2 nonempty, disjoint clusters, hence $|\mathcal{P}_n \cup \mathcal{Q}_n| = B_s$. Repair does not decrease the assigned cosine-similarity sum: the moved token attains similarity 1 and all other assigned similarities remain unchanged before the center update. A zero-sum cluster contributes zero for any unit center. The pivot-relative redundancy definitions and Proposition 2 apply to $K_1 > 0$. Clustering statements assume $K_2 > 0$; seed–pivot separation additionally assumes $K_1 > 0$.

Pruning equations in execution order. We restate the pruning pipeline as a numbered "recipe" for easier reference.

1. **Attention matrix and saliency (relevance score).** Given an attention block with query/key/value Q, K, V, define

$$\mathrm{Attn}(Q, K, V) = \mathrm{softmax}\Big(\frac{QK^\top}{\sqrt{d_k}}\Big)V, \qquad \mathbf{A}_n \triangleq \mathrm{softmax}\Big(\frac{QK^\top}{\sqrt{d_k}}\Big), \tag{27}$$

where $\mathbf{A}_n$ is the attention matrix (typically averaged over heads in practice). For a compact reference index set $\mathcal{S}$ (*e.g.*, visual [CLS] or selected text instruction tokens), we score a visual token $t \in \mathcal{V}_n$ by

$$\phi_{n,t} \triangleq \frac{1}{|\mathcal{S}|}\sum_{i \in \mathcal{S}} A_{n,i,t}, \qquad t \in \mathcal{V}_n. \tag{28}$$

Here $A_{n,i,t}$ measures how much the reference token i attends to visual token t. Larger $\phi_{n,t}$ indicates higher instruction relevance, effectively highlighting the visual regions most aligned with the user prompt.

2. **Budget split into relevance pivots vs. diversity completion.** Given a class-dependent ratio $a_n \in [0, 1]$ and actual stage budget B, the relevance and diversity allocations are:
$$K_1 = \lfloor a_n B \rfloor, \qquad K_2 = B - K_1. \tag{29}$$
3. **Stage I: pivots by top-K_1 saliency.** We keep the K_1 most salient tokens as pivots:
$$\mathcal{P}_n = \mathrm{Top}_{K_1}\big(\{\phi_{n,t}\}_{t\in\mathcal{V}_n}\big). \tag{30}$$
This stage is "relevance-first": it aggressively preserves tokens strongly queried by the instruction.
4. **Pivot-relative redundancy.** Let $\mathcal{U}_n = \mathcal{V}_n \setminus \mathcal{P}_n$ be the non-pivot pool. Using unit features $\mathbf{u}_{n,t}$ (Eq. (19)), define redundancy of a candidate token $t \in \mathcal{U}_n$ as
$$\rho_{n,t} \triangleq \max_{j\in\mathcal{P}_n} \mathbf{u}_{n,t}^\top \mathbf{u}_{n,j}, \qquad t \in \mathcal{U}_n. \tag{31}$$
Because $\mathbf{u}_{n,t}$ are unit vectors, $\rho_{n,t} \in [-1, 1]$ is cosine similarity. Large $\rho_{n,t}$ means t is highly duplicated by some pivot. Small $\rho_{n,t}$ means t lies in a direction poorly covered by pivots.
5. **Deterministic redundancy-aware seeding for Stage II.** We view $\rho_{n,t}$ as a *pivot-overlap cost*: it measures how much a candidate token $t \in \mathcal{U}_n$ resembles the already-selected pivot set $\mathcal{P}_n$ (worst-case cosine overlap). To make our seeding rule explicit, for any seed set $\mathcal{C} \subseteq \mathcal{U}_n$ with $|\mathcal{C}| = K_2$, we define the total pivot-relative redundancy as
$$D(\mathcal{C} \mid \mathcal{P}_n) \triangleq \sum_{t\in\mathcal{C}} \rho_{n,t}. \tag{32}$$
Since Eq. (32) is additive over tokens, the global minimizer under $|\mathcal{C}| = K_2$ is obtained by selecting the K_2 tokens with the smallest $\rho_{n,t}$, yielding a deterministic initialization that starts from directions complementary to $\mathcal{P}_n$. Note that D targets *seed-to-pivot* redundancy. Redundancy *among* completion tokens is then reduced by the subsequent K-means refinement.
6. **Bottom-K_2 seeding (optimal for D).** The minimizer of Eq. (32) is obtained by selecting the K_2 least redundant tokens, since the sum is minimized by the smallest individual terms:
$$\mathcal{C}_n^{(0)} = \mathrm{Bottom}_{K_2}\big(\{\rho_{n,t}\}_{t\in\mathcal{U}_n}\big). \tag{33}$$
We use the corresponding unit features as initial spherical K-means centers, *i.e.*, set $\boldsymbol{\mu}_{n,k}^{(0)} = \mathbf{u}_{n,c_k}$ for $c_k \in \mathcal{C}_n^{(0)}$.
7. **Spherical K-means refinement (coverage).** With cosine similarity, spherical K-means iterates between:
   (a) **Assignment (nearest-center by cosine).**
$$s^{(r)}(t) = \arg\max_{k\in\{1,\dots,K_2\}} \mathbf{u}_{n,t}^\top \boldsymbol{\mu}_{n,k}^{(r-1)}, \qquad t \in \mathcal{U}_n. \tag{34}$$
   (b) **Update (normalized mean direction).**
$$\boldsymbol{\mu}_{n,k}^{(r)} = \frac{\sum_{t:s^{(r)}(t)=k} \mathbf{u}_{n,t}}{\left\|\sum_{t:s^{(r)}(t)=k} \mathbf{u}_{n,t}\right\|_2}, \qquad k \in \{1, \dots, K_2\}. \tag{35}$$
We run $r = 1, \dots, T$ iterations, where small T (*e.g.*, 5) is typically sufficient in practice.
8. **Discrete completion tokens via cluster medoids.** After T iterations, we select one representative per cluster:
$$q_{n,k} = \arg\max_{t:s^{(T)}(t)=k} \mathbf{u}_{n,t}^\top \boldsymbol{\mu}_{n,k}^{(T)}, \qquad \mathcal{Q}_n = \{q_{n,k}\}_{k=1}^{K_2} \subseteq \mathcal{U}_n. \tag{36}$$
Eq. (36) returns *token indices* (not continuous centers), so the retained set is directly usable by the decoder.

Proposition 2 (Optimal redundancy-aware seeding) *Fix $\mathcal{P}_n$ and let $\mathcal{U}_n = \mathcal{V}_n \setminus \mathcal{P}_n$. Among all seed sets $\mathcal{C} \subseteq \mathcal{U}_n$ with $|\mathcal{C}| = K_2$, the minimizer of $D(\mathcal{C} \mid \mathcal{P}_n)$ in Eq. (32) is $\mathcal{C}_n^{(0)}$ in Eq. (33), providing the global optimum for the additive redundancy cost function.*

Proof B.6 *Eq. (32) is a sum of independent scalar costs $\rho_{n,t}$ over the chosen indices. Thus the minimum over all K_2-subsets is achieved by selecting the K_2 smallest $\rho_{n,t}$ values, i.e., Eq. (33).*

A coverage objective for spherical K-means. Define the directional coherence (coverage) objective

$$\mathcal{J}(\{\boldsymbol{\mu}_{n,k}\}_{k=1}^{K_2}) \triangleq \sum_{t\in\mathcal{U}_n} \max_{k\in\{1,\dots,K_2\}} \mathbf{u}_{n,t}^\top \boldsymbol{\mu}_{n,k}. \tag{37}$$

Each summand in Eq. (37) rewards a token by its cosine similarity to the nearest center. Maximizing $\mathcal{J}$ therefore encourages the centers to cover multiple modes in $\mathcal{U}_n$ rather than collapsing to a single region.

Proposition 3 (Monotonicity of spherical K-means) *The alternating updates in Eqs.* (34)–(35) *do not decrease $\mathcal{J}$ in Eq.* (37)*. Moreover, since $\mathcal{J}$ is bounded above, the procedure converges to a stationary point (a local optimum) of $\mathcal{J}$.*

Proof B.7 *We verify monotonicity in the same two-step order as the algorithm.*

- *(a)* ***Assignment improves $\mathcal{J}$ for fixed centers.*** *For fixed $\{\boldsymbol{\mu}_{n,k}\}$, choosing $s^{(r)}(t) = \arg\max_k \mathbf{u}_{n,t}^\top \boldsymbol{\mu}_{n,k}$ maximizes each summand in Eq.* (37)*, guaranteeing a non-decreasing objective value, hence cannot decrease $\mathcal{J}$.*
- *(b)* ***Update improves $\mathcal{J}$ for fixed assignments.*** *For fixed assignments, the contribution of cluster k equals $\sum_{t:s^{(r)}(t)=k} \mathbf{u}_{n,t}^\top \boldsymbol{\mu}_{n,k}$ subject to $\|\boldsymbol{\mu}_{n,k}\|_2 = 1$. By Cauchy–Schwarz, this is maximized by setting $\boldsymbol{\mu}_{n,k}$ proportional to $\sum_{t:s^{(r)}(t)=k} \mathbf{u}_{n,t}$ and normalizing, yielding the optimal directional center, i.e., Eq.* (35)*.*

Since each iteration is non-decreasing and $\mathcal{J} \le |\mathcal{U}_n|$ (each term ≤ 1), monotone ascent implies convergence.

B.4 Geometric Diversity Guarantees

Lemma B.4 (Cosine similarity and Euclidean distance) *For any unit vectors $\mathbf{u}, \mathbf{v}$, we directly have $\|\mathbf{u} - \mathbf{v}\|_2^2 = 2(1 - \mathbf{u}^\top \mathbf{v})$.*

Proof B.8 *Expand $\|\mathbf{u} - \mathbf{v}\|_2^2 = \|\mathbf{u}\|_2^2 + \|\mathbf{v}\|_2^2 - 2\mathbf{u}^\top \mathbf{v} = 2 - 2\mathbf{u}^\top \mathbf{v}$.*

Corollary 2 (Seed–pivot separation) *Let $\mathcal{C}_n^{(0)}$ be the seed set in Eq.* (33)*. Define the (deterministic) redundancy threshold*

$$\delta_n \triangleq \max_{t\in\mathcal{C}_n^{(0)}} \rho_{n,t}, \tag{38}$$

which is equivalently the K_2-th smallest value among $\{\rho_{n,t}\}_{t\in\mathcal{U}_n}$. Then for every seed $\mathbf{c} \in \{\mathbf{u}_{n,t} : t \in \mathcal{C}_n^{(0)}\}$, which are chosen to minimize the redundancy with the pre-selected pivots, and any pivot $j \in \mathcal{P}_n$,

$$\mathbf{c}^\top \mathbf{u}_{n,j} \le \delta_n \qquad \textit{and} \qquad \|\mathbf{c} - \mathbf{u}_{n,j}\|_2 \ge \sqrt{2(1-\delta_n)}. \tag{39}$$

Proof B.9 *By construction of $\mathcal{C}_n^{(0)}$, each selected seed index $t \in \mathcal{C}_n^{(0)}$ satisfies $\rho_{n,t} \le \delta_n$. Since $\rho_{n,t} = \max_{j\in\mathcal{P}_n} \mathbf{u}_{n,t}^\top \mathbf{u}_{n,j}$, we have $\mathbf{u}_{n,t}^\top \mathbf{u}_{n,j} \le \rho_{n,t} \le \delta_n$ for all pivots j. Applying Lemma B.4 yields the Euclidean bound.*

Interpretation. Eq. (39) formalizes what redundancy-aware seeding enforces geometrically: initial centers lie outside a cosine-similarity "cap" around each pivot, guaranteeing a minimum Euclidean distance to the pivot set. Spherical K-means then refines these centers to better represent modes of $\mathcal{U}_n$ without relying on random initialization.

B.5 Computational complexity

For a single pruning stage, let $M_n = |\mathcal{V}_n|$ be the current candidate count, d the embedding dimension, and $B = B_s$ the actual stage budget, with $K_1 = \lfloor a_n B \rfloor$ and $K_2 = B - K_1$. We summarize the additional overhead incurred by pruning beyond a standard forward pass in the same step-wise style.

