AVIS: Adaptive Test-Time Scaling for Vision–Language Models

Ahmadreza Jeddi[1,2,3†*] **Minh N. Le**[1,2,3†*] **Amirhossein Kazerouni**[1,2,3*] **Hakki Karaimer**[1]
Hue Nguyen[1] **Iqbal Mohomed**[1] **Michael Brudno**[2,3] **Konstantinos G. Derpanis**[1,3,4]
Alex Levinshtein[1] **Babak Taati**[2,3] **Radek Grzeszczuk**[1†]
[1] AI Center-Toronto, Samsung Electronics [2] University of Toronto
[3] Vector Institute [4] York University

Abstract

Modern Vision-Language Models (VLMs) benefit from chain-of-thought prompting and test-time scaling, but these gains often come with prohibitive inference cost due to large visual contexts and long decoding chains. We view this cost through two coupled axes: Visual Context Scaling (VCS), which controls how much visual evidence is passed to the language model, and Visual Reasoning Scaling (VRS), which controls how much inference-time reasoning search is performed. Existing methods typically optimize one axis at a time, leaving the joint allocation of compute across these axes underexplored. We introduce Adaptive Visual Inference Scaling (AVIS), a lightweight policy that adapts both VCS and VRS per query. AVIS realizes VCS through Key Diversity Visual (KDV) pruning, a training-free $O(N)$ key-based rule for removing redundant visual tokens before prefilling, and realizes VRS through adaptive self-consistency, using a learned difficulty predictor to select the number of reasoning rollouts. AVIS is deployment-friendly and compatible with shared-prefill inference, where all rollouts reuse a single prefilling pass and KV cache. Across diverse image and video reasoning benchmarks, AVIS improves the accuracy–compute trade-off relative to VCS-only and VRS-only baselines, and remains effective on top of RL post-trained VLMs while keeping compute and latency low. Project page: `https://avis-vlm.github.io/`

1 Introduction

Vision-Language Models (VLMs) are rapidly evolving from captioning systems into strong visual reasoners [1, 2, 3]. A key driver has been chain-of-thought (CoT) prompting and reasoning-oriented post-training, which encourage step-by-step rationales instead of single-shot answers [4, 5, 6]. While effective on complex tasks, these advances substantially increase inference cost: high-resolution images or videos induce large visual contexts, and CoT-based inference often requires long generations or repeated rollouts, such as Best-of-N [7, 8] and self-consistency [9]. As VLMs are increasingly deployed, a central challenge is *adaptive visual reasoning*: *adaptively* allocating inference compute, spending little on easy instances and more on challenging ones, without sacrificing accuracy.

For a given input pair (I, Q), where I is an image or video and Q is a query, inference compute is primarily spent along two coupled axes: *seeing* and *thinking*. The *seeing* compute controls how much visual evidence is passed to the language model as context, such as how many visual tokens are retained after compression or pruning. We refer to this as **Visual Context Scaling (VCS)**. Increasing VCS preserves richer evidence, but raises prefilling cost and KV-cache memory. In contrast, the *thinking* compute controls inference-time reasoning search, either *sequentially* through longer or

*Equal contribution. Correspondence to: `ajeddi@cs.toronto.edu`. † Work done at Samsung.

40th Conference on Neural Information Processing Systems (NeurIPS 2026).

iterative CoT traces, or *in parallel* through sampling multiple trajectories and aggregating them with Best-of-N or self-consistency. We refer to this as **Visual Reasoning Scaling (VRS)**. Increasing VRS expands the search over reasoning paths, but increases decoding compute.

Existing inference methods typically optimize one axis at a time, independently. Visual token pruning and compression [10, 11] reduce VCS for efficiency, often under single-pass decoding, while test-time scaling increases VRS through longer CoT traces or multiple rollouts [12], typically treating the visual context as fixed. This leaves a key gap: *how should a VLM adaptively allocate resources between VCS and VRS for each* (I, Q)*?* We cast this as a compute-allocation problem. Given a multimodal query workload and a configuration θ that controls both visual context size and reasoning search, we characterize the resulting trade-off surface and show how to exploit it to improve accuracy while minimizing compute.

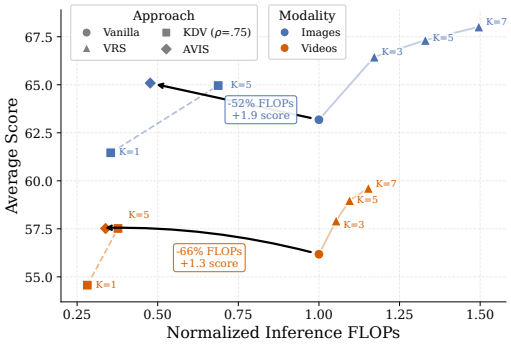


Figure 1: **Accuracy–compute trade-off on Qwen2.5-VL-7B.** We report average score over 12 image benchmarks (blue) and 6 video benchmarks (red), with inference FLOPs normalized to Vanilla (ρ=0, K=1). The plot compares Vanilla, VRS with fixed rollout budgets, KDV with visual pruning, and AVIS with adaptive compute allocation. AVIS improves over Vanilla while substantially reducing FLOPs, achieving a better accuracy–compute trade-off by reallocating compute from redundant visual context to additional rollouts only when needed.

We present **Adaptive Visual Inference Scaling (AVIS)**, a lightweight, sample-dependent policy for test-time compute allocation in VLMsf. Given a target workload, AVIS predicts a per-sample inference configuration over both VCS and VRS. Intuitively, it decides how much visual evidence to retain and how much inference-time reasoning search to perform to improve accuracy without unnecessary computation. We formalize this as a compute-allocation problem over the two inference axes introduced above. In AVIS, VCS is implemented through visual token pruning before prefilling, while VRS is implemented through selfconsistency over K chain-of-thought rollouts.

Based on this instantiation, AVIS adopts a two-stage policy for adaptive compute allocation. To reduce VCS, we introduce **Key Diversity Visual (KDV)** pruning, a training-free visual token selection rule applied before prefilling that uses diversity in attention-key representations as a redundancy signal. KDV runs in $O(N)$ time and requires no additional training. For VRS, AVIS uses self-consistency with majority voting and a difficulty predictor. The predictor allocates the largest rollout counts to an intermediate hard-but-solvable questions, while assigning fewer rollouts to samples at either easy and hard extreme. These choices interact naturally with shared-prefill inference: pruning lowers the one-time prefilling cost, and the resulting KV cache can be reused across all K rollouts, making additional reasoning search substantially cheaper. In this way, AVIS reallocates compute from redundant visual context to useful reasoning search while keeping inference cost low.

We evaluate AVIS across a wide range of visual reasoning tasks, spanning mathematical, visual reasoning, and general VQA benchmarks on both images and videos. We compare against VCS-only and VRS-only baselines, reporting accuracy alongside inference compute measured in FLOPs. Figure 1 shows how our method improves the model performance while also reducing its compute. We further show that shared-prefill makes parallel search efficient: on modern inference engines, AVIS scales reasoning and improves performance while keeping both FLOPs and latency below the corresponding baseline. We also conduct extensive ablations to characterize the VCS–VRS trade-off surface and analyze how AVIS interacts with RL-post-trained models. For consistency, we use Qwen2.5-VL [1] as our primary backbone due to its popularity and strong open-source ecosystem. Across settings, AVIS's sample-dependent allocation consistently improves accuracy at fixed or reduced compute, capturing much of the benefit of test-time scaling without its full cost.

Our main contributions are as follows:

- We introduce a unified view of VLM inference along two coupled axes, Visual Context Scaling (VCS) and Visual Reasoning Scaling (VRS), and characterize their interaction and trade-offs.
- We propose AVIS, a lightweight policy that adaptively prunes redundant visual context and scales reasoning search, enabling sample-dependent compute allocation.

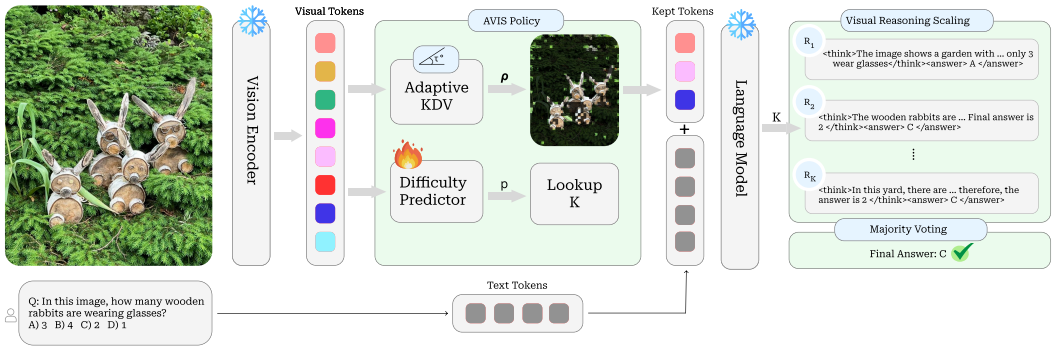


Figure 2: **Adaptive Visual Inference Scaling (AVIS).** Given a query $x = (I, Q)$, we first apply an adaptive, training-free key-diversity pruning rule to remove redundant visual tokens and produce a compact visual prefix. We then run self-consistency with K shared-prefill rollouts and aggregate answers by majority vote. A lightweight difficulty predictor estimates per-sample difficulty and maps it to an appropriate rollout count K, enabling adaptive compute allocation across the workload.

– We introduce KDV, a training-free $O(N)$ visual token pruning rule for VCS, together with a lightweight difficulty predictor for VRS.

2 Related Work

Reasoning VLMs. VLMs have improved dramatically, delivering strong results across a broad set of visual and multimodal tasks [13, 14, 15, 2, 1, 3]. These gains have been enabled by larger and better-aligned LLM backbones, stronger vision encoders with improved resolution/temporal handling, higher-quality data, and reasoning-focused post-training [5, 16, 17, 18, 19, 20, 21, 22]. Despite rapid progress in VLM training, *test-time* compute allocation for visual reasoning remains underexplored, especially how to trade off compute spent on visual context versus reasoning-time search.

Visual Reasoning Scaling (VRS). VRS improves reasoning by allocating additional inference-time compute to search, either through *parallel* sampling and aggregation, such as Best-of-N, self-consistency, and majority voting [9, 23, 24, 25], or through *sequential* refinement, such as iterative revision, resampling, and reflection [26, 27, 28]. This direction has been extensively studied for LLMs, including compute-aware sampling and aggregation policies [29, 30, 31, 32, 33]. For VLMs, recent work has explored RL-based tuning to enable models to allocate CoT length more adaptively based on task difficulty [34, 35, 36, 37]. Other studies report consistent Best-of-N and majority-vote gains on multimodal benchmarks across model families and post-training recipes [38, 12, 39, 40]. However, these approaches typically treat the visual context as fixed, rather than jointly optimizing *seeing* through visual context allocation and *thinking* through reasoning-time search.

Visual Context Scaling (VCS). A key efficiency bottleneck in VLM inference is visual context length: high-resolution images, multi-crop inputs, and video frames produce long visual token sequences that inflate prefilling FLOPs and KV-cache memory. Accordingly, many methods reduce visual tokens/frames before or during prefilling, varying by *where* pruning occurs [41] (ViT-only, early-to-late LLM layers, or hybrid) and by *what signal* is used: (i) *importance/attention-based* heuristics [10, 42, 43, 44, 45] or (ii) *redundancy/similarity-based* grouping [46, 47, 41, 48, 49, 11, 50, 51]. Pruning signals interact with deployment constraints: optimized kernels (e.g., FlashAttention [52]) do not materialize full attention matrices, so attention-based methods that reconstruct attention at selected layers [44, 53] can add substantial overhead and may trigger out-of-memory for long contexts (e.g., videos). Similarity/graph-based approaches can also be costly due to iterative $O(N^2)$ time and memory [47, 41]. While any of these methods could plug into our pipeline, we opt for a simpler key-diversity proxy inspired by KeyDiff [54], which links lower pairwise cosine similarity among keys to higher attention scores and proposes an $O(N)$ eviction rule; we adapt and improve this technique for visual token pruning. Finally, VCS is often studied under single-pass decoding with heuristic prune rates; we extend it to joint, sample-dependent search over both VCS and reasoning-time search.

3 Problem Setup

We consider a multimodal query $x = (I, Q)$, where I is an image or video and Q is a textual prompt. A VLM encodes I into visual embeddings $E_v \in \mathbb{R}^{n_v \times d}$ and tokenizes Q into text embeddings

$E_q \in \mathbb{R}^{n_q \times d}$. Conditioned on this multimodal prefix, the language model generates $y \sim P_\phi(\cdot \mid I, Q)$, which may include reasoning tokens and a final answer $a(y)$.

At test time, inference compute is spent in three places: (i) the vision encoder forward pass, (ii) a *prefill* pass over the multimodal prefix that builds the KV cache, and (iii) autoregressive decoding. We focus on two sample-dependent inference levers that govern this cost: VCS, which controls how much visual evidence is passed to the language model, and VRS, which controls how much inference-time reasoning search is performed. For each query x, we represent an inference configuration by $\theta = (\rho, K)$, where $\rho \in (0, 1]$ denotes the pruned fraction of visual tokens and K the number of reasoning trajectories sampled at decoding time. In general, increasing ρ lowers prefilling cost and KV-cache usage, while increasing K raises decoding cost.

3.1 Adaptive Compute Objective

Given a target workload $\mathcal{D}_{\text{target}} = \{(x_i, y_i^\star)\}_{i=1}^D$, where $x_i = (I_i, Q_i)$, let B_{base} denote the total inference compute of a baseline VLM evaluated with no visual pruning and $K = 1$. Our goal is to improve accuracy by reallocating compute across examples through VCS and VRS, while keeping the total expected compute below this baseline.

For a query x_i, let $\hat{a}(x_i; \theta_i)$ denote the prediction under inference configuration $\theta_i = (\rho_i, K_i)$, and let $F(x_i; \theta_i)$ denote the corresponding inference compute. We seek a *sample-dependent* allocation policy π that maps each query to a configuration ($\theta_i = \pi(x_i)$) to maximize accuracy while having lower compute than that of the baseline:

$$\max_{\pi} \sum_{i=1}^{D} \mathbb{E}[\mathbb{1}(\hat{a}(x_i; \pi(x_i)) = y_i^\star)] \quad \text{s.t.} \quad \sum_{i=1}^{D} \mathbb{E}[F(x_i; \pi(x_i))] \leq B_{\text{base}}. \tag{1}$$

Under mild assumptions on the scaling of prefilling and decoding cost in the shared-prefill regime, the per-example inference compute under configuration $\theta_i = (\rho_i, K_i)$ can be approximated as[2]

$$F(x_i; \theta_i) \approx F_v(I_i) + C_{\mathcal{M}}\big(\rho_i n_{v,i} + n_{q,i} + K_i T_i\big), \tag{2}$$

where $C_{\mathcal{M}}$ is an LLM-dependent constant denoting the average FLOPs incurred by the language model stack per processed token, and T_i is the number of decoded tokens under the vanilla baseline with no pruning and $K = 1$. Since $F_v(I_i)$ is fixed across all inference configurations, Eq. (2) suggests that the total compute allocated to a query can be linearly estimated and controlled through two quantities: the retained prefill tokens and the total decoded tokens.

3.2 Visual Context Scaling

VCS refers to how much visual evidence is provided to the language model. In general, VCS could be controlled through token compression, pooling, or pruning. In AVIS, we instantiate VCS through visual token pruning before prefilling, i.e., by selecting a subset of visual tokens from E_v and passing only those retained tokens to the language model. This choice directly reduces multimodal prefix length, lowering prefilling FLOPs and KV-cache memory.

To implement this efficiently, we introduce **Key Diversity Visual (KDV)** pruning, a training-free key-based scoring rule inspired by KeyDiff [54], originally proposed for KV-cache eviction in LLMs. KDV scores each visual token using diversity in its attention-key representation, favoring tokens whose keys are less aligned with the average key direction and thus less redundant. To better capture multi-head structure, we compute a head-aware score

$$r_i = -\frac{1}{H} \sum_{h=1}^{H} \text{CosSim}\Big(\mu(\hat{K}_h), \hat{k}_{h,i}\Big), \tag{3}$$

where $\hat{k}_{h,i}$ is the normalized key of token i at head h, and $\mu(\hat{K}_h)$ is the mean normalized key for that head. We then retain the top-ranked tokens according to r_i, yielding a retained fraction ρ. KDV is training-free, runs in $O(n_v)$ time up to the top-k selection step, and is compatible with optimized attention kernels. KDV prunes visual tokens before the language-model. This

[2]The derivation and empirical validation are deferred to the appendix.

placement is particularly well suited to AVIS: the shared prefill can be reused across all reasoning rollouts. Moreover, recent work suggests that pruning before the LLM can outperform in-LLM pruning [49, 48], which is consistent with our empirical findings. Together, these choices make KDV a lightweight, practical, and effective VCS mechanism for adaptive multimodal inference.

3.3 Visual Reasoning Scaling

VRS refers to the amount of inference-time reasoning search performed for a query. In principle, this can be increased in several ways, such as longer chain-of-thought traces, iterative refinement, or multiple sampled rollouts. In AVIS, we instantiate VRS through *self-consistency* with majority voting over K independently sampled chain-of-thought trajectories. Larger K expands the search over reasoning paths, but also increases decoding compute.

A key practical advantage of this choice is its compatibility with shared-prefill inference: all K rollouts reuse the same visual features and prefilling KV cache, so increasing K mainly adds decoding-side cost rather than repeating the full multimodal forward pass. In the next section, AVIS uses a lightweight difficulty-aware predictor to choose K adaptively for each query.

Given our adaptive compute allocation objective and the two concrete algorithms that control VCS and VRS, we now introduce **AVIS**, a lightweight policy that approximately optimizes this objective.

4 Adaptive Visual Inference Scaling (AVIS)

AVIS employs a two-stage approach to solve (1): (i) an adaptive visual-token selection rule that reduces redundant visual context, and (ii) a difficulty-aware rule that allocates reasoning rollouts via self-consistency. The overall architecture is shown in Figure 2.

4.1 Adaptive Visual Context Scaling

Adaptive KDV. Recall the KDV retention score r_i (3), which is the (negative) mean cosine alignment of token i to the per-head anchor directions. Instead of selecting a fixed top-N set, we obtain an *adaptive* retained set by thresholding key–anchor angular separation. For a hyperparameter $\tau \in [0, \pi]$, we retain tokens whose average angle from the anchor exceeds τ, i.e.,

$$S_\tau(x) \;=\; \big\{ i \in \{1, \dots, n_v\} \;:\; r_i \geq -\cos(\tau) \big\}. \tag{4}$$

This yields a sample-dependent retained fraction $\rho(x) = |S_\tau(x)|/n_v$ at essentially the same cost as KDV scoring (linear in n_v, up to thresholding).

4.2 Adaptive Visual Reasoning Scaling

Self-consistency draws K i.i.d. reasoning trajectories and aggregates their extracted answers via majority voting. Let $p(x)$ denote the probability that a *single* trajectory produces the correct answer. Under the standard i.i.d. model,[3] the probability that self-consistency is correct is

$$P_{\mathrm{SC}}(p, K) \;=\; \sum_{j=\lceil (K+1)/2 \rceil}^{K} \binom{K}{j} p^j (1-p)^{K-j}. \tag{5}$$

This highlights two regimes: when $p(x)$ is already close to 1, additional trajectories yield diminishing returns; when $p(x)<0.5$, more trajectories may even degrade accuracy. Our policy, therefore, allocates a larger K for queries that are likely solvable but not trivial.

Learned difficulty predictor. We train a lightweight head that predicts, for each query x, a *solvability score* $\hat{p}(x) \in [0, 1]$ from the visual embeddings E_v. The predictor and the rollout-selection rule play distinct roles: the predictor learns an input-dependent signal for whether additional reasoning is likely to help, while the subsequent binning rule is a calibrated decision layer that converts this signal into a discrete rollout budget. This distinction is important because the benefit of self-consistency is not monotonic in difficulty. From Equation 5, increasing K has little marginal value when $p(x)$ is already

[3] For multi-class answers, self-consistency returns the *plurality* winner (largest vote count), not necessarily a strict majority. Eq. (5) corresponds to the binary/"correct vs. incorrect" view and serves as a useful approximation for understanding how accuracy scales with K.

high *(very easy)*, since a single trajectory is likely correct, and can also be ineffective when $p(x)$ is very low *(very unlikely to be solvable)*, since majority voting is unlikely to recover the correct answer. The largest expected gain, therefore, occurs for intermediate, *hard-but-solvable* examples. AVIS uses this observation to allocate additional rollouts only to inputs with high predicted marginal utility, rather than simply assigning more compute to all difficult examples.

To obtain supervision, we construct a representative calibration set and run the base VLM with $K_{\max} = 10$ rollouts for each example. We estimate its empirical solvability as $p(x_j) = \frac{1}{K_{\max}} \sum_{k=1}^{K_{\max}} \mathbb{1}\left[\hat{y}^{(k)}(x_j) = y_j^\star\right]$, and binarize this value with threshold 0.5 to indicate whether the example is solvable under strong self-consistency. The predictor takes the visual embeddings as input, applies a short stack of 1D Conv–GroupNorm–SiLU layers along the token dimension, followed by global average pooling and a shallow MLP, and outputs logits $s(x_j) \in \mathbb{R}^2$. At test time, we interpret $\hat{p}(x_j) = \mathrm{softmax}(s(x_j))_1$ as the predicted solvability score. Finally, because the deployment action space is discrete, $K \in \{1, 3, 5, 7\}$, we map $\hat{p}(x)$ to a rollout count using a calibrated piecewise-constant budget policy:

$\hat{p}(x)$	$[0, b_1)$	$[b_1, b_2)$	$[b_2, b_3)$	$[b_3, b_4)$	$[b_4, b_5)$	$[b_5, 1]$
K	1	5	7	5	3	1

The bin boundaries $\{b_j\}_{j=1}^5$ are selected once on the calibration set to satisfy the global compute budget and are then fixed for all evaluations. Thus, the binning rule is not intended as a learned model by itself; it is a deployment-friendly action-selection layer over a learned solvability predictor, designed to concentrate reasoning compute on the examples where self-consistency is expected to provide the highest return. The non-monotonic shape follows the inverted-U marginal utility of majority voting Equation 5: rollouts help most for intermediate $\hat{p}(x)$ and are wasted at either extreme.