1. **Layer fusion (Eq. (21)).** Computing $\{\bar{\mathbf{z}}_{n,t}\}_{t\in\mathcal{V}_n}$ costs $O(L\,M_n\,d_v)$: a single weighted sum over cached layer outputs.
2. **Saliency top-K_1 (Eq. (30)).** Selecting Top_{K_1} costs $O(M_n \log K_1)$ via partial sort / heap.
3. **Redundancy computation for seeding (Eq. (31)).** Computing the redundancy scores for all candidates in $\mathcal{U}_n$ requires pairwise similarities with the K_1 pivots and costs $O((M_n - K_1)K_1 d)$. This can be implemented as a matrix multiplication between $\mathbf{U}_{\mathcal{U}} \in \mathbb{R}^{(M_n-K_1)\times d}$ and $\mathbf{U}_{\mathcal{P}} \in \mathbb{R}^{K_1\times d}$ followed by a row-wise max, yielding the required redundancy values.
4. **Bottom-K_2 seeding (Eq. (33)).** Selecting Bottom_{K_2} costs $O((M_n - K_1)\log K_2)$.
5. **Spherical K-means refinement (Eqs. (34)–(35)).** Each iteration costs $O((M_n - K_1)K_2 d)$ for similarity evaluation/assignment plus $O((M_n - K_1)d)$ for accumulating cluster sums and normalization. Over T iterations, the refinement cost is $O(T(M_n - K_1)K_2 d)$, which remains efficient for small iteration counts.
6. **Medoid selection (Eq. (36)).** Computing similarities of each token to its final cluster center costs $O((M_n - K_1)d)$.

Memory overhead is dominated by storing normalized token matrices, *e.g.*, $O((M_n - K_1)d)$ if $\mathbf{U}_{\mathcal{U}}$ is materialized. Since $K_2 \le B$ and T is fixed (*e.g.*, $T = 5$), refinement is limited by the actual stage budget. For progressive pruning, the stage-specific costs are summed over the pruning layers. The timing measurements in in Table 5 and Table 23 include category preprocessing, feature fusion, token selection, clustering, and MLLM execution. Reported memory is the peak GPU memory measured over the complete two-pass workflow.

C Detailed Experiment Settings

Benchmarks and Metrics. We conduct experiments on a comprehensive suite of benchmarks to evaluate our model's multimodal capabilities across both static and dynamic scenarios. For **image understanding**, we assess general perception, compositional reasoning, OCR, and hallucination using eight widely used datasets in our main evaluation: GQA (Hudson & Manning, 2019), MMBench (MMB) (Liu et al., 2024d), MME (Fu et al., 2026), POPE (Li et al., 2023c), ScienceQA (SQA) (Lu et al., 2022), VQAv2 (Goyal et al., 2017), TextVQA (Singh et al., 2019), and MMVet (Yu et al., 2023). Additionally, to further demonstrate the robustness of our approach, we provide supplementary evaluations on LLaVA-Bench (LLaVA-B) (Liu et al., 2023), VizWiz (Gurari et al., 2018), MMBench-Chinese (MMB-CN) (Liu et al., 2024d), and SEED-Bench (SEED) (Li et al., 2023a) in the appendix. For video understanding, we evaluate question answering on TGIF-QA (Jang et al., 2017), MSVD-QA, and MSRVTT-QA (Xu et al., 2017). These names refer to the QA benchmarks rather than the captioning tasks of their source datasets. Unless otherwise specified, we report accuracy following the official evaluation protocols to ensure a fair comparison.

GQA. GQA (Hudson & Manning, 2019) evaluates scene understanding and compositional reasoning. It is built upon images, structured scene graphs, and questions designed to probe fine-grained object attributes, spatial relations, and multi-step reasoning. Generated from the Visual Genome dataset, GQA contains over 22 million questions across approximately 113K images. A key characteristic of GQA is its rigorous balancing of the answer distribution for each question group, which significantly mitigates language priors and forces models to rely on visual evidence rather than statistical correlations. In the context of model compression, GQA provides a diagnostic view of a model's ability to jointly perceive and reason over complex visual scenes, serving as a stress test for preserving multi-hop reasoning capabilities after pruning.

MMBench. MMBench (Liu et al., 2024d) provides a multi-dimensional evaluation framework organized in a three-level hierarchy of capabilities. Level-1 targets the two core abilities of perception and reasoning; Level-2 expands these into six sub-abilities; and Level-3 further refines the assessment into 20 fine-grained ability dimensions. Constructed from various sources, MMBench contains approximately 3,000 multiple-choice questions covering diverse domains. A distinguishing feature of MMBench is its CircularEval strategy, which inputs the same question with shifted options multiple times to the model. This mechanism effectively mitigates the model's sensitivity to option ordering and random guessing. For VLM pruning, MMBench serves as a critical benchmark to verify that the

compressed model maintains robust instruction-following and reasoning capabilities across a broad spectrum of tasks, rather than overfitting to specific patterns.

MME. MME (Fu et al., 2026) is a comprehensive and quantitative benchmark designed to evaluate multimodal LLMs across 14 distinct subtasks. These subtasks are structurally categorized into two primary axes: Perception (*e.g.*, coarse-grained recognition, OCR, color) and Cognition (*e.g.*, commonsense reasoning, numerical calculation). MME employs concise instruction–answer pairs, predominantly in a "Yes/No" format, to minimize the influence of prompt engineering and reduce potential data leakage. With manually annotated samples, it facilitates a reliable measurement of multimodal performance. For pruning studies, MME is particularly useful for diagnosing whether parameter reduction disproportionately impacts basic visual perception or higher-order cognitive reasoning.

POPE. POPE (Li et al., 2023c) is a specialized benchmark focusing on evaluating object hallucination in VLMs. It reformulates hallucination assessment as a series of binary "Yes/No" questions about the presence of specific objects in an image (based on MSCOCO validation sets). Crucially, POPE evaluates models under three distinct sampling settings to disentangle visual perception from language priors: Random (objects not in the image), Popular (frequent objects in the dataset), and Adversarial (objects that often co-occur with present objects but are absent). We use the COCO-POPE annotations referenced by LLaVA (revision `e3e3926`), containing $2{,}910$ random, $3{,}000$ popular, and $3{,}000$ adversarial questions ($8{,}910$ in total). For the main performance tables, we report the arithmetic mean of the F1 scores (%) across these three settings.

ScienceQA. ScienceQA (Lu et al., 2022) spans a wide array of domains, including natural, social, and language sciences. Questions are hierarchically categorized into 26 topics, 127 categories, and 379 skills, providing a diverse and comprehensive testbed for evaluating multimodal understanding and multi-step reasoning. Distinctively, ScienceQA includes annotated lectures and explanations, facilitating the assessment of Chain-of-Thought (CoT) capabilities. All reported SQA scores are accuracy (%) on image-containing questions (SQA-IMG). The $4{,}241$ ScienceQA questions in Table 7 are counted for taxonomy statistics, not as the denominator of SQA-IMG accuracy.

VQAv2. VQAv2 (Goyal et al., 2017) is a standard open-ended VQA benchmark covering diverse real-world images and questions derived from the MS COCO dataset. A critical feature of VQAv2 is its design to minimize language bias through balanced pairs: for every question, the dataset includes complementary images that result in different answers (*e.g.*, "Yes" vs. "No"). This structure compels the model to rely on visual evidence rather than exploiting statistical language correlations. With over 1.1 million questions and multiple human annotations per question, VQAv2 serves as a large-scale testbed for general visual question answering. For pruning, it provides a fundamental baseline to ensure that the compressed model retains generalized visual recognition capabilities and aligns visual features correctly with textual queries.

TextVQA. TextVQA (Singh et al., 2019) evaluates reasoning over embedded text in images, addressing a specific capability often overlooked by general VQA benchmarks. Built upon images from the Open Images dataset, it comprises 45,336 questions that require the model to detect, recognize, and reason about text appearing in diverse scenes (*e.g.*, signboards, book covers). Successful performance demands a tight integration of optical character recognition (OCR) and semantic reasoning, as the model must often transcribe specific text strings from the image to formulate the answer. For VLM pruning, TextVQA serves as a rigorous test of fine-grained feature preservation, verifying that the compressed model retains sufficient resolution and local attention to process small, symbol-rich visual elements without degradation.

MM-Vet. MM-Vet (Yu et al., 2023) targets complex multimodal problem solving by defining six core vision–language capabilities: Recognition, Knowledge, OCR, Spatial Awareness, Language Generation, and Math. Beyond testing these in isolation, MM-Vet evaluates 16 distinct integrations of these capabilities (*e.g.*, recognizing an object and then answering a knowledge-based question about it). The benchmark employs an LLM-based evaluation system (typically GPT-4) to score open-ended responses, offering a nuance that rigid string-matching metrics lack. For token pruning, MM-Vet

assesses whether visual-token reduction preserves the integrated multimodal capabilities required for multi-step reasoning.

LLaVA-Bench. LLaVA-Bench (Liu et al., 2023) is designed to evaluate the capability of MLLMs in handling complex, open-ended visual instructions in real-world scenarios. Comprising a diverse set of "in-the-wild" images and detailed queries, it relies on GPT-4 as an impartial judge to score the model's generated responses against reference answers. Unlike standard objective benchmarks, LLaVA-Bench captures the nuances of conversational fluency, helpfulness, and visual grounding. In the context of token pruning, it is crucial for verifying that aggressive visual compression does not degrade the model's capacity for rich, free-form multimodal generation and its adherence to unpredictable human instructions.

VizWiz. VizWiz (Gurari et al., 2018) originates from authentic visual questions asked by blind and visually impaired individuals. Consequently, the dataset is characterized by images with severe real-world noise, such as poor lighting, blurriness, and occlusions, coupled with occasionally unanswerable queries. This benchmark rigorously tests a model's robustness and zero-shot generalization under suboptimal visual conditions. For VLM pruning, VizWiz serves as a unique diagnostic tool: it assesses whether the pruning algorithm remains robust when visual cues are inherently scarce or distorted, ensuring that the token reduction process does not disproportionately discard critical, albeit noisy, visual information.

MMBench-Chinese. MMBench-Chinese (MMB-CN) (Liu et al., 2024d) is the bilingual extension of the MMBench evaluation framework, translating the rigorous, multi-dimensional assessment into the Chinese linguistic context. Retaining the CircularEval strategy to mitigate option-selection bias, it evaluates the same broad spectrum of perception and reasoning capabilities but fundamentally requires robust cross-lingual alignment. Within our pruning study, MMB-CN is utilized to confirm that the class-adaptive layer fusion and token reduction mechanisms generalize effectively across diverse linguistic spaces, ensuring that cross-modal semantic alignment is not structurally impaired when evaluated in non-English languages.

SEED-Bench. SEED-Bench (Li et al., 2023a) is a comprehensive multimodal benchmark specifically designed to evaluate fine-grained visual understanding and spatial reasoning. It comprises thousands of multiple-choice questions with meticulously crafted distractors, targeting specific dimensions such as object attributes, instance locations, and spatial relations. The precise nature of these queries forces models to rely on exact visual evidence rather than holistic semantic guessing. In the evaluation of pruned MLLMs, SEED-Bench acts as a stringent stress test for fine-grained token retention; it reveals whether our dual-stage pruning strategy successfully preserves the crucial, highly localized visual tokens required to distinguish between subtle, misleading options.

TGIF-QA. TGIF-QA (Jang et al., 2017) extends animated GIF data with question–answer annotations. We use the question-answering task to evaluate retained visual evidence for answering questions about GIF content, rather than generating GIF descriptions. The TGIF-QA column in the video results reports QA performance.

MSVD-QA. MSVD-QA (Xu et al., 2017) provides question–answer annotations over videos from MSVD (Chen & Dolan, 2011). Our evaluation measures video question-answering accuracy, not video-caption quality. This benchmark tests whether visual-token reduction preserves evidence needed to answer questions about video content.

MSRVTT-QA. MSRVTT-QA (Xu et al., 2017) is the question-answering benchmark derived from MSR-VTT (Xu et al., 2016). We report QA accuracy on questions about the videos, rather than captioning metrics. Together with TGIF-QA and MSVD-QA, it evaluates whether token pruning preserves information useful for video question answering.

Models. We instantiate and evaluate CLASP on top of several representative *open-source* multimodal large language models (MLLMs). For **image understanding**, we primarily build on the LLaVA family, including the standard LLaVA-1.5 (Liu et al., 2024a) and the LLaVA-NeXT (Liu et al., 2024b). We specifically adopt LLaVA-NeXT to validate performance under **high-resolution** visual

inputs, following the official inference settings. To demonstrate the generalization of our method across different architectures, we further include Qwen2.5-VL (Bai et al., 2025). As a state-of-the-art model, it introduces Naive Dynamic Resolution mechanisms to handle arbitrary aspect ratios, serving as a rigorous testbed for our pruning strategy on variable-length visual tokens. Finally, to assess capabilities in the **temporal domain**, we extend our evaluation to Video-LLaVA (Lin et al., 2024). This model unifies image and video feature alignment, allowing us to verify whether CLASP can effectively reduce temporal redundancies without compromising motion understanding.