Calibration set. We construct a calibration set $\mathcal{D}_{\mathrm{cal}} = \{(x_j, y_j^\star)\}_{j=1}^M$ representative of the deployment workload. Each example $x_j = (I_j, Q_j)$ is a *multi-choice* query with a ground-truth answer $y_j^\star$. Following S1 guidelines [29], we curate $\mathcal{D}_{cal}$ using three principles: **Quality**, **Diversity**, and **Difficulty**. Starting with a set of 73,139 images and videos we arrive at 5000 multi-choice questions (4000 images, 1000 videos) to train the policy. Details of the curation process are provided in §Appendix B. Next, for each $(x_j, y_j^\star)$, we run the base VLM with $K_{\max}$=10 rollouts, $p(x_j)$ is then calculated as $\frac{1}{K_{\max}} \sum_{k=1}^{K_{\max}} \mathbb{1}\big(\hat{y}^{(k)}(x_j) = y_j^\star\big)$. We then binarize this signal using a fixed threshold 0.5 to obtain a supervision label $y_j = \mathbb{1}\big(p_j \geq 0.5\big)$, which indicates whether x_j is "solvable" under strong self-consistency.

During the training, the predictor takes as input the sequence of visual token embeddings, processes them with a short stack of 1D Conv–GroupNorm–SiLU layers along the token dimension, followed by global average pooling, and feeds the resulting vector into a shallow MLP producing logits $s(x_j) \in \mathbb{R}^2$. At test time, we interpret $\hat{p}(x_j) = \mathrm{softmax}\big(s(x_j)\big)_1$ as the predicted solvability probability and deterministically map it to a rollout count $K \in \{1, 3, 5, 7\}$ via the binning rule.

5 Experiments

5.1 Experimental Setup

We evaluate the effectiveness and efficiency of our adaptive visual reasoning policy, AVIS, on image and video benchmarks. We use Qwen2.5-VL-7B [1] as the backbone and run all methods in CoT mode using the `<think> </think>` and `<answer> </answer>` format [4]. For AVIS, we use the pruning threshold $\tau = \pi/4$, which yields an average pruning ratio of $\approx 75\%$; our analysis (Figure 3) shows this choice preserves key visual evidence while providing substantial compute savings. When K=1 we use greedy decoding; when K>1 we sample with temperature 0.7 and top-p 0.9 and aggregate by majority vote. We evaluate image and video benchmarks using VLMEvalKit [55].

Baselines. We compare against a *Vanilla* baseline (K=1, no pruning) and fixed VRS-VCS setups. For pruning baselines, we choose a fixed pruning ratio ρ=0.75 which is motivated by our finding in Figure 3 that this pruning level offers a favorable accuracy–efficiency trade-off.

Image benchmarks. We evaluate on a diverse set of multimodal benchmarks. For images our benchmarks span math reasoning (MathVista [56], MathVerse [57], MathVision [58]), OCR (DOCVQA [59]), multi-discipline reasoning (MMMU-Pro [60]), and general VQA (MME [61], MMStar [62], MMBench [63], CVBench-2D [64], POPE [65], BLINK [66], TreeBench [67]).

Table 1: **Performance across inference strategies.** We compare VRS-only scaling, KDV-based VCS pruning, fixed joint VCS–VRS configurations, and AVIS, which adaptively selects both ρ and K at test time. AVIS achieves the best overall compute–performance trade-off across 12 image and 6 video benchmarks. #F denotes FLOPs relative to Vanilla (ρ=0, K=1); lower #F is better.

Category	Benchmark	Vanilla		VRS-Only						KDV-Prune				AVIS	
		ρ=0, K=1		ρ=0, K=3		ρ=0, K=5		ρ=0, K=7		ρ=75%, K=1		ρ=75%, K=5		Adaptive	
		Score	#F	Score	#F	Score	#F	Score	#F	Score	#F	Score	#F	Score	#F
Math	MathVista [56]	67.5	1.0	68.7	1.23	70.5	1.45	70.2	1.67	66.6	0.38	68.9	0.83	68.1	0.46
	MathVerse [57]	57.5	1.0	59.1	1.21	62.9	1.42	63.9	1.67	55.7	0.39	59.5	0.82	58.9	0.45
	MathVision [58]	22.4	1.0	29.9	1.35	27.0	1.71	31.6	2.04	22.3	0.45	25.4	1.04	25.0	0.78
Image VQA	DocVQA [59]	81.43	1.0	88.44	1.03	89.71	1.06	90.73	1.09	76.79	0.27	83.97	0.36	87.71	0.32
	MMMU-Pro [60]	42.62	1.0	43.66	1.18	45.68	1.30	46.23	1.46	40.54	0.32	43.95	0.88	43.10	0.68
	MME [61]	2263	1.0	2396	1.13	2386	1.26	2397	1.39	2227	0.33	2373	0.58	2364	0.42
	MMStar [62]	62.0	1.0	63.6	1.20	65.8	1.40	66.4	1.61	59.0	0.38	61.5	0.79	61.6	0.52
	MMBench [63]	81.4	1.0	87.8	1.16	88.3	1.32	88.6	1.48	79.2	0.36	86.8	0.68	86.4	0.44
	CVBench [64]	70.6	1.0	73.9	1.11	74.9	1.22	75.5	1.42	70.5	0.33	74.3	0.55	73.1	0.43
	POPE [65]	84.3	1.0	84.6	1.10	84.9	1.19	84.8	1.28	83.9	0.32	84.7	0.50	84.5	0.38
	BLINK [66]	56.8	1.0	56.9	1.15	58.2	1.28	57.4	1.42	51.9	0.34	55.1	0.58	56.7	0.42
	TreeBench [67]	37.3	1.0	41.0	1.21	40.5	1.35	41.0	1.42	38.3	0.38	36.5	0.64	37.5	0.42
Video VQA	Video-MME [68]	65.2	1.0	66.4	1.08	70.0	1.12	70.2	1.24	62.8	0.28	66.0	0.36	65.8	0.32
	TempCompass [69]	73.6	1.0	77.8	1.02	79.0	1.04	78.6	1.06	72.4	0.26	75.6	0.30	76.2	0.28
	Video-TT [70]	35.2	1.0	37.6	1.02	36.8	1.04	38.6	1.05	36.4	0.27	38.4	0.30	37.6	0.29
	MVBench [71]	56.2	1.0	57.6	1.09	59.8	1.17	60.0	1.27	54.5	0.31	57.4	0.49	57.8	0.46
	Q-Bench-Video [72]	60.5	1.0	59.9	1.06	60.5	1.12	62.2	1.18	58.4	0.29	61.3	0.42	60.9	0.38
	Video-MMMU [73]	46.4	1.0	48.2	1.05	47.8	1.08	48.1	1.12	42.9	0.28	46.4	0.33	46.8	0.30

Video benchmarks. For video understanding and reasoning, we evaluate on VideoMME [68], TempCompass [69], Video-TT [70], MVBench [71], Q-Bench-Video [72], and Video-MMMU [73].

Compute measurement. We compare efficiency using FLOPs normalized by the *Vanilla* setting. Since all configurations share the same vision encoder, differences are mainly driven by language-model prefill and decoding. We also report wall-clock latency in §5.3.

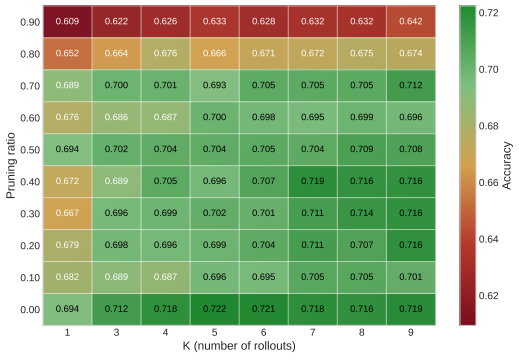


Figure 3: **VCS-VRS Trade-off.** Accuracy heatmap over pruning ratio ρ and rollout count K on the calibration dataset. Each cell shows the resulting accuracy, with lighter colors indicating better performance. Accuracy improves as K increases under varying pruning ratios.

Table 2: **Matched-FLOPs comparison.** We compare AVIS against fixed configurations chosen to approximate its average compute budget ($\sim$45% of Vanilla FLOPs). Under comparable FLOPs, AVIS consistently outperforms fixed VCS–VRS policies, showing the benefit of input-adaptive allocation. #F denotes FLOPs relative to Vanilla (ρ=0, K=1).

Benchmark	Vanilla		Fixed				AVIS	
	ρ=0, K=1		ρ=0.6, K=1		ρ=0.85, K=5		Adaptive	
	Score	#F	Score	#F	Score	#F	Score	#F
MathVista	67.5	1.0	66.6	0.48	65.2	0.45	68.1	0.46
MathVerse	57.5	1.0	56.5	0.44	56.4	0.48	58.9	0.45
MME	2263	1.0	2230	0.43	2311	0.48	2364	0.42
CVBench	70.6	1.0	71.2	0.40	71.4	0.47	73.1	0.43
BLINK	56.8	1.0	52.1	0.45	54.5	0.48	56.7	0.42
TreeBench	37.3	1.0	36.8	0.39	35.1	0.43	37.5	0.42

5.2 Main Results

Table 1 compares AVIS against a broad set of inference policies, including the *Vanilla* baseline, VRS-only and VCS-only variants, and fixed joint VCS–VRS configurations. Figure 1 summarizes the corresponding Pareto trade-off between performance and compute, while Table 2 provides an iso-compute comparison across policies.

Findings. Tables 1 and 2 highlight three main trends:

– As expected, increasing VRS improves accuracy at the cost of higher compute, whereas aggressive VCS reduction degrades performance when applied in isolation. Coupling visual pruning with reasoning scaling yields a better operating point: even fixed joint configurations can outperform the base model while using fewer FLOPs than the baseline.

Table 4: **AVIS on RL post-trained VLMs.** We compare each checkpoint under Vanilla, fixed joint scaling (ρ=75%, K=5), and AVIS with adaptive ρ and K. Across 6 image benchmarks, AVIS preserves or improves accuracy with substantially lower FLOPs, showing that adaptive VCS–VRS allocation transfers to RL post-trained VLMs.

Checkpoint	Method	Setting		MathVista		MathVision		MME		MMStar		CVBench		POPE	
		ρ	K	Score	#F	Score	#F	Score	#F	Score	#F	Score	#F	Score	#F
Qwen 2.5 VL [1]	**Baseline**	0%	1	67.5	1.0	22.4	1.0	2263	1.0	62.0	1.0	70.6	1.0	84.3	1.0
	Fixed	75%	5	68.9	0.83	25.4	1.04	2373	0.58	61.5	0.79	74.3	0.55	84.7	0.50
	AVIS	Adaptive		68.1	0.46	25.0	0.78	2364	0.42	61.6	0.52	73.1	0.43	84.5	0.38
VL-Rethinker [18]	**Baseline**	0%	1	72.2	1.0	33.2	1.0	2328	1.0	64.6	1.0	76.9	1.0	82.9	1.0
	Fixed	75%	5	69.9	0.88	36.5	1.25	2402	0.61	59.2	0.84	73.6	0.64	82.0	0.52
	AVIS	Adaptive		71.8	0.43	37.3	0.82	2463	0.46	61.3	0.54	76.1	0.42	84.7	0.40
Vision-R1 [5]	**Baseline**	0%	1	68.5	1.0	39.5	1.0	2408	1.0	63.4	1.0	72.7	1.0	88.5	1.0
	Fixed	75%	5	69.3	0.81	40.8	1.20	2436	0.63	61.5	0.75	73.4	0.56	88.9	0.57
	AVIS	Adaptive		69.6	0.39	41.1	0.78	2469	0.43	65.1	0.49	74.4	0.43	89.1	0.40
OpenVLThinker [75]	**Baseline**	0%	1	68.7	1.0	27.9	1.0	2291	1.0	65.0	1.0	72.9	1.0	82.6	1.0
	Fixed	75%	5	69.8	0.79	28.6	1.05	2435	0.55	63.4	0.72	74.2	0.50	83.2	0.48
	AVIS	Adaptive		70.6	0.41	28.4	0.77	2437	0.39	64.2	0.48	74.8	0.39	84.5	0.34

– AVIS consistently provides a stronger compute–performance trade-off. Compared to the *Vanilla* baseline, it improves mean accuracy on image and video benchmarks by 3% and 2.4%, respectively, while reducing compute by 52% and 66%. Compared to the closest fixed baseline (ρ=0.75, K=5), AVIS achieves higher accuracy with 40% lower compute.

– Under approximately matched FLOPs, AVIS outperforms the closest competing policy by 3.7%.

5.3 Latency

We also measure wall-clock latency to validate the benefits of combining visual context reduction with self-consistency. Table 3 reports decoding time for 100 image–question pairs under shared prefilling. The Vanilla baseline (ρ=0, K=1) takes **795 s**. Among fixed configurations, increasing rollouts (ρ=0%, K=5) slightly exceeds the baseline at **810 s**, while KDV pruning (ρ=75%, K=1) reduces latency to **558 s**. Combining both axes (ρ=75%, K=5) brings latency back near baseline at **789 s**, while achieving higher accuracy with fewer visual tokens. AVIS achieves **757 s**, confirming that adaptive VCS–VRS yields tangible latency benefits in shared-prefill setups.

Table 3: **Latency Comparison.** Decoding latency for 100 image-question pairs with shared prefilling. Measurements are obtained on a single NVIDIA L40 (48 GB) GPU using vLLM v0.10.0 [74].

Method	Setting		Total time (sec.)	Reduction ratio
	ρ	K		
Vanilla	0	1	795	1.0
Fixed	0%	5	810	0.98
	75%	1	558	1.42
	75%	5	789	1.01
AVIS	Adaptive		757	1.05

5.4 RL Post-trained VLMs

Recent VLM work improves multimodal reasoning through GRPO-style RL post-training [4, 5, 18], reporting gains across image and video reasoning benchmarks. In Table 4, we evaluate three well-regarded RL post-trained models under a fixed VCS–VRS setting of (ρ=75%, K=5) as well as AVIS, and include the baseline Qwen2.5-VL results for reference. Across models and benchmarks, AVIS preserves or improves accuracy while substantially reducing FLOPs. Overall, these results show that the benefits from our approach extend well to RL post-trained VLMs.

5.5 Ablation Studies

Ablating KDV. KDV adapts the retained visual context to each input, preserving tokens around salient regions while removing large, low-information background areas. This behavior supports the premise that substantial visual pruning is possible without discarding evidence needed for downstream reasoning. Quantitatively, KDV also compares favorably against other training-free pruning baselines at a fixed 75% pruning rate, achieving the best average performance and retaining 97.99% of the *Vanilla* accuracy across eight benchmarks. We provide qualitative visualizations of adaptive pruning in Figure 5 and the full pruning comparison in Appendix C.3.

Table 5: **Adaptive vs. fixed allocation per axis.** We compare one-axis adaptive policies against AVIS, which jointly adapts visual context ρ and reasoning budget K. Joint adaptation achieves the strongest compute–performance trade-off, showing that VCS and VRS are complementary.

Benchmark	Vanilla		Fixed		One-Axis Adaptive				AVIS	
	ρ=0, K=1		ρ=75%, K=5		Adapt-ρ, K=5		ρ=75%, Adapt-K		Adaptive	
	Score	#F	Score	#F	Score	#F	Score	#F	Score	#F
MathVision	22.4	1.0	22.3	1.04	24.9	1.05	25.3	0.83	25.0	0.78
MME	2263	1.0	2373	0.58	2347	0.52	2280	0.41	2364	0.42
DocVQA	81.4	1.0	83.97	0.36	84.7	0.35	86.2	0.31	87.7	0.32
CVBench	70.6	1.0	74.3	0.55	73.8	0.53	73.2	0.49	73.1	0.43
POPE	84.3	1.0	84.7	0.50	84.8	0.49	84.1	0.41	84.5	0.38

Ablating the difficulty predictor. We analyze the learned VRS controller to verify that AVIS does not simply increase reasoning compute uniformly. As shown in Figure 4 in the appendix, the selected rollout distribution is strongly dataset-dependent, with average rollout counts of 2.12 on MMBench, 2.53 on MME, and 3.79 on Video-TT. This suggests that the predictor learns non-trivial allocation patterns, assigning $K = 1$ to many easy or low-gain examples while reserving larger budgets where self-consistency is more likely to help. We also compare alternative difficulty predictor architectures in Table 7. Compared with Perceiver [76] and Transformer-style predictors, our predictor achieves the highest held-out accuracy (79.2%) and uses fewer rollouts at comparable downstream accuracy. For example, on MMBench, it reduces the average rollout count from 3.25 to 2.12 relative to the Perceiver predictor, lowering FLOPs from $0.59\times$ to $0.44\times$ while maintaining similar accuracy. These results support the difficulty predictor as an effective compute-allocation mechanism: sharper solvability estimates let AVIS avoid unnecessary rollouts while preserving most self-consistency gains. Full rollout-allocation and predictor-architecture results are provided in Appendix C.1 and C.2.

Adaptive vs. fixed allocation per axis. We also evaluate mixed policies that adapt only one axis while keeping the other fixed, as shown in Table 5. These ablations isolate the effects of adaptive visual context allocation and adaptive reasoning allocation. Adaptive ρ improves over fixed KDV pruning on several benchmarks, showing that input-dependent pruning retains more useful visual evidence than a global pruning ratio. Adaptive K similarly improves the compute–performance trade-off by concentrating rollouts on examples where self-consistency is most beneficial. Combining the two yields the strongest overall policy: on DocVQA, AVIS reaches 87.7, exceeding both Adapt-ρ, K=5 (84.7) and ρ=75%, Adapt-K (86.2), with comparable or lower compute. These results show that visual context and reasoning budget are best adapted jointly rather than independently.

VCS–VRS trade-off. In Figure 3, we sweep the pruning ratio ρ from 0 to 0.9 and the rollout count K from 1 to 9 (except $K = 2$) on the calibration set and plot the resulting accuracy. Along each row (fixed ρ), accuracy generally increases as K grows. Along each column (fixed K), moderate pruning ($\rho \approx 0.2$–0.7) keeps the model competitive with, or slightly better than the no-pruning case, indicating the high redundancy and the effectiveness of our KDV. Only in the extreme regime ($\rho \geq 0.8$) do we observe clear accuracy degradation, showing that KDV can safely remove a large fraction of visual tokens before performance begins to drop.

6 Conclusion and discussion

We studied adaptive visual reasoning in VLM inference through the joint allocation of visual context and reasoning compute. We introduced AVIS, a deployment-friendly policy that prunes visual context with lightweight KDV and reallocates the saved compute to parallel self-consistency via a difficulty-aware rollout selector. Across image and video benchmarks, AVIS improves the accuracy–compute trade-off over fixed configurations, showing that visual context and reasoning effort should be treated as coupled inference-time resources rather than independent design choices.

AVIS is a first instantiation of adaptive VCS–VRS allocation. Predicting per-sample difficulty is itself a challenging problem, which we do not claim to solve fully. Instead, following recent adaptive-compute work for LLMs [32, 33], we approximate difficulty using a coarse binary solvability signal and a lightweight head over visual embeddings. Although deliberately simple, this design already learns a meaningful allocation signal, as shown in our ablation studies. Two approximations are particularly worth revisiting. First, the predictor uses only visual embeddings: incorporating the question text could sharpen difficulty estimates, but using later LLM representations would require running part of the LM prefill before deciding K, eroding the compute savings that motivate

our pre-prefill design. Second, AVIS treats visual pruning and reasoning allocation as largely decoupled decisions, even though the retained visual context can directly affect both sample difficulty and the marginal value of additional reasoning. Future work can explore unified, text-aware, and budget-aware policies that jointly select visual context and reasoning effort, directly optimize the compute-constrained objective, and adapt to distribution shifts in deployment workloads.

References

[1] Shuai Bai, Keqin Chen, Xuejing Liu, Jialin Wang, Wenbin Ge, Sibo Song, Kai Dang, Peng Wang, Shijie Wang, Jun Tang, Humen Zhong, Yuanzhi Zhu, Mingkun Yang, Zhaohai Li, Jianqiang Wan, Pengfei Wang, Wei Ding, Zheren Fu, Yiheng Xu, Jiabo Ye, Xi Zhang, Tianbao Xie, Zesen Cheng, Hang Zhang, Zhibo Yang, Haiyang Xu, and Junyang Lin. Qwen2.5-VL Technical Report. *arXiv preprint arXiv:2502.13923*, 2025.

[2] Bo Li, Yuanhan Zhang, Dong Guo, Renrui Zhang, Feng Li, Hao Zhang, Kaichen Zhang, Yanwei Li, Ziwei Liu, and Chunyuan Li. LLaVA-OneVision: Easy Visual Task Transfer. *Transactions on Machine Learning Research*, 2025.

[3] Jinguo Zhu, Weiyun Wang, Zhe Chen, Zhaoyang Liu, Shenglong Ye, Lixin Gu, Hao Tian, Yuchen Duan, Weijie Su, Jie Shao, Zhangwei Gao, Erfei Cui, Xuehui Wang, Yue Cao, Yangzhou Liu, Xingguang Wei, Hongjie Zhang, Haomin Wang, Weiye Xu, Hao Li, Jiahao Wang, Nianchen Deng, Songze Li, Yinan He, Tan Jiang, Jiapeng Luo, Yi Wang, Conghui He, Botian Shi, Xingcheng Zhang, Wenqi Shao, Junjun He, Yingtong Xiong, Wenwen Qu, Peng Sun, Penglong Jiao, Han Lv, Lijun Wu, Kaipeng Zhang, Huipeng Deng, Jiaye Ge, Kai Chen, Limin Wang, Min Dou, Lewei Lu, Xizhou Zhu, Tong Lu, Dahua Lin, Yu Qiao, Jifeng Dai, and Wenhai Wang. InternVL3: Exploring Advanced Training and Test-Time Recipes for Open-Source Multimodal Models. *arXiv preprint arXiv:2504.10479*, 2025.