ToMe. ToMe (Bolya et al., 2022) accelerates vision transformers by merging similar tokens within transformer layers using lightweight token matching. Unlike traditional pruning methods that discard "unimportant" tokens, ToMe employs a bipartite matching algorithm based on feature similarity (typically using the Key or Query projections) to identify redundant tokens. It then aggregates these tokens via weighted averaging, progressively reducing the sequence length across the network depth. Since it requires no additional parameters or retraining, ToMe serves as a highly efficient, plug-and-play baseline for evaluating the trade-off between inference speed and information preservation in the VLM's visual encoder.

LLaVA-PruMerge. LLaVA-PruMerge (Shang et al., 2025) introduces an adaptive hybrid strategy that combines the benefits of both pruning and merging. It first evaluates the importance of visual tokens based on the attention scores from the special [CLS] token to image patches, effectively identifying regions most relevant to the global semantic context. Based on these scores, the method divides tokens into subsets: tokens with low importance scores are directly pruned to remove background noise, while highly correlated foreground tokens are merged based on key similarity. This approach significantly reduces the sequence length of the visual encoder without retraining, ensuring that critical visual details are preserved while spatial redundancy is efficiently eliminated before the tokens enter the LLM.

FastV. FastV (Chen et al., 2024a) identifies the inefficiency of visual tokens within Large Language Models (LLMs) and performs early-stage token pruning. It is grounded in the observation that while visual tokens are crucial in the initial transformer layers, the attention mechanism in deeper layers tends to ignore the vast majority of them. Leveraging this, FastV ranks visual tokens based on their average attention weights in the early layers (*e.g.*, the second layer) and discards the least significant ones for all subsequent layers. This strategy significantly reduces the KV-cache memory footprint and FLOPs during inference, allowing the model to maintain high performance while processing significantly fewer tokens in the computation-heavy deep layers.

HiRED. HiRED (Arif et al., 2025) introduces a spatially-aware pruning strategy designed to mitigate the "tunnel vision" problem often observed in global ranking methods. Instead of competing all visual tokens in a single pool, HiRED divides the image into spatial partitions. It then dynamically allocates a specific token budget to each partition based on the attention distribution of the [CLS] token. By selecting the most informative tokens within these localized budgets, HiRED ensures that the pruned model maintains broad spatial coverage across the image. This hierarchical approach allows for aggressive compression while preventing the complete suppression of background regions or secondary objects, which are essential for tasks requiring holistic scene understanding.

PDrop. PDrop (Xing et al., 2024), or PyramidDrop, implements a progressive pruning strategy that mimics the hierarchical structure of CNNs within a standard Vision Transformer. Instead of maintaining a constant sequence length throughout the encoder, PDrop drastically reduces the token count at specific intermediate layers based on attention importance scores. This constructs a "pyramid-like" information flow, where high-resolution details are processed in shallow layers, while only the most semantically salient tokens are retained for the computation-heavy deep layers. By effectively funneling visual information, PDrop achieves a significant reduction in FLOPs and latency, serving as a representative baseline for structural pruning methods.

Multi-Stage Vision Token Dropping (MustDrop). Multi-Stage Vision Token Dropping (Liu et al., 2024c) adopts a progressive compression strategy within the vision encoder to optimize computational efficiency. Instead of performing a single-step reduction, it executes token dropping at multiple intermediate layers (stages) based on attention significance or learned policies. This

hierarchical approach allows the model to retain fine-grained low-level details in the shallow stages while aggressively reducing spatial redundancy in the deeper, more semantic layers. Consequently, it significantly lowers the FLOPs of the vision backbone and minimizes the sequence length passed to the multimodal projector, achieving a favorable balance between inference speed and task performance.

SparseVLM. SparseVLM (Zhang et al., 2024b) presents a dynamic pruning framework that goes beyond uni-modal visual salience. It ranks token importance using cross-modal attention, ensuring that visual tokens are preserved based on their relevance to the specific textual query or instruction. Recognizing that different images contain varying amounts of information, SparseVLM introduces adaptive sparsity ratios, dynamically adjusting the token budget for each input. Furthermore, to mitigate the risks of aggressive pruning, it proposes a novel token recycling mechanism. Instead of permanently discarding pruned tokens, this mechanism aggregates or buffers them, allowing the model to retrieve context from these "discarded" regions if necessary. This significantly improves efficiency under varying input complexity without compromising the semantic integrity of the visual representation.

VisionZip. VisionZip (Yang et al., 2025) addresses the spatial redundancy problem often found in attention-based pruning methods. While standard approaches simply select the top-k tokens with the highest attention scores, VisionZip observes that these high-scoring tokens tend to cluster around the same visual object, leading to repetitive information. To mitigate this, it implements a two-step pipeline: first, it evaluates token saliency via encoder attention to filter out background noise; second, it clusters the remaining high-saliency tokens based on key similarity. By selecting representative tokens from these clusters, VisionZip ensures that the final compressed sequence maintains high semantic diversity, covering various distinct objects and regions within the image for downstream multimodal reasoning.

DART. DART (Wen et al., 2025b) proposes a Duplication-Aware Reduction of Tokens strategy designed to maximize information diversity while maintaining hardware efficiency. Unlike magnitude-based methods that may select redundant high-norm tokens, DART explicitly filters out redundancy. It operates by selecting a small set of "pivot" tokens and computing the cosine similarity between these pivots and the remaining tokens. Tokens with high similarity (high duplication) are discarded, ensuring that the retained set covers a broad semantic range. Crucially, DART is designed to be compatible with FlashAttention and other efficient attention operators. By avoiding complex gather-scatter operations or irregular memory access patterns, it ensures that the theoretical reduction in FLOPs translates directly into significant wall-clock speedups during inference.

Implementation details. For image-based benchmarks, we run experiments using the official LLaVA implementation. For video question answering, we evaluate Video-LLaVA (Lin et al., 2024) via `lmms-eval`. For more recent VLM architectures (*e.g.*, Qwen2.5-VL), we rely on `VLMEvalKit` as the evaluation toolkit. All inference evaluations were executed on NVIDIA A100 (80GB) GPUs. To ensure reproducibility, we detail the specific configurations for dataset construction, parameter validation, and model-specific architectures below.

Dataset Categorization and Intent Mapping. To enable class-adaptive routing, we constructed a categorized dataset using the Qwen3-8B model to identify visual question answering intents. We designed a system prompt defining 15 fine-grained intents (*e.g.*, *arithmetic reasoning*, *temporal order*) and instructed the model to output classifications in a strict JSON format. To enhance robustness, we applied deterministic heuristic overrides (*e.g.*, mapping keywords like "who wrote" to *Text/Symbol Recognition*). These labels were merged into the 9 core categories (Table 6) to ensure sufficient sample density for routing.

Hyperparameter Search and Optimization. We select fusion configurations and pruning ratios using the discrete subspace search in Appendix A.2. The default calibration set contains 18,000 labeled samples drawn from the released LLaVA-v1.5 instruction-tuning mixture, `llava_v1_5_mix665k.json`, with 2,000 samples for each of the nine categories. Reference outputs are inherited from the source annotations, while category labels follow the intent-mapping procedure described above. We rely on the original LLaVA-v1.5 training/evaluation data separation, rather than constructing the calibration set from benchmark evaluation splits. The intent taxonomy, mapping rules, and candidate prototypes are predefined; the category-wise fusion configuration and split ratio

are selected by evaluating candidates against reference outputs in the corresponding category subsets of this 18,000-sample calibration set, rather than by averaging the reported benchmark scores. The vision encoder, projector, and language decoder remain frozen throughout. Except for ablations that explicitly vary the calibration size or configuration, results use the selected settings in Table 6. For each model, the calibrated configurations are shared across benchmarks without benchmark-specific recalibration and reused during inference without updating the backbone parameters.

Model-Specific Configurations. We devised distinct fusion strategies tailored to the architecture of the vision encoders. The LLaVA family (LLaVA-v1.5, LLaVA-NeXT, and Video-LLaVA) shares a unified set of fusion parameters, while Qwen2.5-VL employs a separate configuration adapted to its layer structure. The split ratios (a_c) for the two-stage pruning were kept consistent across all models.

Pruning Schedule and Budgets. We adopt the progressive sparsification strategy from SparseVLM, executing token pruning at three designated intermediate layers of the multimodal decoder on the **projected (decoder-space) visual tokens**: **Layer 2, Layer 6, and Layer 15**. Following the progressive-budget convention used for comparison with SparseVLM (Zhang et al., 2024b), R specifies the target layer-averaged visual-token budget. For a decoder with D layers, the realized average is

$$\bar{R} = \frac{1}{D} \sum_{\ell=1}^{D} N_\ell,$$

where N_ℓ is the visual-token count at decoder layer ℓ during prefill. The sum covers all decoder layers, including those processing the full visual sequence before the first pruning operation. The actual stage budgets are $[B_1, B_2, B_3]$, and the final retained-token count is B_3.In our paired comparisons with SparseVLM on the same backbone and at the same budget setting, we match the visual-token count at every decoder layer:

$$N_\ell^{\mathrm{CLASP}} = N_\ell^{\mathrm{SparseVLM}}, \qquad \ell = 1, \ldots, D.$$

Thus, both the realized layer-averaged count and the complete layer-wise token-count profile are identical. CLASP adapts visual representations and token selection within this shared schedule, enabling comparison under the same layer-wise visual-token allocation. Complementary runtime comparisons and component measurements are reported in Table 5 and Table 23, respectively. Taking the standard input resolution ($N = 576$ projected visual tokens) as a baseline, the stepwise retention counts are configured as follows: (1) $R = 192$ uses $[B_1, B_2, B_3] = [300, 200, 110]$, ending with 110 tokens; (2) $R = 128$ uses $[303, 110, 36]$, ending with 36 tokens; (3) $R = 64$ uses $[66, 30, 17]$, ending with 17 tokens. For high-resolution architectures (*e.g.*, LLaVA-NeXT) processing $N = 2880$ tokens, these budgets are scaled proportionally. Unless stated otherwise, reported token-reduction percentages are nominal effective-budget reductions $1 - R/N$, not the final-stage fraction $1 - B_3/N$ or a measured FLOPs reduction. We use η for a retention fraction specified directly as a percentage, keeping it distinct from the count-valued budget R.

Relation to SparseVLM. For fair efficiency comparisons, we follow SparseVLM's (Zhang et al., 2024b) progressive three-stage pruning schedule in the multimodal decoder (Layer 2/6/15), while differing in how the prompt conditions the visual representation and how the token budget is allocated. SparseVLM ranks visual tokens using query-conditioned cross-modal attention and further mitigates aggressive sparsification via token recycling and adaptive sparsity ratios (Zhang et al., 2024b). In contrast, we adopt a text-only router and use its predicted category to drive two class-adaptive components: (i) *class-adaptive multi-layer feature fusion*, where token representations are formed by a temperature-controlled mixture over vision layers to match the level of visual abstraction required by the prompt; and (ii) *attention–similarity adaptive-ratio pruning*, where the actual stage budget B_s is split into $K_{1,s} = \lfloor a_c B_s \rfloor$ attention-salient pivots (relevance) and $K_{2,s} = B_s - K_{1,s}$ completion tokens (coverage). This formulation explicitly combines relevance-driven attention selection (*e.g.*, FastV (Chen et al., 2024a), HiRED (Arif et al., 2025), PyramidDrop (Xing et al., 2024)) with redundancy-aware retention strategies (*e.g.*, ToMe (Bolya et al., 2022), VisionZip (Yang et al., 2025), DART (Wen et al., 2025b), DivPrune (Alvar et al., 2025), conditional-diversity pruning (Zhang et al., 2026)), while keeping the backbone frozen. The timing measurements in Table 5 and Table 23 include category preprocessing, feature fusion, token selection, clustering, and MLLM execution. Reported memory is the peak GPU memory measured over the complete two-pass workflow.

Table 6: Hyperparameters for class-adaptive layer fusion and pruning across MLLM backbones. **Fusion weights** are reported as {layer: coefficient}. The **split ratio** a_c governs the prompt-conditioned budget allocation between attention-salient pivots for relevance and redundancy-aware completion tokens for coverage.