[4] Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Peiyi Wang, Qihao Zhu, Runxin Xu, Ruoyu Zhang, Shirong Ma, Xiao Bi, Xiaokang Zhang, Xingkai Yu, Yu Wu, Z. F. Wu, Zhibin Gou, Zhihong Shao, Zhuoshu Li, Ziyi Gao, Aixin Liu, Bing Xue, Bingxuan Wang, Bochao Wu, Bei Feng, Chengda Lu, Chenggang Zhao, Chengqi Deng, Chong Ruan, Damai Dai, Deli Chen, Dongjie Ji, Erhang Li, Fangyun Lin, Fucong Dai, Fuli Luo, Guangbo Hao, Guanting Chen, Guowei Li, H. Zhang, Hanwei Xu, Honghui Ding, Huazuo Gao, Hui Qu, Hui Li, Jianzhong Guo, Jiashi Li, Jingchang Chen, Jingyang Yuan, Jinhao Tu, Junjie Qiu, Junlong Li, J. L. Cai, Jiaqi Ni, Jian Liang, Jin Chen, Kai Dong, Kai Hu, Kaichao You, Kaige Gao, Kang Guan, Kexin Huang, Kuai Yu, Lean Wang, Lecong Zhang, Liang Zhao, Litong Wang, Liyue Zhang, Lei Xu, Leyi Xia, Mingchuan Zhang, Minghua Zhang, Minghui Tang, Mingxu Zhou, Meng Li, Miaojun Wang, Mingming Li, Ning Tian, Panpan Huang, Peng Zhang, Qiancheng Wang, Qinyu Chen, Qiushi Du, Ruiqi Ge, Ruisong Zhang, Ruizhe Pan, Runji Wang, R. J. Chen, R. L. Jin, Ruyi Chen, Shanghao Lu, Shangyan Zhou, Shanhuang Chen, Shengfeng Ye, Shiyu Wang, Shuiping Yu, Shunfeng Zhou, Shuting Pan, S. S. Li, Shuang Zhou, Shaoqing Wu, Tao Yun, Tian Pei, Tianyu Sun, T. Wang, Wangding Zeng, Wen Liu, Wenfeng Liang, Wenjun Gao, Wenqin Yu, Wentao Zhang, W. L. Xiao, Wei An, Xiaodong Liu, Xiaohan Wang, Xiaokang Chen, Xiaotao Nie, Xin Cheng, Xin Liu, Xin Xie, Xingchao Liu, Xinyu Yang, Xinyuan Li, Xuecheng Su, Xuheng Lin, X. Q. Li, Xiangyue Jin, Xiaojin Shen, Xiaosha Chen, Xiaowen Sun, Xiaoxiang Wang, Xinnan Song, Xinyi Zhou, Xianzu Wang, Xinxia Shan, Y. K. Li, Y. Q. Wang, Y. X. Wei, Yang Zhang, Yanhong Xu, Yao Li, Yao Zhao, Yaofeng Sun, Yaohui Wang, Yi Yu, Yichao Zhang, Yifan Shi, Yiliang Xiong, Ying He, Yishi Piao, Yisong Wang, Yixuan Tan, Yiyang Ma, Yiyuan Liu, Yongqiang Guo, Yuan Ou, Yuduan Wang, Yue Gong, Yuheng Zou, Yujia He, Yunfan Xiong, Yuxiang Luo, Yuxiang You, Yuxuan Liu, Yuyang Zhou, Y. X. Zhu, Yanping Huang, Yaohui Li, Yi Zheng, Yuchen Zhu, Yunxian Ma, Ying Tang, Yukun Zha, Yuting Yan, Z. Z. Ren, Zehui Ren, Zhangli Sha, Zhe Fu, Zhean Xu, Zhenda Xie, Zhengyan Zhang, Zhewen Hao, Zhicheng Ma, Zhigang Yan, Zhiyu Wu, Zihui Gu, Zijia Zhu, Zijun Liu, Zilin Li, Ziwei Xie, Ziyang Song, Zizheng Pan, Zhen Huang, Zhipeng Xu, Zhongyu Zhang, and Zhen Zhang. DeepSeek-R1 Incentivizes Reasoning in LLMs through Reinforcement Learning. *Nature*, 645:633–638, 2025.

[5] Wenxuan Huang, Bohan Jia, Zijie Zhai, Shaosheng Cao, Zheyu Ye, Fei Zhao, Zhe Xu, Yao Hu, and Shaohui Lin. Vision-R1: Incentivizing Reasoning Capability in Multimodal Large Language Models. In *International Conference on Learning Representations (ICLR)*, 2026.

[6] Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Brian Ichter, Fei Xia, Ed Chi, Quoc Le, and Denny Zhou. Chain-of-Thought Prompting Elicits Reasoning in Large Language Models. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2022.

[7] Nisan Stiennon, Long Ouyang, Jeff Wu, Daniel M. Ziegler, Ryan Lowe, Chelsea Voss, Alec Radford, Dario Amodei, and Paul Christiano. Learning to Summarize with Human Feedback. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2020.

[8] Yuki Ichihara, Yuu Jinnai, Tetsuro Morimura, Kaito Ariu, Kenshi Abe, Mitsuki Sakamoto, and Eiji Uchibe. Evaluation of Best-of-N Sampling Strategies for Language Model Alignment. *Transactions on Machine Learning Research*, 2025.

[9] Xuezhi Wang, Jason Wei, Dale Schuurmans, Quoc Le, Ed Chi, Sharan Narang, Aakanksha Chowdhery, and Denny Zhou. Self-Consistency Improves Chain of Thought Reasoning in Language Models. In *International Conference on Learning Representations (ICLR)*, 2023.

[10] 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*, pages 19–35. Springer, 2024.

[11] Yuzhang Shang, Mu Cai, Bingxin Xu, Yong Jae Lee, and Yan Yan. LLaVA-PruMerge: Adaptive Token Reduction for Efficient Large Multimodal Models. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 22857–22867, 2025.

[12] Mohammadjavad Ahmadpour, Amirmahdi Meighani, Payam Taebi, Omid Ghahroodi, Amirmohammad Izadi, and Mahdieh Soleymani Baghshah. Limits and Gains of Test-Time Scaling in Vision-Language Reasoning. *arXiv preprint arXiv:2512.11109*, 2025.

[13] Haotian Liu, Chunyuan Li, Qingyang Wu, and Yong Jae Lee. Visual Instruction Tuning. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2023.

[14] 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*, 2023.

[15] Haotian Liu, Chunyuan Li, Yuheng Li, Bo Li, Yuanhan Zhang, Sheng Shen, and Yong Jae Lee. LLaVA-NeXT: Improved Reasoning, OCR, and World Knowledge, January 2024.

[16] Guowei Xu, Peng Jin, Ziang Wu, Hao Li, Yibing Song, Lichao Sun, and Li Yuan. LLaVA-CoT: Let Vision Language Models Reason Step-by-Step. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 2087–2098, 2025.

[17] Huanjin Yao, Jiaxing Huang, Wenhao Wu, Jingyi Zhang, Yibo Wang, Shunyu Liu, Yingjie Wang, Yuxin Song, Haocheng Feng, Li Shen, and Dacheng Tao. Mulberry: Empowering MLLM with o1-like Reasoning and Reflection via Collective Monte Carlo Tree Search. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2025.

[18] Haozhe Wang, Chao Qu, Zuming Huang, Wei Chu, Fangzhen Lin, and Wenhu Chen. VL-Rethinker: Incentivizing Self-Reflection of Vision-Language Models with Reinforcement Learning. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2025.

[19] Ahmadreza Jeddi, Hakki Can Karaimer, Hue Nguyen, Zhongling Wang, Ke Zhao, Javad Rajabi, Ran Zhang, Raghav Goyal, Babak Taati, and Radek Grzeszczuk. Puzzle Curriculum GRPO for Vision-Centric Reasoning. *arXiv preprint arXiv:2512.14944*, 2025.

[20] Haozhan Shen, Peng Liu, Jingcheng Li, Chunxin Fang, Yibo Ma, Jiajia Liao, Qiaoli Shen, Zilun Zhang, Kangjia Zhao, Qianqian Zhang, Ruochen Xu, and Tiancheng Zhao. VLM-R1: A Stable and Generalizable R1-style Large Vision-Language Model. *arXiv preprint arXiv:2504.07615*, 2025.

[21] Jingyi Zhang, Jiaxing Huang, Huanjin Yao, Shunyu Liu, Xikun Zhang, Shijian Lu, and Dacheng Tao. R1-VL: Learning to Reason with Multimodal Large Language Models via Step-wise Group Relative Policy Optimization. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 2025.

[22] Ahmadreza Jeddi, Kimia Shaban, Negin Baghbanzadeh, Natasha Sharan, Abhishek Moturu, Elham Dolatabadi, and Babak Taati. When Does RL Help Medical VLMs? Disentangling Vision, SFT, and RL Gains. *arXiv preprint arXiv:2603.01301*, 2026.

[23] Mark Chen, Jerry Tworek, Heewoo Jun, Qiming Yuan, Henrique Ponde de Oliveira Pinto, Jared Kaplan, Harri Edwards, Yuri Burda, Nicholas Joseph, Greg Brockman, Alex Ray, Raul Puri, Gretchen Krueger, Michael Petrov, Heidy Khlaaf, Girish Sastry, Pamela Mishkin, Brooke Chan, Scott Gray, Nick Ryder, Mikhail Pavlov, Alethea Power, Lukasz Kaiser, Mohammad Bavarian, Clemens Winter, Philippe Tillet, Felipe Petroski Such, Dave Cummings, Matthias Plappert, Fotios Chantzis, Elizabeth Barnes, Ariel Herbert-Voss, William Hebgen Guss, Alex Nichol, Alex Paino, Nikolas Tezak, Jie Tang, Igor Babuschkin, Suchir Balaji, Shantanu Jain, William Saunders, Christopher Hesse, Andrew N. Carr, Jan Leike, Josh Achiam, Vedant Misra, Evan Morikawa, Alec Radford, Matthew Knight, Miles Brundage, Mira Murati, Katie Mayer, Peter Welinder, Bob McGrew, Dario Amodei, Sam McCandlish, Ilya Sutskever, and Wojciech Zaremba. Evaluating Large Language Models Trained on Code. *arXiv preprint arXiv:2107.03374*, 2021.

[24] Bradley Brown, Jordan Juravsky, Ryan Ehrlich, Ronald Clark, Quoc V Le, Christopher Ré, and Azalia Mirhoseini. Large Language Monkeys: Scaling Inference Compute with Repeated Sampling. *arXiv preprint arXiv:2407.21787*, 2024.

[25] Amin Rakhsha, Kanika Madan, Tianyu Zhang, Amir massoud Farahmand, and Amir Khasahmadi. Majority of the Bests: Improving Best-of-N via Bootstrapping. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2025.

[26] Aman Madaan, Niket Tandon, Prakhar Gupta, Skyler Hallinan, Luyu Gao, Sarah Wiegreffe, Uri Alon, Nouha Dziri, Shrimai Prabhumoye, Yiming Yang, Shashank Gupta, Bodhisattwa Prasad Majumder, Katherine Hermann, Sean Welleck, Amir Yazdanbakhsh, and Peter Clark. Self-Refine: Iterative Refinement with Self-Feedback. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2023.

[27] Hunter Lightman, Vineet Kosaraju, Yuri Burda, Harrison Edwards, Bowen Baker, Teddy Lee, Jan Leike, John Schulman, Ilya Sutskever, and Karl Cobbe. Let's Verify Step by Step. In *The Twelfth International Conference on Learning Representations*, 2023.

[28] Peiyi Wang, Lei Li, Zhihong Shao, Runxin Xu, Damai Dai, Yifei Li, Deli Chen, Yu Wu, and Zhifang Sui. Math-Shepherd: Verify and Reinforce LLMs Step-by-step without Human Annotations. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 9426–9439, 2024.

[29] Niklas Muennighoff, Zitong Yang, Weijia Shi, Xiang Lisa Li, Li Fei-Fei, Hannaneh Hajishirzi, Luke Zettlemoyer, Percy Liang, Emmanuel Candès, and Tatsunori B Hashimoto. s1: Simple Test-Time Scaling. In *Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2025.

[30] Zheng Li, Qingxiu Dong, Jingyuan Ma, Di Zhang, Kai Jia, and Zhifang Sui. SelfBudgeter: Adaptive Token Allocation for Efficient LLM Reasoning. *arXiv preprint arXiv:2505.11274*, 2025.

[31] Rohin Manvi, Anikait Singh, and Stefano Ermon. Adaptive Inference-Time Compute: LLMs Can Predict if They Can Do Better, Even Mid-Generation. *arXiv preprint arXiv:2410.02725*, 2024.

[32] Charlie Snell, Jaehoon Lee, Kelvin Xu, and Aviral Kumar. Scaling LLM Test-Time Compute Optimally can be More Effective than Scaling Model Parameters. In *International Conference on Learning Representations (ICLR)*, 2025.