ID	Category	Fusion weights (LLaVA family)	Fusion weights (Qwen2.5-VL)	Split ratio a_c
0	Object identification	$\{L_5{:}0.2,\ L_{15}{:}0.3,\ L_{22}{:}0.5\}$	$\{L_9{:}0.2,\ L_{22}{:}0.3,\ L_{31}{:}0.5\}$	0.8
1	Attribute / breed identification	$\{L_5{:}0.2,\ L_{22}{:}0.8\}$	$\{L_9{:}0.2,\ L_{31}{:}0.8\}$	0.4
2	Text / symbol recognition	$\{L_5{:}0.2,\ L_{22}{:}0.8\}$	$\{L_9{:}0.2,\ L_{31}{:}0.8\}$	0.7
3	Scene understanding	$\{L_{20}{:}0.2,\ L_{22}{:}0.8\}$	$\{L_{28}{:}0.2,\ L_{31}{:}0.8\}$	0.7
4	Spatial relations	$\{L_{14}{:}0.2,\ L_{17}{:}0.3,\ L_{22}{:}0.5\}$	$\{L_{21}{:}0.2,\ L_{24}{:}0.3,\ L_{31}{:}0.5\}$	0.7
5	Counting	$\{L_5{:}0.2,\ L_{15}{:}0.3,\ L_{22}{:}0.5\}$	$\{L_9{:}0.2,\ L_{22}{:}0.3,\ L_{31}{:}0.5\}$	0.6
6	Action / interaction	$\{L_{12}{:}0.2,\ L_{15}{:}0.3,\ L_{19}{:}0.5\}$	$\{L_{18}{:}0.2,\ L_{22}{:}0.3,\ L_{28}{:}0.5\}$	0.8
7	Intention / function	$\{L_3{:}0.2,\ L_{12}{:}0.3,\ L_{18}{:}0.5\}$	$\{L_6{:}0.2,\ L_{18}{:}0.3,\ L_{26}{:}0.5\}$	0.2
8	Default	$\{L_{20}{:}0.2,\ L_{22}{:}0.8\}$	$\{L_{29}{:}0.2,\ L_{31}{:}0.8\}$	0.9

Table 7: Detailed statistical distribution of sample counts per task category across all evaluated multimodal benchmarks.

ID	Category	GQA	MMB	MME	MMVet	POPE	SQA	VQA_{v2}	VQA_{Text}
0	Object identification	2095	272	355	21	2277	178	18263	239
1	Attribute / breed identification	4015	268	85	19	0	410	19380	1037
2	Text / symbol recognition	75	111	147	12	0	465	4368	2856
3	Scene understanding	2319	1950	1209	58	6633	521	35553	388
4	Spatial relations	3632	379	70	18	0	193	7378	39
5	Counting	9	100	87	5	0	54	10007	53
6	Action / interaction	200	74	2	4	0	1	5850	8
7	Intention / function	43	287	79	9	0	8	3440	121
8	Default	190	936	340	72	0	2411	3155	259

D Additional Experiments

Throughout these supplementary experiments, count-valued budgets R refer to effective-budget settings, not final retained token counts. Actual stage budgets are denoted by B_s (Appendix C). Percentage-valued retention settings are denoted by η.

D.1 Calibration and Routing Evaluations

Impact of Calibration Set Size and Distribution. To rigorously justify our choices regarding the calibration dataset ($\mathcal{D}_{\text{cal}}$), we investigate how the size of the calibration set and its categorical distribution (sampling strategy) affect the final pruned model's performance. We evaluate two data selection strategies: a *Balanced* distribution (enforcing an equal number of samples per intent category) and a *Random* distribution (naturally imbalanced based on the source dataset's available distribution). We test total calibration sizes of 4.5k, 9k, and 18k samples. For each calibration size and sampling strategy, we draw five calibration sets using seeds $\{42, 43, 44, 45, 46\}$ and rerun the offline search for $(\mathbf{W}, \mathbf{a})$ on each set. The resulting parameters remain fixed during each evaluation, and we report the mean and standard deviation across the five runs. Clustering initialization remains deterministic given the token features and pivot set.

Table 8 shows that performance largely saturates around 9k samples, with small changes when the size increases to 18k. In the 4.5k setting, balanced sampling also yields lower reported variation than random sampling (*e.g.*, ± 6 versus ± 15 on MME and ± 0.3 versus ± 0.6 on TextVQA). Compared with the balanced 9k setting, the balanced 18k configuration maintains comparable mean performance with equal or smaller reported standard deviations across the evaluated benchmarks (*e.g.*,± 2 versus ± 3 on MME). We therefore select the balanced 18k calibration set as the default for the main experiments, prioritizing the lower run-to-run variation observed in this ablation, unless otherwise specified.

Table 8: **Ablation on calibration set size and sampling strategy.** Results are mean ± standard deviation across five random seeds. Performance largely saturates at 9k samples; the default calibration set used outside this ablation contains 18k samples.

Calibration Size	Sampling Strategy	MME	TextVQA	MMVet
4.5k	Balanced (Avg)	1814 ± 6	56.6 ± 0.3	32.0 ± 0.3
4.5k	Random	1809 ± 15	56.4 ± 0.6	32.1 ± 0.4
9k	Balanced (Avg)	1845 ± 3	57.5 ± 0.1	33.2 ± 0.1
9k	Random	1840 ± 5	57.6 ± 0.2	33.2 ± 0.2
18k	Balanced (Avg)	1845 ± 2	57.5 ± 0.1	33.3 ± 0.0
18k	Random	1845 ± 3	57.6 ± 0.1	33.2 ± 0.1

Empirical Cost of Offline Calibration and Category Preprocessing. Table 9 reports resource measurements for offline calibration and category preprocessing on a NVIDIA A100 (80 GB) GPU.

With cached vision-encoder features, the searches for layer configurations and split ratios take approximately 1.5 and 1.0 hours, with memory footprints of 19.4 and 17.6 GB, respectively. The selected configurations remain fixed during evaluation.

For benchmark evaluation, we first classify the prompts and save their predicted categories, then run the MLLM using these categories for fusion, pruning, and answer generation. The average classification time per question is below 0.2 ms, computed as the measured classification-pass runtime divided by the number of classified questions. The classifier's standalone peak GPU memory is 15.5 GB, including its model weights. The workflow timings in Table 5 and Table 23 include category preprocessing. Memory in these tables denotes the peak GPU memory measured over the complete two-pass workflow.

Table 9: **Resource costs of offline calibration and category preprocessing.** Classifier time is the average preprocessing time per question. Classifier memory includes its model weights. Calibration times are total search durations. Table 5 and Table 23 report workflow timings including category preprocessing and peak GPU memory over the complete execution.

Component	Time Cost	Memory Usage
Intent Classifier (Category Preprocessing)	< 0.2 ms/question	15.5 GB
Calibration of Layer Weights (W) (Offline)	≈ 1.5 h	19.4 GB
Calibration of Split Ratios (a) (Offline)	≈ 1.0 h	17.6 GB

D.2 Additional Results and Ablations

Token Pruning Visualization. Figure 5 visualizes the spatial distribution of retained tokens across representative samples. Unlike baselines that often discard salient regions, CLASP consistently preserves task-critical areas: blue regions denote attention-selected tokens that emphasize instruction relevance, whereas red regions denote similarity-selected tokens that maintain complementary visual coverage. Even as the pruning rate increases, the retained set remains aligned with foreground objects while suppressing redundant background patches. This qualitative evidence confirms that the proposed relevance–coverage strategy preserves the visual cues needed for robust downstream reasoning across diverse multimodal benchmarks and challenging deployment settings.

Ablation of Core Components. To isolate and quantify the individual contributions of our proposed modules, we conduct a component-wise ablation study on LLaVA-v1.5-7B under a fixed effective budget of $R = 192$. Table 10 presents the performance of the baseline pruning model, the model augmented solely with Class-Adaptive Layer Fusion (+Fusion), the model utilizing only the Category-Dependent Split Ratio (+Ratio), and the complete CLASP framework combining both (+Both).

The results demonstrate that each component independently provides robust improvements over the baseline. Introducing the class-adaptive layer fusion yields substantial gains (*e.g.*, $+93$ points on MME and $+2.0$ on GQA), highlighting the critical importance of dynamically matching the visual

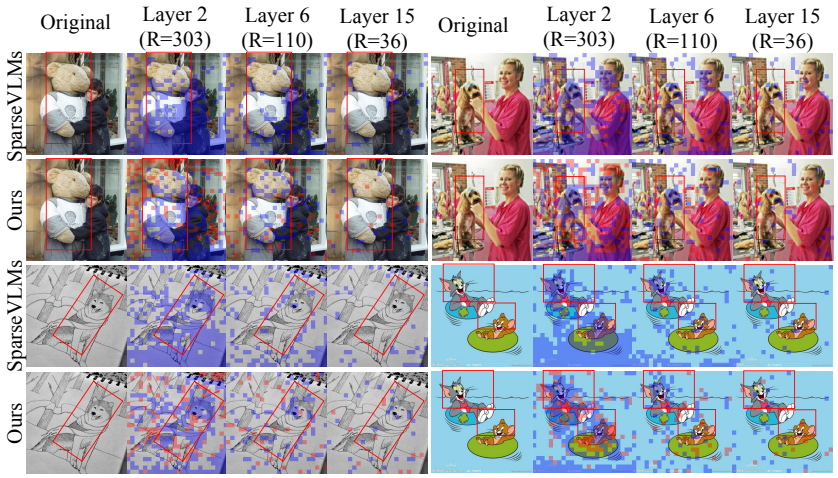

Figure 5: Example visualization of the original image and the corresponding token-retention map.

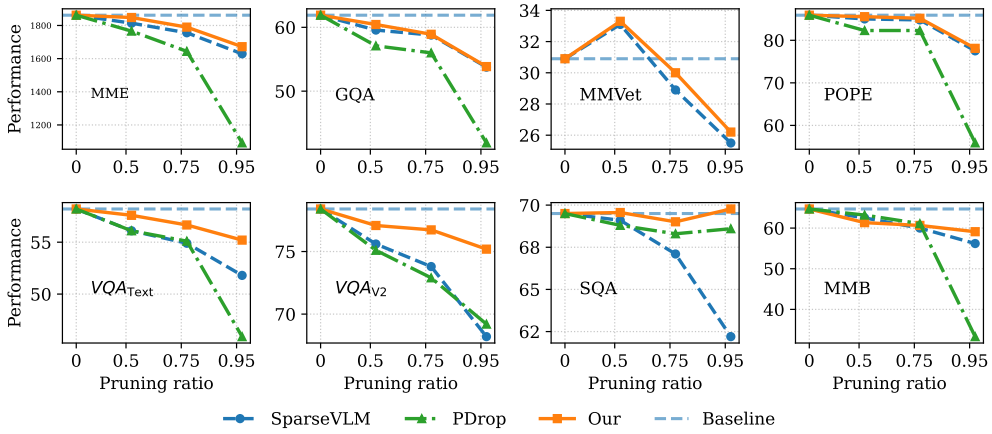

Figure 6: **Per-benchmark performance under increasing token pruning.** We plot performance as a function of pruning ratio on eight evaluation suites, comparing our method with representative pruning baselines (SparseVLM and PDrop). The dashed line denotes the unpruned model performance. Our method exhibits the slowest degradation and stays consistently closest to the unpruned upper bound, particularly at aggressive pruning ratios.

abstraction depth to the user's intent. Similarly, adjusting the relevance-coverage pruning budget based on the prompt category (+Ratio) consistently boosts performance across all tasks. Ultimately, the integration of both components achieves the highest accuracy across all metrics (*e.g.*, MME reaching 1848 and GQA reaching 60.4), proving that optimal visual representation extraction and adaptive budget allocation are highly complementary.

Table 10: **Ablation study of core components.** We evaluate Class-Adaptive Layer Fusion (+Fusion) and dual-stage pruning with a Category-Dependent Split Ratio (+Ratio) on LLaVA-v1.5-7B ($R = 192$). Each module improves the baseline, and their combination yields the best performance among the evaluated variants.

Config	MME	TextVQA	GQA	POPE
Baseline	1721	56.1	57.6	83.6
+ Fusion	1814	56.9	59.6	85.0
+ Ratio	1787	56.6	58.4	84.7
+ Both (CLASP)	**1848**	**57.6**	**60.4**	**85.6**

Sensitivity analysis of pruning ratios across diverse benchmarks. Figure 6 provides a comprehensive visualization of performance trajectories under varying degrees of pruning ratio (ranging from 0% to 95%). We compare our proposed method against two representative state-of-the-art baselines: SparseVLM and PDrop. Three key observations can be drawn from these results. First, our method (orange line) consistently forms the upper envelope of the performance curves across all eight distinct evaluation suites, demonstrating superior token retention capabilities compared to the baselines. Second, we observe a phenomenon of graceful degradation: while competing methods, particularly PDrop (green), suffer from catastrophic performance collapse in the high-pruning regime (pruning ratio > 0.75), our approach maintains remarkably robust efficacy. For instance, on the MME and GQA benchmarks, even when discarding 95% of visual tokens, our method retains a significant portion of the unpruned baseline accuracy, whereas PDrop approaches severely degraded performance. Third, in challenging reasoning tasks such as MMVet (note that PDrop results are unavailable for this benchmark) and SQA, our method exhibits high resilience, significantly widening the performance gap against SparseVLM as the compression rate increases. These findings corroborate that our similarity-based selection strategy effectively isolates and preserves semantically critical visual regions, ensuring model reliability even under extreme computational constraints.

Performance analysis on a larger language model. To further evaluate the robustness of our method under extreme token scarcity, we compare CLASP against SparseVLM across four distinct pruning levels (effective budgets $R = 192, 128, 64, 32$). Table 11 details these results. CLASP consistently outperforms SparseVLM across all compression ratios, demonstrating superior information retention capabilities. Specifically, in the high-retention regime (192 tokens), CLASP achieves an average performance retention of 98.1%, closely matching the upper bound. At the aggressive pruning level of 64 tokens (88.9% reduction), CLASP maintains 92.4% of the original performance, significantly surpassing SparseVLM's 87.2%. Even under the extreme constraint of 32 tokens, where the baseline performance drops sharply to 80.2%, CLASP sustains a robust 87.3%. This trend is particularly evident in fine-grained tasks such as POPE and TextVQA, suggesting that our method effectively preserves critical visual semantics.

Performance analysis across varying token budgets. Tables 12 and 13 compare CLASP against SparseVLM on LLaVA-NeXT-7B and LLaVA-NeXT-13B under three budgets (640/320/160). Across all benchmarks, CLASP consistently surpasses the baseline and exhibits graceful degradation as the budget shrinks. With high retention (640 tokens), CLASP approaches the unpruned upper bound, reaching 97.0% (7B) and 97.2% (13B) average performance, improving over SparseVLM by 2.6 and 2.3 points, respectively. As sparsity increases, the advantage becomes more pronounced: at 320 tokens, CLASP attains 95.7% vs. 89.9% on 7B (+5.8) and 95.6% vs. 92.7% on 13B (+2.9). Under the most aggressive setting (160 tokens), CLASP maintains 92.2% (7B) and 92.4% (13B), while SparseVLM drops to 84.5% and 88.5%, widening the gap to 7.7 and 3.9 points. Notably, CLASP yields consistent gains on grounding- and reasoning-sensitive benchmarks (*e.g.*, GQA/MME) under heavy pruning, indicating that it better preserves critical visual evidence for multi-step inference.

Table 11: Performance comparison of various methods on LLaVA-v1.5-13B across different benchmarks. Results are shown for different pruning ratios, with accuracy and average performance highlighted. Best results in **blue**.

Methods	GQA	MMB	MME	POPE	SQA	VQA_{Text}	Average
Upper Bound	63.3	68.9	1818	85.9	72.8	61.8	100.0%
LLaVA-1.5 13B	*Effective budget* $R = 192$ ($\downarrow$ **66.7%**)						
SparseVLM	58.7	67.4	1768	82.2	**73.1**	**59.6**	96.7%
CLASP (ours)	**60.1**	**68.0**	**1785**	**86.2**	72.8	**59.6**	**98.1%**
LLaVA-1.5 13B	*Effective budget* $R = 128$ ($\downarrow$ **77.8%**)						
SparseVLM	57.9	65.8	1774	81.1	69.9	58.4	94.9%
CLASP (ours)	**59.6**	**68.0**	**1789**	**85.7**	**73.3**	**59.0**	**97.9%**
LLaVA-1.5 13B	*Effective budget* $R = 64$ ($\downarrow$ **88.9%**)						
SparseVLM	54.0	61.3	1641	65.0	69.0	54.6	87.2%
CLASP (ours)	**56.2**	**64.4**	**1675**	**77.2**	**72.8**	**55.6**	**92.4%**
LLaVA-1.5 13B	*Effective budget* $R = 32$ ($\downarrow$ **94.4%**)						
SparseVLM	50.2	56.2	1451	55.9	67.9	50.7	80.2%
CLASP (ours)	**53.5**	**60.7**	**1592**	**68.7**	**71.9**	**52.6**	**87.3%**

Table 12: Performance comparison of various methods on LLaVA-NeXT-7B across different benchmarks. Results are shown for different pruning ratios, with accuracy and average performance highlighted. Best results in **blue**.

Methods	GQA	MMB	MME	POPE	SQA	VQA_{Text}	Average
Upper Bound	64.2	67.4	1851	86.5	70.1	64.9	100.0%
LLaVA-NeXT 7B	*Effective budget* $R = 640$						
SparseVLM	61.2	62.2	1697	85.3	67.6	59.7	94.4%
CLASP (ours)	**63.1**	**62.8**	**1746**	**87.9**	**69.8**	**61.8**	**97.0%**
LLaVA-NeXT 7B	*Effective budget* $R = 320$						
SparseVLM	56.1	60.6	1533	82.4	66.1	58.4	89.9%
CLASP (ours)	**62.7**	**63.2**	**1723**	**85.8**	**67.0**	**61.7**	**95.7%**
LLaVA-NeXT 7B	*Effective budget* $R = 160$						
SparseVLM	55.8	56.9	1420	78.2	57.8	55.9	84.5%
CLASP (ours)	**62.1**	**59.8**	**1699**	**84.5**	**60.9**	**59.3**	**92.2%**

Performance analysis on InternVL2-26B. To further validate the scalability and robustness of our class-adaptive pruning framework on more recent and massive multimodal foundation models, we extended our evaluation to InternVL2-26B Chen et al. (2024b). As shown in Table 14, we compared CLASP against existing token reduction baselines (FastV and ToMe) under a retention fraction of $\eta = 35\%$. Our method demonstrates exceptional information preservation capabilities at this massive scale. For example, on TextVQA and MMVet, CLASP retains high accuracy (81.7 and 63.2), substantially outperforming both FastV (75.6 and 45.0) and ToMe (75.7 and 52.5). Notably, on GQA, CLASP (65.0) even marginally exceeds the unpruned upper bound (64.9), suggesting that our redundancy-aware pruning effectively acts as a noise filter, benefiting complex compositional reasoning. These supplementary results confirm that CLASP remains highly effective and generalizable, regardless of the underlying model's parameter size.

Supplementary Evaluations on Diverse Open Datasets. To further validate the generalization and robustness of our approach beyond standard academic benchmarks, we conduct supplementary evaluations on a diverse set of open datasets. These include open-ended conversational evaluation (LLaVA-Bench, MMVet), real-world noisy visual perception (VizWiz), cross-lingual multimodal understanding (MMBench-Chinese), and fine-grained spatial reasoning (SEED-Bench).

As summarized in Table 16, we compare CLASP against a strong token reduction baseline, SparseVLM, across different effective budgets ($R = 192, 128, 64$). While both methods perform comparably well at a higher budget ($R = 192$), CLASP demonstrates significantly superior robustness under aggressive pruning conditions. For instance, at an extreme sparsity level of $R = 64$, CLASP outperforms SparseVLM by a massive margin of $+6.5\%$ on LLaVA-Bench and $+9.5\%$ on MMB-CN. This substantial gap underscores that our class-adaptive layer fusion and dual-stage pruning mechanisms

Table 13: Performance comparison of various methods on LLaVA-NeXT-13B across different benchmarks. Results are shown for different pruning ratios, with accuracy and average performance highlighted. Best results in **blue**.

Methods	GQA	MMB	MME	POPE	SQA	VQA_{Text}	Average
Upper Bound	65.4	70.0	1901	86.2	73.5	64.3	100.0%
LLaVA-NeXT 13B	*Effective budget* $R = 640$						
SparseVLM	62.7	62.0	1821	86.0	71.5	59.3	94.9%
CLASP (ours)	**64.2**	**63.2**	**1867**	**86.8**	**73.5**	**61.5**	**97.2%**
LLaVA-NeXT 13B	*Effective budget* $R = 320$						
SparseVLM	60.9	60.8	1798	83.5	70.6	57.2	92.7%
CLASP (ours)	**63.0**	**62.8**	**1848**	**85.2**	**71.1**	**60.8**	**95.6%**
LLaVA-NeXT 13B	*Effective budget* $R = 160$						
SparseVLM	59.4	56.5	1755	81.1	66.2	53.3	88.5%
CLASP (ours)	**61.8**	**59.8**	**1801**	**83.7**	**68.1**	**57.8**	**92.4%**

Table 14: Performance comparison on InternVL2-26B across widely-used benchmarks. Results are shown for an aggressive pruning ratio (retaining 35% of visual tokens), with accuracy highlighted. Best results in **blue**.

Methods	TextVQA	MME	GQA	MMVet
Upper Bound	82.5	2270	64.9	64.0
InternVL2 26B	*Retain 35% Tokens*			
FastV	75.6	2140	61.2	45.0
ToMe	75.7	2178	63.6	52.5
CLASP (ours)	**81.7**	**2262**	**65.0**	**63.2**

effectively preserve the critical visual tokens necessary for complex, open-ended generation and cross-lingual semantic alignment, whereas static pruning methods suffer severe degradation.

Ablation study of layer-mixture strategies across different task categories. Table 15 and Figure 7 report a category-targeted layer-mixture ablation on LLaVA-v1.5-7B under a fixed effective token budget of $R = 192$. We evaluate five representative fusion strategies (A–E), which combine visual features from different ViT layers using the layer indices and coefficients specified in Table 15.

Category-targeted configuration. This ablation follows a one-category-at-a-time protocol. For a target category $c \in \{0, \ldots, 8\}$ and a candidate fusion strategy $m \in \{A, B, C, D, E\}$, strategy m is applied only to samples assigned to category c. Samples assigned to all other categories retain the original backbone's default visual-feature readout. For LLaVA-v1.5-7B, this corresponds to using the penultimate ViT layer, rather than the calibrated category-specific fusion configurations used by the complete CLASP model.

Here, the backbone's default readout refers specifically to its original visual-feature extraction rule. It should not be confused with the calibrated fusion configuration associated with the "Default" category (Class 8). When Class 8 is the target category, the candidate strategy is applied to Class 8 samples, while samples in Classes 0–7 use the backbone's default readout.

Score definition. Each table entry or heatmap cell reports the overall benchmark score on the entire evaluation split under the corresponding category-targeted configuration. It does not report the score computed only on samples belonging to the target category. The column identifies the category whose fusion strategy is modified, and the row identifies the candidate strategy applied to that category. Consequently, each cell corresponds to a separate evaluation configuration. The nine category columns are not category-level scores obtained from a single configuration.

Aggregation and interpretation. For a given benchmark, let $S(c, m)$ denote the overall benchmark score when strategy m is applied only to samples assigned to category c, with all other categories using the backbone's default readout. The *Avg* column in Table 15 is defined as

$$\mathrm{Avg}(m) = \frac{1}{9} \sum_{c=0}^{8} S(c, m).$$

Table 15: **Ablation on layer-mixture strategies across different task categories with $R = 192$.** The columns correspond to specific task types: (0) Object Identification, (1) Attribute/Breed ID, (2) Text/Symbol Recognition, (3) Scene Understanding, (4) Spatial Relations, (5) Counting, (6) Action/Interaction, (7) Intention/Function, and (8) Default. **Layer Mixture Legend:** A: $0.2L_2 + 0.3L_6 + 0.5L_{11}$; B: $0.2L_5 + 0.3L_{15} + 0.5L_{22}$; C: $0.2L_{12} + 0.3L_{15} + 0.5L_{19}$; D: $0.2L_5 + 0.8L_{22}$; E: $0.2L_{20} + 0.8L_{22}$. Best results are highlighted in **blue**, and the row with the highest average score in each dataset is shaded.