[33] Mehul Damani, Idan Shenfeld, Andi Peng, Andreea Bobu, and Jacob Andreas. Learning How Hard to Think: Input-Adaptive Allocation of LM Computation. In *International Conference on Learning Representations (ICLR)*, 2025.

[34] Jiahao Meng, Shuyang Sun, Yue Tan, Lu Qi, Yunhai Tong, Xiangtai Li, and Longyin Wen. CyberV: Cybernetics for Test-time Scaling in Video Understanding. *arXiv preprint arXiv:2506.07971*, 2025.

[35] Hongbo Jin, Jiayu Ding, Siyi Xie, Guibo Luo, and Ge Li. VISTA: Mitigating Semantic Inertia in Video-LLMs via Training-Free Dynamic Chain-of-Thought Routing. *arXiv preprint arXiv:2505.11830*, 2025.

[36] Yixu Huang, Tinghui Zhu, and Muhao Chen. Learning Adaptive Reasoning Paths for Efficient Visual Reasoning. *arXiv preprint arXiv:2604.14568*, 2026.

[37] Shuang Chen, Yue Guo, Yimeng Ye, Shijue Huang, Wenbo Hu, Haoxi Li, Manyuan Zhang, Jiayu Chen, Song Guo, and Nanyun Peng. ARES: Multimodal Adaptive Reasoning via Difficulty-Aware Token-Level Entropy Shaping. In *International Conference on Learning Representations (ICLR)*, 2026.

[38] Mehmet Onurcan Kaya, Desmond Elliott, and Dim P Papadopoulos. Efficient Test-Time Scaling for Small Vision-Language Models. In *International Conference on Learning Representations (ICLR)*, 2026.

[39] Ruohong Zhang, Bowen Zhang, Yanghao Li, Haotian Zhang, Zhiqing Sun, Zhe Gan, Yinfei Yang, Ruoming Pang, and Yiming Yang. Improve Vision Language Model Chain-of-thought Reasoning. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1631–1662, 2025.

[40] Mingyuan Wu, Meitang Li, Jingcheng Yang, Jize Jiang, Kaizhuo Yan, Zhaoheng Li, Hanchao Yu, Minjia Zhang, and Klara Nahrstedt. Aha Moment Revisited: Are VLMs Truly Capable of Self Verification in Inference-time Scaling? *arXiv preprint arXiv:2506.17417*, 2025.

[41] Ahmadreza Jeddi, Negin Baghbanzadeh, Elham Dolatabadi, and Babak Taati. Similarity-Aware Token Pruning: Your VLM but Faster. *arXiv preprint arXiv:2503.11549*, 2025.

[42] Long Xing, Qidong Huang, Xiaoyi Dong, Jiajie Lu, Pan Zhang, Yuhang Zang, Yuhang Cao, Conghui He, Jiaqi Wang, Feng Wu, and Dahua Lin. PyramidDrop: Accelerating Your Large Vision-Language Models via Pyramid Visual Redundancy Reduction. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2025.

[43] Yuan Zhang, Chun-Kai Fan, Junpeng Ma, Wenzhao Zheng, Tao Huang, Kuan Cheng, Denis Gudovskiy, Tomoyuki Okuno, Yohei Nakata, Kurt Keutzer, and Shanghang Zhang. SparseVLM: Visual Token Sparsification for Efficient Vision-Language Model Inference. In *International Conference on Machine Learning*, 2025.

[44] Senqiao Yang, Yukang Chen, Zhuotao Tian, Chengyao Wang, Jingyao Li, Bei Yu, and Jiaya Jia. VisionZip: Longer is Better but Not Necessary in Vision Language Models. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pages 19792–19802, 2025.

[45] Xiaohu Huang, Hao Zhou, and Kai Han. PruneVid: Visual Token Pruning for Efficient Video Large Language Models. In *Findings of the Association for Computational Linguistics: ACL 2025*, pages 19959–19973, 2025.

[46] Daniel Bolya, Cheng-Yang Fu, Xiaoliang Dai, Peizhao Zhang, Christoph Feichtenhofer, and Judy Hoffman. Token Merging: Your ViT But Faster. In *International Conference on Learning Representations (ICLR)*, 2023.

[47] Saeed Ranjbar Alvar, Gursimran Singh, Mohammad Akbari, and Yong Zhang. DivPrune: Diversity-based Visual Token Pruning for Large Multimodal Models. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pages 9392–9401, 2025.

[48] Qizhe Zhang, Mengzhen Liu, Lichen Li, Ming Lu, Yuan Zhang, Junwen Pan, Qi She, and Shanghang Zhang. Beyond Attention or Similarity: Maximizing Conditional Diversity for Token Pruning in MLLMs. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2025.

[49] Qizhe Zhang, Aosong Cheng, Ming Lu, Renrui Zhang, Zhiyong Zhuo, Jiajun Cao, Shaobo Guo, Qi She, and Shanghang Zhang. Beyond Text-Visual Attention: Exploiting Visual Cues for Effective Token Pruning in VLMs. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2025.

[50] Cheng Yang, Yang Sui, Jinqi Xiao, Lingyi Huang, Yu Gong, Chendi Li, Jinghua Yan, Yu Bai, Ponnuswamy Sadayappan, Xia Hu, and Bo Yuan. TopV: Compatible Token Pruning with Inference Time Optimization for Fast and Low-Memory Multimodal Vision Language Model. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, 2025.

[51] Yanshu Li, Jianjiang Yang, Zhennan Shen, Ligong Han, Haoyan Xu, and Ruixiang Tang. CATP: Contextually Adaptive Token Pruning for Efficient and Enhanced Multimodal In-Context Learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, 2026.

[52] Tri Dao, Dan Fu, Stefano Ermon, Atri Rudra, and Christopher Ré. FlashAttention: Fast and Memory-Efficient Exact Attention with IO-Awareness. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2022.

[53] Ce Zhang, Kaixin Ma, Tianqing Fang, Wenhao Yu, Hongming Zhang, Zhisong Zhang, Yaqi Xie, Katia Sycara, Haitao Mi, and Dong Yu. VScan: Rethinking Visual Token Reduction for Efficient Large Vision-Language Models. *Transactions on Machine Learning Research*, 2026.

[54] Junyoung Park, Dalton Jones, Matthew J Morse, Raghavv Goel, Mingu Lee, and Chris Lott. KeyDiff: Key Similarity-Based KV Cache Eviction for Long-Context LLM Inference in Resource-Constrained Environments. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2025.

[55] Haodong Duan, Xinyu Fang, Junming Yang, Xiangyu Zhao, Yuxuan Qiao, Mo Li, Amit Agarwal, Zhe Chen, Lin Chen, Yuan Liu, Yubo Ma, Hailong Sun, Yifan Zhang, Shiyin Lu, Tack Hwa Wong, Weiyun Wang, Peiheng Zhou, Xiaozhe Li, Chaoyou Fu, Junbo Cui, Jixuan Chen, Enxin Song, Song Mao, Shengyuan Ding, Tianhao Liang, Zicheng Zhang, Xiaoyi Dong, Yuhang Zang, Pan Zhang, Jiaqi Wang, Dahua Lin, and Kai Chen. VLMEvalKit: An Open-Source Toolkit for Evaluating Large Multi-Modality Models. In *Proceedings of the ACM International Conference on Multimedia*, 2024.

[56] Pan Lu, Hritik Bansal, Tony Xia, Jiacheng Liu, Chunyuan Li, Hannaneh Hajishirzi, Hao Cheng, Kai-Wei Chang, Michel Galley, and Jianfeng Gao. MathVista: Evaluating Mathematical Reasoning of Foundation Models in Visual Contexts. In *International Conference on Learning Representations (ICLR)*, 2024.

[57] Renrui Zhang, Dongzhi Jiang, Yichi Zhang, Haokun Lin, Ziyu Guo, Pengshuo Qiu, Aojun Zhou, Pan Lu, Kai-Wei Chang, Peng Gao, and Hongsheng Li. MathVerse: Does Your Multi-modal LLM Truly See the Diagrams in Visual Math Problems? In *European Conference on Computer Vision*, 2024.

[58] Ke Wang, Junting Pan, Weikang Shi, Zimu Lu, Houxing Ren, Aojun Zhou, Mingjie Zhan, and Hongsheng Li. Measuring Multimodal Mathematical Reasoning with MATH-Vision Dataset. *Advances in Neural Information Processing Systems (NeurIPS)*, 37:95095–95169, 2024.

[59] Minesh Mathew, Dimosthenis Karatzas, and CV Jawahar. DocVQA: A Dataset for VQA on Document Images. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*, 2021.

[60] Xiang Yue, Tianyu Zheng, Yuansheng Ni, Yubo Wang, Kai Zhang, Shengbang Tong, Yuxuan Sun, Botao Yu, Ge Zhang, Huan Sun, Yu Su, Wenhu Chen, and Graham Neubig. MMMU-Pro: A More Robust Multi-discipline Multimodal Understanding Benchmark. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 15134–15186, 2025.

[61] Chaoyou Fu, Peixian Chen, Yunhang Shen, Yulei Qin, Mengdan Zhang, Xu Lin, Jinrui Yang, Xiawu Zheng, Ke Li, Xing Sun, Yunsheng Wu, Rongrong Ji, Caifeng Shan, and Ran He. MME: A Comprehensive Evaluation Benchmark for Multimodal Large Language Models. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2025.

[62] Lin Chen, Jinsong Li, Xiaoyi Dong, Pan Zhang, Yuhang Zang, Zehui Chen, Haodong Duan, Jiaqi Wang, Yu Qiao, Dahua Lin, and Feng Zhao. Are We on the Right Way for Evaluating Large Vision-Language Models? In *Advances in Neural Information Processing Systems (NeurIPS)*, 2024.

[63] Yuan Liu, Haodong Duan, Yuanhan Zhang, Bo Li, Songyang Zhang, Wangbo Zhao, Yike Yuan, Jiaqi Wang, Conghui He, Ziwei Liu, Kai Chen, and Dahua Lin. MMBench: Is Your Multi-modal Model an All-around Player? In *European Conference on Computer Vision*, 2024.

[64] Shengbang Tong, Ellis Brown, Penghao Wu, Sanghyun Woo, Manoj Middepogu, Sai Charitha Akula, Jihan Yang, Shusheng Yang, Adithya Iyer, Xichen Pan, Ziteng Wang, Rob Fergus, Yann LeCun, and Saining Xie. Cambrian-1: A Fully Open, Vision-Centric Exploration of Multimodal LLMs. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2024.

[65] Yifan Li, Yifan Du, Kun Zhou, Jinpeng Wang, Wayne Xin Zhao, and Ji-Rong Wen. Evaluating Object Hallucination in Large Vision-Language Models. In *Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2023.

[66] Xingyu Fu, Yushi Hu, Bangzheng Li, Yu Feng, Haoyu Wang, Xudong Lin, Dan Roth, Noah A. Smith, Wei-Chiu Ma, and Ranjay Krishna. BLINK: Multimodal Large Language Models Can See but Not Perceive. In *European Conference on Computer Vision*, 2024.

[67] Haochen Wang, Xiangtai Li, Zilong Huang, Anran Wang, Jiacong Wang, Tao Zhang, Jiani Zheng, Sule Bai, Zijian Kang, Jiashi Feng, Zhuochen Wang, and Zhaoxiang Zhang. Traceable Evidence Enhanced Visual Grounded Reasoning: Evaluation and Methodology. In *International Conference on Learning Representations (ICLR)*, 2026.

[68] Chaoyou Fu, Yuhan Dai, Yongdong Luo, Lei Li, Shuhuai Ren, Renrui Zhang, Zihan Wang, Chenyu Zhou, Yunhang Shen, Mengdan Zhang, Peixian Chen, Yanwei Li, Shaohui Lin, Sirui Zhao, Ke Li, Tong Xu, Xiawu Zheng, Enhong Chen, Caifeng Shan, Ran He, and Xing Sun. Video-MME: The First-Ever Comprehensive Evaluation Benchmark of Multi-modal LLMs in Video Analysis. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, 2025.

[69] Yuanxin Liu, Shicheng Li, Yi Liu, Yuxiang Wang, Shuhuai Ren, Lei Li, Sishuo Chen, Xu Sun, and Lu Hou. TempCompass: Do Video LLMs Really Understand Videos? In *Findings of the Association for Computational Linguistics: ACL 2024*, pages 8731–8772, 2024.

[70] Yuanhan Zhang, Yunice Chew, Yuhao Dong, Aria Leo, Bo Hu, and Ziwei Liu. Towards Video Thinking Test: A Holistic Benchmark for Advanced Video Reasoning and Understanding. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2025.

[71] Kunchang Li, Yali Wang, Yinan He, Yizhuo Li, Yi Wang, Yi Liu, Zun Wang, Jilan Xu, Guo Chen, Ping Luo, Limin Wang, and Yu Qiao. MVBench: A Comprehensive Multi-modal Video Understanding Benchmark. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2024.

[72] Zicheng Zhang, Ziheng Jia, Haoning Wu, Chunyi Li, Zijian Chen, Yingjie Zhou, Wei Sun, Xiaohong Liu, Xiongkuo Min, Weisi Lin, and Guangtao Zhai. Q-Bench-Video: Benchmark the Video Quality Understanding of LMMs. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, 2025.

[73] Kairui Hu, Penghao Wu, Fanyi Pu, Wang Xiao, Yuanhan Zhang, Xiang Yue, Bo Li, and Ziwei Liu. Video-MMMU: Evaluating Knowledge Acquisition from Multi-Discipline Professional Videos. *arXiv preprint arXiv:2501.13826*, 2025.

[74] Woosuk Kwon, Zhuohan Li, Siyuan Zhuang, Ying Sheng, Lianmin Zheng, Cody Hao Yu, Joseph E. Gonzalez, Hao Zhang, and Ion Stoica. Efficient memory management for large language model serving with pagedattention. In *Proceedings of the ACM SIGOPS 29th Symposium on Operating Systems Principles*, 2023.

[75] Yihe Deng, Hritik Bansal, Fan Yin, Nanyun Peng, Wei Wang, and Kai-Wei Chang. OpenVLThinker: Complex Vision-Language Reasoning via Iterative SFT-RL Cycles. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2025.

[76] Andrew Jaegle, Felix Gimeno, Andy Brock, Oriol Vinyals, Andrew Zisserman, and Joao Carreira. Perceiver: General perception with iterative attention. In *International conference on machine learning*, pages 4651–4664. PMLR, 2021.

[77] Yuanhan Zhang, Jinming Wu, Wei Li, Bo Li, Zejun Ma, Ziwei Liu, and Chunyuan Li. LLaVA-Video: Video Instruction Tuning With Synthetic Data. *Transactions on Machine Learning Research*, 2025.

[78] Dustin Schwenk, Apoorv Khandelwal, Christopher Clark, Kenneth Marino, and Roozbeh Mottaghi. A-OKVQA: A Benchmark for Visual Question Answering using World Knowledge. In *European Conference on Computer Vision*, 2022.

[79] Aniruddha Kembhavi, Mike Salvato, Eric Kolve, Minjoon Seo, Hannaneh Hajishirzi, and Ali Farhadi. A Diagram Is Worth A Dozen Images. In *European Conference on Computer Vision*, 2016.

[80] Aniruddha Kembhavi, Minjoon Seo, Dustin Schwenk, Jonghyun Choi, Ali Farhadi, and Hannaneh Hajishirzi. Are You Smarter Than a Sixth Grader? Textbook Question Answering for Multimodal Machine Comprehension. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017.

[81] Bohao Li, Rui Wang, Guangzhi Wang, Yuying Ge, Yixiao Ge, and Ying Shan. SEED-Bench: Benchmarking Multimodal LLMs with Generative Comprehension. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2024.

[82] 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.

[83] Ahmed Masry, Xuan Long Do, Jia Qing Tan, Shafiq Joty, and Enamul Hoque. ChartQA: A Benchmark for Question Answering about Charts with Visual and Logical Reasoning. In *Findings of the Association for Computational Linguistics: ACL 2022*, 2022.

[84] Xiang Yue, Yuansheng Ni, Kai Zhang, Tianyu Zheng, Ruoqi Liu, Ge Zhang, Samuel Stevens, Dongfu Jiang, Weiming Ren, Yuxuan Sun, Cong Wei, Botao Yu, Ruibin Yuan, Renliang Sun, Ming Yin, Boyuan Zheng, Zhenzhu Yang, Yibo Liu, Wenhao Huang, Huan Sun, Yu Su, and Wenhu Chen. MMMU: A Massive Multi-discipline Multimodal Understanding and Reasoning Benchmark for Expert AGI. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2024.

[85] Drew A Hudson and Christopher D Manning. GQA: A New Dataset for Real-World Visual Reasoning and Compositional Question Answering. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019.

Contents

A Proofs **17**

A.1 Token-linearity of Language Model FLOPs 17

A.2 FLOPs Scaling With K Trajectories Under Shared-prefill 18

B Data Curation for Training the Difficulty Predictor **19**

B.1 Quality Filtering 19

B.2 Difficulty Estimation and Validity Checks 19

B.3 Diversity 19

B.4 Balancing and De-duplication 20

C Ablation Studies **20**

C.1 Adaptive Rollout Allocation 20

C.2 Ablating the Learned Difficulty Predictor 20

C.3 Ablations on KDV 21

D Limitations and Future Work **22**

E Societal Impact **23**

A Proofs

A.1 Token-linearity of Language Model FLOPs

Proposition A.1 (Token-linearity of LM FLOPs)**.** *Let $n_{\text{ctx}} = n_v + n_q$ denote the multimodal prefix length (visual + text tokens) and let T be the number of decoded tokens. For a given model $\mathcal{M}$, define $C_{\mathcal{M}}$ as the average FLOPs incurred by the LM stack per processed token (summed over all transformer layers, excluding the vision encoder). Then the total inference FLOPs for one completion satisfy*

$$F(I, Q, T) \;\approx\; F_v(I) \;+\; C_{\mathcal{M}}\,(n_{\text{ctx}} + T). \tag{6}$$

Proof. We separate the vision encoder from the LM stack:

$$F(I, Q, T) \;=\; F_v(I) \;+\; F_{\text{LM}}(n_{\text{ctx}}, T). \tag{7}$$

It suffices to show that $F_{\text{LM}}(n_{\text{ctx}}, T)$ is well-approximated by a constant times $(n_{\text{ctx}} + T)$.

We use the per-layer FLOPs model

$$C_{\text{pre}}(n) = 4d\,n^2 + B_0 n, \qquad C_{\text{step}}(n) = 4d\,n + B_0, \tag{8}$$

where token-linear terms (projections + MLP) are grouped into B_0:

$$B_0 \;:=\; 4d^2 \;+\; 4d\,d_{\text{kv}} \;+\; \gamma dm, \tag{9}$$

In (8), n is the sequence length (tokens), $C_{\text{pre}}(n)$ is per-layer prefill FLOPs (KV-cache build), and $C_{\text{step}}(n)$ is per-layer FLOPs for one KV-cached decode step at cache length n. In (9), d is the LLM hidden size, d_{kv} is the KV projection width, m is the FFN/MLP intermediate size, B_0 groups token-linear terms (projections + FFN), and γ is the FFN constant ($\gamma = 6$ for SwiGLU).

Let the LM have L layers. Under KV-cached decoding:

- **Prefill:** build the KV cache on the prefix of length n_{ctx}, costing $L\,C_{\text{pre}}(n_{\text{ctx}})$.
- **Decode:** at step $t \in \{1, \ldots, T\}$, the cache length is $n_{\text{ctx}} + (t-1)$, costing $L\,C_{\text{step}}(n_{\text{ctx}} + t - 1)$.

Therefore,

$$F_{\text{LM}}(n_{\text{ctx}}, T) = L\,C_{\text{pre}}(n_{\text{ctx}}) + \sum_{t=1}^{T} L\,C_{\text{step}}(n_{\text{ctx}} + t - 1). \tag{10}$$

Substitute (8) into (10):

$$\begin{aligned} F_{\text{LM}}(n_{\text{ctx}}, T) &= L\big(4d\,n_{\text{ctx}}^2 + B_0 n_{\text{ctx}}\big) \\ &\quad + \sum_{t=1}^{T} L\big(4d\,(n_{\text{ctx}} + t - 1) + B_0\big) \\ &= LB_0(n_{\text{ctx}} + T) \\ &\quad + 4Ld\left(n_{\text{ctx}}^2 + \sum_{t=1}^{T}(n_{\text{ctx}} + t - 1)\right). \end{aligned} \tag{11}$$

Compute the sum:

$$\sum_{t=1}^{T}(n_{\text{ctx}} + t - 1) = \sum_{t=1}^{T} n_{\text{ctx}} + \sum_{t=1}^{T}(t-1) = n_{\text{ctx}}T + \frac{T(T-1)}{2}.$$

Plugging in gives the exact expression

$$\begin{aligned} F_{\text{LM}}(n_{\text{ctx}}, T) &= LB_0(n_{\text{ctx}} + T) \\ &\quad + 4Ld\left(n_{\text{ctx}}^2 + n_{\text{ctx}}T + \frac{T(T-1)}{2}\right). \end{aligned} \tag{12}$$

Binding nonlinear term by linear function of $(n_{\text{ctx}} + T)$. Assume an operating regime with

$$0 \le n_{\text{ctx}} \le n_{\max}, \qquad 0 \le T \le T_{\max}.$$

Define

$$\kappa \;:=\; n_{\max} + \frac{T_{\max} - 1}{2}. \tag{13}$$

We bound the three nonlinear terms one-by-one:

$$n_{\text{ctx}}^2 \le n_{\max} n_{\text{ctx}}, n_{\text{ctx}} T \le n_{\max} T, \frac{T(T-1)}{2} \le \frac{T_{\max} - 1}{2}\, T.$$

Adding these inequalities gives

$$\begin{aligned} n_{\text{ctx}}^2 + n_{\text{ctx}} T + \frac{T(T-1)}{2} &\le n_{\max} n_{\text{ctx}} + n_{\max} T + \frac{T_{\max} - 1}{2}\, T \\ &= n_{\max} n_{\text{ctx}} + \Big(n_{\max} + \frac{T_{\max} - 1}{2}\Big) T \\ &= n_{\max} n_{\text{ctx}} + \kappa T. \end{aligned} \tag{14}$$

Since $\kappa \ge n_{\max}$ and $n_{\text{ctx}} \ge 0$, we have $n_{\max} n_{\text{ctx}} \le \kappa n_{\text{ctx}}$, hence

$$n_{\text{ctx}}^2 + n_{\text{ctx}} T + \frac{T(T-1)}{2} \;\le\; \kappa n_{\text{ctx}} + \kappa T = \kappa\,(n_{\text{ctx}} + T). \tag{15}$$

Linear sandwich and token-linearity. Start from (12). The bracketed term is nonnegative, so

$$F_{\text{LM}}(n_{\text{ctx}}, T) \;\ge\; L B_0 (n_{\text{ctx}} + T). \tag{16}$$