Layers	Class 0	Class 1	Class 2	Class 3	Class 4	Class 5	Class 6	Class 7	Class 8	Avg
					MMVet ($R = 192$)					
A	28.0	28.2	28.6	28.5	27.8	28.4	28.3	**28.7**	28.3	28.3
B	**28.9**	28.3	27.1	**28.8**	25.6	27.8	28.2	28.1	28.2	27.9
C	28.4	**28.6**	25.8	28.5	22.9	28.1	**28.9**	28.4	**28.4**	27.6
D	28.7	28.0	**29.1**	28.3	28.3	**29.5**	28.8	28.5	28.2	**28.6**
E	28.2	28.3	27.9	28.4	**28.6**	27.9	28.1	28.0	28.2	28.2
					TextVQA ($R = 192$)					
A	57.56	57.56	55.94	57.51	**57.60**	57.19	57.55	57.58	57.51	57.33
B	**57.62**	57.58	45.43	57.54	57.57	56.64	57.57	57.55	**57.57**	56.12
C	57.61	57.59	12.03	57.56	57.55	56.64	57.53	57.49	57.53	52.39
D	57.58	57.58	57.44	**57.60**	57.57	57.53	57.56	**57.59**	57.55	57.56
E	57.59	**57.62**	**57.74**	57.54	57.54	**57.54**	**57.60**	**57.59**	57.34	**57.57**
					SQA ($R = 192$)					
A	68.62	68.47	63.41	**68.52**	**68.72**	**68.77**	**68.57**	68.57	**68.67**	68.04
B	**68.72**	**68.67**	54.78	68.27	68.57	68.17	**68.57**	68.57	**68.67**	67.00
C	68.52	68.62	51.51	68.17	68.67	68.42	68.52	68.57	68.57	66.62
D	68.47	68.52	68.27	**68.52**	68.57	68.62	**68.57**	68.62	68.52	68.52
E	68.67	68.62	**68.42**	68.47	68.62	68.57	68.52	**68.67**	**68.67**	**68.58**
					MME ($R = 192$)					
A	**1793**	1790	1702	1774	1778	1785	**1788**	1780	1778	1774.2
B	1786	1791	1433	1776	1785	1784	**1788**	**1788**	1777	1745.3
C	1780	1783	1321	1787	**1788**	1777	1783	1781	1786	1731.8
D	1788	**1792**	**1806**	1785	1784	1779	1785	1784	1785	**1787.6**
E	1782	1786	1790	**1793**	1783	**1788**	1778	**1788**	**1794**	1786.9

Table 16: Supplementary evaluations on diverse open datasets using LLaVA-v1.5-7B under varying token budgets. We compare our CLASP framework against SparseVLM. The best results between the two pruning methods at each budget are highlighted in bold.

Budget	Method	LLaVA-B	MMVet	VizWiz	MMB-CN	MMB	SEED
Unpruned	LLaVA-v1.5-7B	66.8	30.9	50.0	58.1	64.7	66.2
R=192	SparseVLM	66.1	33.1	50.5	53.7	**62.5**	64.2
	CLASP (Ours)	**66.7**	**33.3**	**52.1**	**57.9**	61.3	**65.4**
R=128	SparseVLM	62.7	29.0	51.4	51.1	60.0	**63.6**
	CLASP (Ours)	**65.6**	**30.0**	**51.9**	**57.1**	**60.7**	63.0
R=64	SparseVLM	57.5	24.9	50.1	46.1	56.2	56.8
	CLASP (Ours)	**64.0**	**26.2**	**51.6**	**55.6**	**59.1**	**58.5**

This quantity is a descriptive mean over nine separate category-targeted configurations. It is neither a macro-average of per-category scores nor the measured benchmark score of applying strategy m to all categories simultaneously.

Within a column, the entries compare overall benchmark scores under alternative fusion strategies for the same target category. Across columns, the entries correspond to configurations targeting different categories. Accordingly, references to a category's score in the discussion of this ablation should be read as the overall benchmark score under the configuration targeting that category, rather than its standalone accuracy.

Ablation study of similarity and attention for different classes. Table 17 and Figure 8 present the performance variations across nine distinct task categories under varying attention selection ratios

Table 17: **Ablation study on the sensitivity of different task categories to the attention selection ratio p (with fixed $R = 192$).** The columns correspond to specific task types mapped as follows: (0) Object Identification, (1) Attribute/Breed ID, (2) Text/Symbol Recognition, (3) Scene Understanding, (4) Spatial Relations, (5) Counting, (6) Action/Interaction, (7) Intention/Function, and (8) Default. Best results are highlighted in **blue**, and the row with the robust average score in each dataset is shaded.

p	Class 0	Class 1	Class 2	Class 3	Class 4	Class 5	Class 6	Class 7	Class 8
					MME ($R = 192$)				
0.1	1815	1810	**1815**	1812	1815	1782	1812	1812	1813
0.3	1809	**1814**	1770	1814	**1818**	1802	1805	1809	1804
0.5	1799	1813	1791	1812	1811	1801	1813	**1815**	1811
0.7	**1817**	1812	1813	**1828**	**1818**	1812	1812	1814	1813
0.9	1816	1813	1809	1816	1817	**1816**	1813	1814	**1814**
1.0	1814	**1814**	1814	1814	1814	1814	**1814**	1814	**1814**
					MMVet ($R = 192$)				
0.1	29.6	30.2	30.4	29.6	30.0	29.9	29.0	29.8	29.2
0.3	29.4	31.0	**30.6**	29.7	30.1	30.3	29.0	30.1	30.0
0.5	29.9	29.7	30.1	**31.0**	29.4	30.0	28.8	30.1	30.4
0.7	**30.2**	30.4	29.5	29.7	29.2	**30.5**	28.6	30.2	30.5
0.9	29.3	**31.2**	29.5	29.9	**30.4**	29.6	**30.2**	**30.5**	**30.7**
1.0	29.8	29.8	29.8	29.8	29.8	29.8	29.8	29.8	29.8
					TextVQA ($R = 192$)				
0.1	56.88	56.48	55.89	56.86	56.88	56.97	56.97	56.98	56.92
0.3	56.96	56.66	56.24	56.90	56.90	**57.48**	**56.99**	**57.02**	**57.10**
0.5	56.96	56.90	56.50	56.93	56.93	57.01	56.94	56.95	56.98
0.7	56.96	57.07	56.85	56.93	56.98	57.07	56.92	56.99	56.90
0.9	56.96	**57.09**	**57.00**	56.94	56.96	57.00	56.97	**57.02**	56.91
1.0	**56.99**	56.99	56.99	**56.99**	**56.99**	56.99	**56.99**	56.99	56.99
					SQA ($R = 192$)				
0.1	68.37	68.27	**68.37**	**69.56**	68.02	**68.32**	68.27	68.32	65.29
0.3	68.27	68.12	68.17	68.82	68.22	**68.32**	68.27	**68.42**	64.95
0.5	68.32	68.27	**68.37**	68.47	68.27	**68.32**	68.27	68.22	64.35
0.7	**68.42**	**68.32**	**68.37**	68.22	68.37	**68.32**	68.27	68.22	65.64
0.9	68.17	68.22	68.27	68.32	**68.67**	68.22	**68.32**	68.17	66.39
1.0	68.27	68.27	68.27	68.27	68.27	68.27	68.27	68.27	**68.27**

p, where p denotes the proportion of retained tokens based on attention scores (ranging from 0.1 for high sparsity to 1.0 for full attention). Our results indicate that the optimal attention ratio is highly task-dependent. First, we observe that the model maintains competitive performance even at lower ratios (*e.g.*, $p = 0.3$ or $p = 0.5$) across benchmarks like MMVet and SQA, demonstrating the redundancy in standard visual tokens. More notably, reducing p can lead to performance gains over the baseline ($p = 1.0$). For instance, in the MME dataset, Class 3 (Scene Understanding) achieves a peak score of 1828 at $p = 0.7$, surpassing the full-attention score of 1814. Similarly, TextVQA shows improved accuracy in Class 5 (Counting) at $p = 0.3$. These results demonstrate that the optimal attention ratio is highly category-dependent, with specific tasks achieving peak performance at intermediate p values rather than the full-attention baseline.

Table 18: Impact of the number of clustering iterations on performance ($R = 192$).

Iterations	MME	GQA	MMVet	POPE	SQA	TextVQA
3	1828	60.41	31.1	85.57	69.58	56.83
5	1848	60.44	33.3	85.55	69.56	57.59
7	1823	59.08	31.3	85.48	68.32	56.81
9	1762	59.02	30.5	84.33	68.37	56.92
11	1757	58.61	30.1	84.41	68.22	56.94

Clustering iterations. Under a fixed effective budget ($R = 192$), we ablate the number of clustering refinement steps used in the similarity-driven stage of our pruning (Stage II), where token affinity

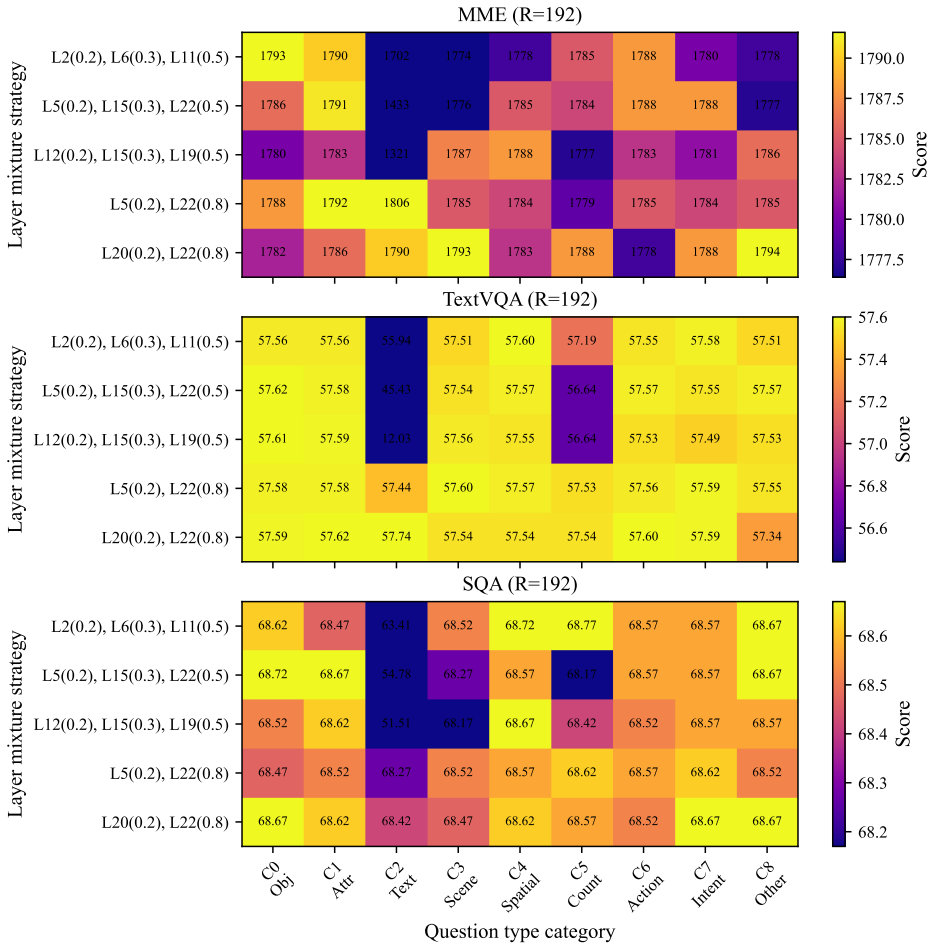

Figure 7: Layer mixture ablation under a fixed token budget ($R = 192$) on MME, TextVQA and SQA. Rows are layer mixture strategies with weights in parentheses. Columns are question types: C0 object identification, C1 attribute or breed identification, C2 text or symbol recognition, C3 scene understanding, C4 spatial relations, C5 counting, C6 action or interaction, C7 intention or function, C8 default.

Table 19: **Effect of initialization under a fixed budget** (R=192)**.** We compare our search-derived weight configuration with three initialization schemes while keeping the backbone frozen and using the same effective budget. Evaluation covers three general multimodal benchmarks (MMBench, MMVet, MME), four VQA/QA benchmarks (TextVQA, VQA-v2, GQA, SQA), and the hallucination benchmark POPE. Higher is better for all metrics. Best and second-best results are **bold** and underlined, respectively.

Methods	MMBench	MMVet	MME	TextVQA	VQA-v2	GQA	SQA	POPE
Search-derived Weights (fixed W)	**61.34**	**33.3**	**1848**	**57.59**	77.08	**60.44**	**69.56**	85.55
Search-derived Weights *init* + Calibrated W	60.13	32.2	1759	56.15	77.08	60.22	69.39	85.58
Penultimate-biased *init* + Calibrated W	57.7	33.2	1722	56.05	**77.14**	59.78	68.69	**85.67**
Uniform *init* + Calibrated W	17.70	12.8	1127	30.28	62.43	43.13	47.63	78.03

is computed by cosine similarity on ℓ_2-normalized aligned features, *i.e.*, $\mathrm{sim}(t, t') = \mathbf{u}_{n,t}^{\top}\mathbf{u}_{n,t'}$ (Eq. (10)). As shown in Table 18, $T = 5$ achieves the highest scores on MME, GQA, MMVet, and TextVQA among the tested iteration counts, while maintaining competitive performance on POPE and SQA. We therefore use $T = 5$ as the default number of spherical K-means refinement iterations.

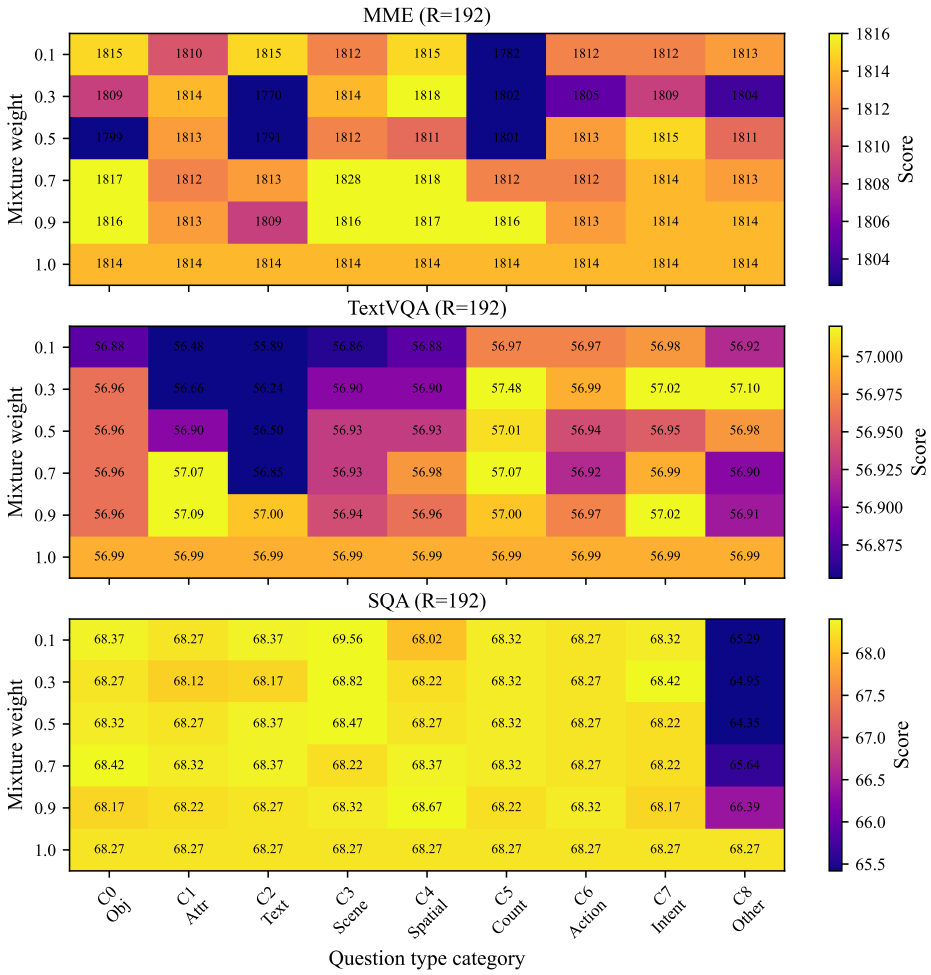

Figure 8: Ablation on attention and similarity mixture weight under a fixed token budget ($R = 192$) on MME, TextVQA and SQA. Rows are the mixture ratios. Columns are question types: C0 object identification, C1 attribute or breed identification, C2 text or symbol recognition, C3 scene understanding, C4 spatial relations, C5 counting, C6 action or interaction, C7 intention or function, C8 default.

Qualitative Analysis. Figure 9 and Figure 10 present a qualitative comparison of visual token pruning. We visualize the results across different network layers and varied pruning ratios (R). In these visualizations, the red bounding boxes denote the ground-truth target regions. The retained tokens are visualized to illustrate the rationale behind their selection. Specifically, blue points indicate regions preserved primarily due to their attention significance. Meanwhile, red points represent regions selected based on similarity metrics. As clearly observed, our method demonstrates superior capability in semantic preservation compared to state-of-the-art baselines such as PDrop and SparseVLM. Methods like PDrop and SparseVLM often struggle to maintain the semantic structure. They tend to lose critical foreground information or retain excessive background noise. This limitation is particularly evident at a nominal effective-budget reduction of 88.9%. In contrast, our approach consistently aligns the retained tokens with the target object within the red boxes. This robust alignment is maintained across all depths, including Layers 2, 6, and 15. These results indicate that our method effectively filters redundancy. It successfully maintains a focused representation of the region of interest throughout the inference process.

Search-derived Weights vs. Further Calibration. To assess the robustness of the selected parameters, we compare our *Search-derived Weights* (fixed W) against a variant initialized with the same

weights and further calibrated on a newly sampled 18,000-sample calibration set (*Search-derived Weights init + Calibrated* W), under a fixed effective token budget of $R = 192$. Specifically, we construct this further-calibration set by drawing 2,000 samples from each of the nine categories in `llava_v1_5_mix665k.json` through a new sampling round, rather than reusing the original 18,000-sample set used for discrete search. The MLLM backbone remains frozen throughout. Table 19 shows that the fixed *Search-derived Weights* yield the strongest overall performance: it improves MMBench (61.34 vs. 60.13), MMVet (33.3 vs. 32.2), MME (1848 vs. 1759), and TextVQA (57.59 vs. 56.15), and provides consistent gains on GQA/SQA, while matching VQA-v2 and remaining comparable on POPE. Moreover, naive alternatives are clearly suboptimal: a penultimate-biased initialization underperforms, and uniform averaging across layers severely degrades all benchmarks, highlighting the effectiveness of our discrete search strategy.

We hypothesize that this gap reflects a mismatch between *continuous optimization behavior* and *what token pruning needs to preserve*. Vision encoders exhibit a strong depth hierarchy: shallow layers retain high-frequency local evidence (*e.g.*, edges, strokes, and fine layouts) that is crucial for OCR- and counting-heavy queries, whereas deep layers emphasize invariant semantics. Our *Search-derived Weights* effectively capture this inductive bias by finding an optimal allocation of mixture mass to shallow/mid layers, ensuring that fine-grained evidence is not washed out. In contrast, further continuous calibration of W on limited data can exhibit shortcut behavior: optimization is naturally attracted to deep-layer features that yield strong, easy-to-fit semantic signals, resulting in a deep-biased mixture that is locally optimal for coarse semantics but dilutes the precise visual cues required by detail-sensitive tasks (*e.g.*, TextVQA). This explanation is consistent with common ablation patterns reported in recent VLM pruning studies, where shallow/early cues often matter disproportionately for fine-grained perception, and overly aggressive averaging or smoothing across layers harms OCR-centric performance.

Comparison with Static Multi-Layer Fusion Baseline. To further validate the necessity and superiority of our class-adaptive layer fusion strategy, we compare CLASP against a strong static multi-layer fusion baseline. Specifically, we adopt the optimal static layer combination $[2, 17, 23]$ (0-indexed, corresponding to layers 3, 18, and 24) identified as the empirical best practice in recent literature Lin et al. (2025a).

As shown in Table 20, we evaluate both methods on LLaVA-v1.5-7B across various effective budgets ($R = 192, 128, 64$). While the optimally searched static fusion provides a reasonable baseline, CLASP consistently outperforms it across all budgets and benchmarks. For instance, at a moderate budget of $R = 192$, CLASP achieves a significant $+92$ point improvement on MME and $+3.0\%$ on GQA compared to the static baseline. The performance gap widens further under extreme sparsity ($R = 64$), where CLASP retains 1709 on MME versus the static baseline's 1614 ($+95$ points). These results strongly suggest that different instruction intents inherently require distinct levels of visual abstraction. A fixed combination of layers, even when globally optimized, is fundamentally limited compared to our dynamic, class-conditioned routing approach.

Table 20: Performance comparison between the optimal Static Fusion baseline and our Class-Adaptive Fusion (CLASP) on LLaVA-v1.5-7B under varying token budgets. Best results are highlighted in bold.

Budget	Method	MME	GQA	POPE	TextVQA
Unpruned	LLaVA-v1.5-7B	1862	61.9	85.9	58.2
R=192	Static [3, 18, 24]	1756	57.4	83.9	57.6
	CLASP (Ours)	**1848**	**60.4**	**85.6**	**57.6**
R=128	Static [3, 18, 24]	1701	56.2	82.7	55.3
	CLASP (Ours)	**1790**	**58.9**	**85.2**	**56.7**
R=64	Static [3, 18, 24]	1614	54.1	78.2	51.9
	CLASP (Ours)	**1709**	**57.0**	**82.8**	**55.2**

Robustness Analysis across Random Seeds. To assess the run-to-run stability of CLASP, we conducted a variance analysis on the video QA benchmarks. Specifically, we evaluated our method under a 50% retention budget using five run-level random seeds $(42, 43, 44, 45, 46)$. As shown

in Table 21, the standard deviations across all three video benchmarks (TGIF-QA, MSVD-QA, MSRVTT-QA) are small, ranging from 0.13 to 0.21. Furthermore, the mean scores across the five runs exceed the reported SparseVLM scores on all three benchmarks. These results indicate stable end-to-end performance under the tested evaluation protocol.

Table 21: **Variance analysis of CLASP on video QA benchmarks (TGIF-QA, MSVD-QA, MSRVTT-QA).** Results are reported as mean $\pm$ standard deviation across five random seeds $(42, 43, 44, 45, 46)$ under a 50% retention budget.

Retention Fraction (η)	**TGIF-QA**	**MSVD-QA**	**MSRVTT-QA**
50%	45.53 ± 0.18	61.72 ± 0.21	51.45 ± 0.13

Robustness of the Prompt-to-Class Router. To further evaluate the robustness of our intent classifier and understand the impact of potential routing errors, we conduct two simulated stress tests. First, we simulate classifier uncertainty by forcing a specific proportion of samples to be assigned to the "Default" category. Second, we simulate critical routing failures by randomly misclassifying a certain ratio of samples into incorrect non-default categories.

As shown in Table 22, falling back to the "Default" category yields a highly graceful degradation. Even when 50% of the samples are forced to the default class, the model maintains competitive performance (*e.g.*, MME merely drops to 1756, and POPE to 83.6). In contrast, random misclassification leads to severe performance penalties, with MME dropping to 1722 at a 50% error rate, and experiencing a catastrophic collapse to 1627 at a 100% error rate. These findings not only demonstrate the robustness of our routing mechanism but also strongly justify our design choice: when the router encounters ambiguous or out-of-distribution prompts, falling back to a generalized "Default" prototype effectively preserves a robust baseline representation. This strategy is significantly safer and more effective than making arbitrary categorical guesses.

Table 22: **Ablation on router robustness and classification errors.** We compare the impact of forcing a specific ratio of samples to the "Default" class (top) versus randomly misclassifying them (bottom). Results demonstrate that falling back to the default class under uncertainty is significantly safer than random assignment.

Forced "Default" Ratio	**MME**	**TextVQA**	**SQA**	**MMVet**	**POPE**
5%	1840	57.4	69.5	33.3	85.5
20%	1799	55.0	68.7	32.8	84.7
50%	1756	55.1	68.0	31.3	83.6
100%	1715	53.7	66.7	29.6	82.2
Random Error Ratio	**MME**	**TextVQA**	**SQA**	**MMVet**	**POPE**
5%	1834	57.4	69.4	33.1	85.3
10%	1819	56.0	68.9	32.2	84.6
20%	1789	54.1	67.2	29.3	83.1
50%	1722	51.5	64.9	26.2	80.7
100%	1627	49.7	60.1	23.8	77.4

Detailed Latency and Overhead Breakdown. Table 23 reports execution time, peak GPU memory, and component timings for our two-pass inference workflow under three effective-budget settings. We first predict prompt categories in a preprocessing pass and then use the saved categories for MLLM answer generation. Total Time includes category preprocessing, visual feature extraction and fusion, dual-stage pruning, and MLLM prefill and decoding. Classifier Time reports the average category-preprocessing cost per question and is included in Total Time; it should not be interpreted as single-query online routing latency. Total Prefill measures the complete prefill stage during the second pass, excluding category preprocessing and autoregressive decoding. Memory is the peak GPU memory recorded across both passes under the evaluated execution setup.

With category preprocessing included, reducing the effective budget from $R = 192$ to $R = 64$ decreases Total Time from 223.1 ms to 155.8 ms, while Total Prefill decreases from 125.2 ms to

74.7 ms. In Table 5, Time reports the total evaluation runtime in min:sec over $N = 8{,}910$ POPE questions and includes both passes for CLASP. For all methods, Latency is computed as T/N in seconds per question, where T is the total runtime converted to seconds. It is an amortized workflow-level metric rather than an independently measured single-query online response latency. The Mem. column follows the same workflow-level peak-memory definition. The reported speedups therefore characterize the two-pass evaluation workflow, including category preprocessing for CLASP, rather than single-query online response latency.

Table 23: **Two-pass inference costs and component timings.** Total Time includes category preprocessing and MLLM answer generation. Memory is the peak GPU memory measured over the complete workflow. †Classifier Time is the average preprocessing time per question and is included in Total Time. Times are in ms, memory in GB, and FLOPs in T.

Effective Budget (R)	Total Time	Memory	FLOPs	Total Prefill	Classifier Time†	Weight Fusion	K-means
192	223.1 ms	17.62 GB	2.67 T	125.2 ms	< 0.2 ms	< 0.5 ms	27.2 ms
128	201.7 ms	17.59 GB	1.97 T	109.6 ms	< 0.2 ms	< 0.5 ms	23.1 ms
64	155.8 ms	17.45 GB	1.39 T	74.7 ms	< 0.2 ms	< 0.5 ms	17.2 ms

Robustness to Router Model Choice. To demonstrate that the CLASP framework is robust and not heavily reliant on the specific instruction-following model used for routing, we conduct an ablation study comparing our default text-only router against alternative architectures. Table 24 details the downstream performance across diverse multimodal benchmarks (MME, TextVQA, SQA, MMVet, and POPE) when substituting the default Qwen3-8B router with Llama3-8B Grattafiori et al. (2024) and a larger Qwen3-32B model.

The results clearly demonstrate that while varying the router model introduces slight differences in the intent assignment distribution (indicated by the Count Variance), the overall multimodal performance remains remarkably stable. Specifically, the MME score fluctuates marginally between 1840 and 1848, and the POPE F1 score remains tightly bounded between 85.3% and 85.7% across all settings. Importantly, this stability indicates that our discrete subspace search strategy and category-conditioned fusion mechanisms are highly resilient to minor routing shifts. This confirms that CLASP is effectively model-agnostic with respect to the prompt-to-class router, ensuring reliable and consistent deployment in diverse environments.

Table 24: **Performance comparison of CLASP across different prompt-to-class router models.** We report the Count Variance (measuring the intent assignment deviation relative to the default Qwen3-8B router) alongside task accuracy across five representative benchmarks. The downstream performance remains highly stable regardless of the chosen router.

Router Model	Count Variance	MME	TextVQA	SQA	MMVet	POPE
Qwen3-8B (Default)	0	1848	57.6	69.6	33.3	85.6
Llama3-8B	37.2	1840	57.5	69.4	32.9	85.7
Qwen3-32B	7.9	1848	57.7	69.4	33.1	85.3

Comparison with Concurrent State-of-the-Art Methods. To further contextualize our contributions against the most recent advancements in the field, we compare CLASP with several concurrent and newly introduced visual token reduction methods: Nuwa Huang et al. (2026), Holov Zou et al. (2026), and BTP Li et al. (2025). We conduct this evaluation on the LLaVA-v1.5-7B architecture under an effective budget of $R = 192$.

As detailed in Table 25, CLASP maintains a strong competitive edge against these latest baselines. Most notably, CLASP achieves the highest performance on complex multi-step reasoning benchmarks, securing 1848 on MME and 33.3 on MMVet, which substantially outperforms the recent Nuwa model (1834 and 30.5, respectively). On OCR-centric and general scientific QA tasks like TextVQA and SQA, CLASP consistently ranks at or near the top (57.6 and 69.6), demonstrating that our dual-stage class-adaptive pruning preserves essential local visual cues better than fixed-strategy concurrent works. Even on hallucination metrics (POPE), our method remains extremely robust (85.6). These results affirm that CLASP represents a leading solution in the rapidly evolving landscape of efficient MLLM inference.

Table 25: **Comparison with concurrent state-of-the-art methods.** Evaluation is performed on LLaVA-v1.5-7B with an effective budget of $R = 192$. "-" indicates that the result is not reported by the respective authors.

Method	MME	TextVQA	SQA	MMVet	POPE
Nuwa (ICLR 2026) Huang et al. (2026)	1834	57.4	68.2	30.5	**86.4**
Holov (NeurIPS 2025) Zou et al. (2026)	1820	57.4	**69.8**	-	85.6
BTP (NeurIPS 2025) Li et al. (2025)	1816	-	69.1	29.1	85.6
CLASP (ours)	**1848**	**57.6**	69.6	**33.3**	85.6

D.3 Qualitative Case Studies

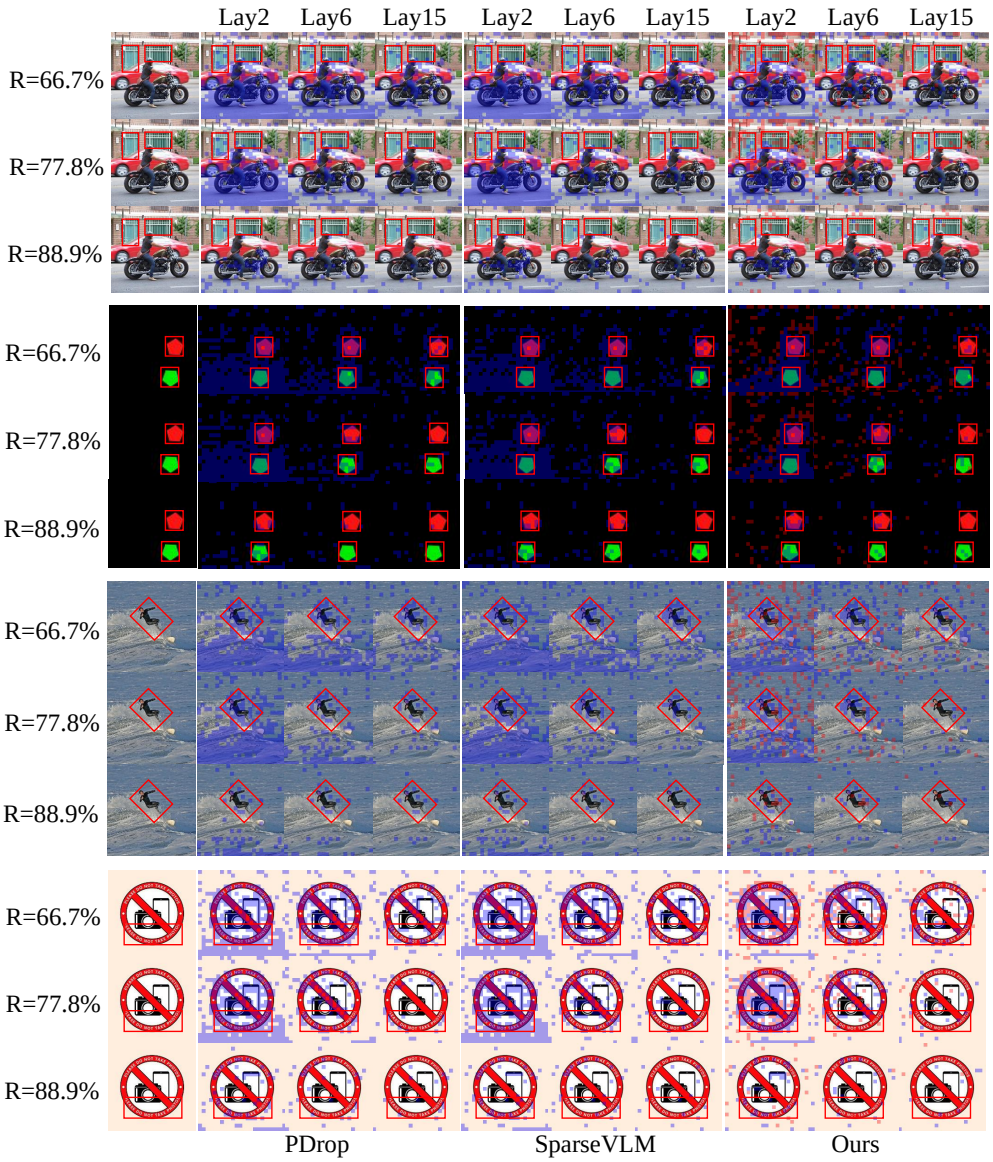

Figure 9: Qualitative comparisons on Cases 1–4. Each case visualizes the original image and the corresponding token-retention map results under three pruning ratios ($R = 66.7\%, 77.8\%, 88.9\%$) at layers 2, 6, and 15.

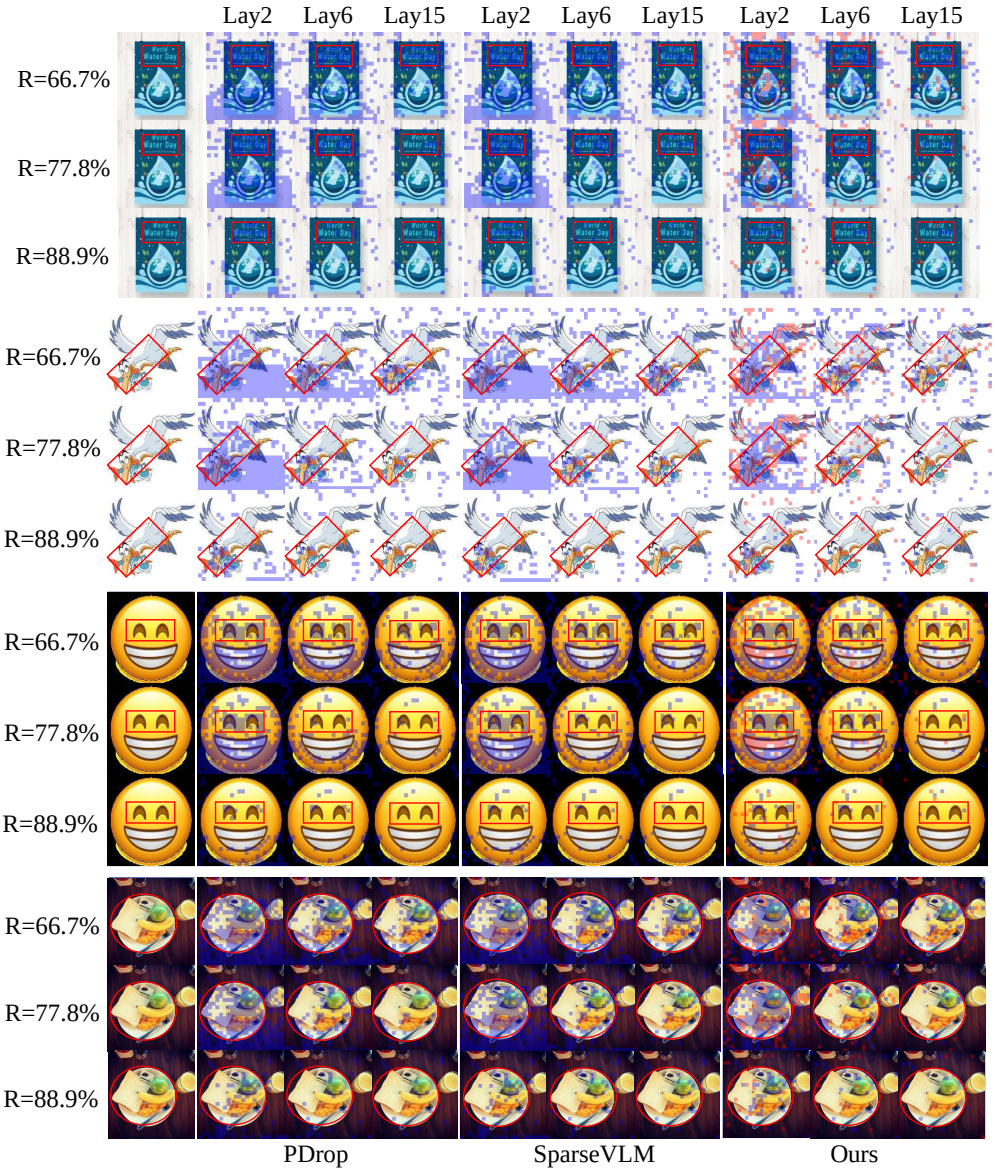

Figure 10: Qualitative comparisons on Cases 5–8. Each case visualizes the original image and the corresponding token-retention map results under three pruning ratios ($R = 66.7\%, 77.8\%, 88.9\%$) at layers 2, 6, and 15.