Using (15) in (12) gives

$$\begin{aligned} F_{\text{LM}}(n_{\text{ctx}}, T) &\le L B_0 (n_{\text{ctx}} + T) + 4 L d\, \kappa\, (n_{\text{ctx}} + T) \\ &= L(B_0 + 4 d \kappa)\, (n_{\text{ctx}} + T). \end{aligned} \tag{17}$$

Thus

$$L B_0 (n_{\text{ctx}} + T) \;\le\; F_{\text{LM}}(n_{\text{ctx}}, T) \;\le\; L(B_0 + 4 d \kappa)\, (n_{\text{ctx}} + T). \tag{18}$$

We therefore use the token-linear surrogate $F_{\text{LM}}(n_{\text{ctx}}, T) \approx C_{\mathcal{M}}(n_{\text{ctx}} + T)$, where $C_{\mathcal{M}}$ is measured once for $\mathcal{M}$, and substituting into (7) yields (6). □

A.2 FLOPs Scaling With K Trajectories Under Shared-prefill

Theorem A.2 (FLOPs scaling with K trajectories under shared-prefill)**.** *Under shared-prefill execution, all K trajectories reuse the same vision features and prefill KV cache. If the k-th trajectory decodes T_k tokens, then*

$$F_K(I, Q; T_{1:K}) = F_v(I) + F_{\text{pre}}(n_v, n_q) + \sum_{k=1}^{K} F_{\text{dec}}(n_v, n_q, T_k), \tag{19}$$

where F_K is the inference FLOPs for generating Y_K, or K trajectories. If $T_k \overset{\text{i.i.d.}}{\sim} T$, then $\mathbb{E}[F_K(I, Q)] = F_v(I) + F_{\text{pre}}(n_v, n_q) + K\, \mathbb{E}[F_{\text{dec}}(n_v, n_q, T)]$.

Proof. The vision encoder depends only on I and is computed once, incurring $F_v(I)$ FLOPs; its output is reused across samples. Given the vision features and prompt text s, the prefill pass that constructs the KV cache depends only on (I, s) and is likewise computed once, incurring $F_{\text{pre}}(I, s)$ FLOPs. For each sample k, autoregressive generation produces T_k tokens; by definition, decoding T_k tokens given the cached states costs $F_{\text{dec}}(I, s, T_k)$ FLOPs. Decoding computations across different samples cannot be shared beyond the cached states because they depend on distinct sampled token sequences, hence the total FLOPs equal the shared vision and prefill costs plus the sum of per-sample decoding costs, yielding Equation 19. Taking expectations over i.i.d. lengths T_k gives Theorem A.2. □

Table 6: Token counts and scaling behavior as a function of K. The table shows that increasing K induces an approximately linear growth in total decoded tokens, with an $8.7\times$ increase at $K = 9$ relative to $K = 1$, confirming that inference scaling is dominated by token accumulation across self-consistency passes.

K	$\sum_{i\in\mathcal{D}} T_{i,1}$	$\sum_{i\in\mathcal{D}} \sum_{k=1}^{K} T_{i,k}$	Token ratio $\left(\frac{\sum_i \sum_{k\le K} T_{i,k}}{\sum_i T_{i,1}}\right)$
1	155,038	155,038	1.00
3		459,114	2.96
5		753,010	4.86
7		1,055,468	6.81
9		1,352,350	8.72

A.2.1 Empirical example of the FLOPs model

Theorem A.2 shows that under shared-prefill, increasing K affects only the decoding term, which sums over trajectories. If rollouts are sampled i.i.d. under a fixed decoding policy, the total number of generated tokens over a curated dataset scales approximately linearly with K: $\sum_{i\in\mathcal{D}} \sum_{k=1}^{K} T_{i,k} \approx K \sum_{i\in\mathcal{D}} T_{i,1}$ (Table 6). Combined with Proposition A.1, which models decoding FLOPs as approximately linear in decoded tokens, this yields near-linear scaling of decoding compute with K, while F_v and F_{pre} remain one-time costs.

B Data Curation for Training the Difficulty Predictor

We curate a calibration set for training the difficulty predictor by downselecting from an initial pool of multimodal samples (62,430 images and 10,709 videos) collected from open-source benchmarks, including LLaVA-Video-178K [77], AOKVQA [78], AI2D [79], TQA [80], SEEDBench [81], ScienceQA [82], ChartQA [83], and MMMU [84]. Our filtering follows three principles, inspired by S1 [29]: **Quality**, **Difficulty**, and **Diversity**.

B.1 Quality Filtering

We use Qwen2.5-VL-32B as an automatic judge to remove low-quality or ill-posed items, including: (i) vague or underspecified questions, (ii) poor formatting or ambiguous answer options, (iii) cases where the image/video is not required to solve the question, and (iv) corrupt or low-quality visual inputs. We also retain only multiple-choice questions with explicit answer options. After this stage, we retain 45,000 samples.

B.2 Difficulty Estimation and Validity Checks

We estimate difficulty using Qwen2.5-VL-7B with CoT prompting. For each candidate, we run 5 stochastic passes (temperature sampling) and record the empirical success rate. Since all questions are multiple-choice, we apply a lightweight answer extractor and use a Qwen-7B judge to verify that each output (i) conforms to the expected choice format and (ii) matches the ground-truth answer. We discard samples for which any of the 5 passes yields an invalidly formatted output. After this stage, 31,423 samples remain.

B.3 Diversity

To encourage broad coverage across visual domains and question types, we use Qwen2.5-VL-32B to assign each candidate to one of 36 coarse subject categories (e.g., documents/graphics, indoor scenes, outdoor scenes, people, sports, tools, STEM, diagrams, charts). We then construct a subject-stratified subset by sampling approximately uniformly across categories while preserving a balanced spread over the difficulty spectrum. This yields 8,000 samples.

B.4 Balancing and De-duplication

To avoid over-representing particular difficulty bands, we further subsample approximately uniformly over the success-rate spectrum. Finally, because several benchmarks share overlapping visuals, we remove near-duplicates between the training pool and evaluation benchmarks. Specifically, we embed all images using CLIP and discard any training example whose visual embedding has cosine similarity greater than 0.95 with an evaluation image. For videos, we apply the same criterion using uniformly sampled frames and remove a video if any sampled frame exceeds this threshold. This conservative threshold is intended to remove exact or near-exact visual overlap while avoiding the removal of merely semantically similar examples. The final dataset contains 4,000 images and 1,000 videos.

C Ablation Studies

C.1 Adaptive Rollout Allocation

Figure 4 shows the empirical distribution of rollout counts selected by the learned VRS controller, where $K \in \{1, 3, 5, 7\}$. The allocation is highly dataset-dependent, indicating that the controller learns non-trivial compute profiles rather than collapsing to a fixed decoding budget. The average selected rollout count is 2.12 on MMBench, 2.53 on MME, and 3.79 on Video-TT, reflecting increasing use of reasoning-time search on benchmarks where additional rollouts are more likely to be beneficial.

In addition, Figure 4 shows that AVIS preserves low-cost inference for many queries while selectively increasing compute for a targeted subset. On MMBench, the controller is relatively conservative, but still assigns larger budgets to non-negligible fractions of the data: 14.2% of samples receive $K = 5$ and 7.6% receive $K = 7$. On MME, more than one-third of samples are assigned $K > 1$, suggesting that the predictor identifies a substantial set of examples where self-consistency may improve reliability. The effect is strongest on Video-TT, where $K = 5$ is the most frequent assignment, covering 51.4% of samples, consistent with the greater reasoning and temporal grounding demands of video understanding.

These results confirm that the learned rollout selector performs input-adaptive reasoning allocation. It does not simply increase test-time compute uniformly, nor does it degenerate to greedy decoding. Instead, it amortizes the inference budget across the workload: easy or low-gain examples are handled with a small K, while additional rollouts are concentrated on examples predicted to benefit from reasoning-time search. This behavior explains how AVIS can capture part of the accuracy gain of self-consistency while maintaining a substantially lower average compute cost.

C.2 Ablating the Learned Difficulty Predictor

Effect of calibration set size. We first study how the size of the calibration pool affects predictor quality. Using a fixed held-out split of 250 samples, predictor accuracy improves monotonically as the amount of training data increases. Training with 20%, 50%, 75%, and 100% of the remaining calibration pool yields predictor accuracies of 64.4%, 73.8%, 77.4%, and 79.2%, respectively. This trend indicates that the solvability signal is learnable from a moderate number of calibration examples, and that the full 5000 calibration pool provides a stable operating point for adaptive rollout selection.

Architecture and end-to-end impact. We compare our AVIS predictor against alternative lightweight designs. On the held-out calibration split, a two-layer Transformer and a Perceiver-style predictor achieve 72.0% and 69.6% predictor accuracy, respectively, while our final predictor achieves the highest held-out accuracy, 79.2%. We then evaluate these variants end-to-end on MathVista and MMBench, as shown in Table 7.

The end-to-end results show that predictor quality primarily improves compute allocation rather than simply increasing downstream accuracy. The Perceiver and Transformer variants select larger rollout counts, leading to higher FLOPs with little or no accuracy gain. On MathVista, our predictor matches the Perceiver score of 68.1 while reducing average K from 3.57 to 2.33 and FLOPs from $0.64\times$ to $0.46\times$. The Transformer obtains a slightly higher MathVista score of 68.6, but requires 2.93 average rollouts and $0.53\times$ FLOPs. On MMBench, our predictor achieves comparable accuracy to the Perceiver, 86.4 versus 86.6, while reducing average K from 3.25 to 2.12 and FLOPs from

Table 7: **Ablation of difficulty predictor architectures.** We keep the pruning policy, decoding settings, and budget calibration fixed, and change only the predictor architecture. Avg. K denotes the average number of selected reasoning rollouts, and #F denotes normalized FLOPs relative to the vanilla $K = 1$ baseline.

Predictor	Pred. Acc.	MathVista			MMBench		
		Score	Avg. K	#F	Score	Avg. K	#F
Perceiver	69.6	68.1	3.57	0.64	86.6	3.25	0.59
Transformer	72.0	68.6	2.93	0.53	86.3	2.57	0.51
AVIS (Ours)	79.2	68.1	2.33	0.46	86.4	2.12	0.44

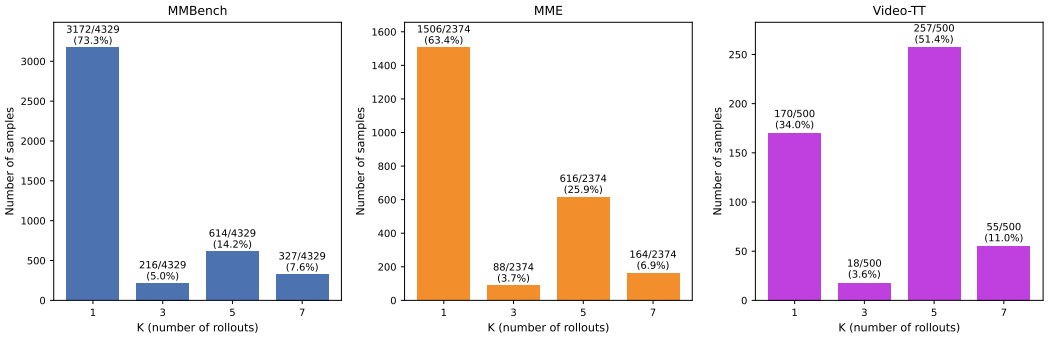


Figure 4: **Distribution of predicted rollout counts K.** Histogram of the learned difficulty-aware policy's K assignments on representative image and video benchmarks. Because the binning rule reflects the inverted-U marginal utility of majority voting, $K{=}1$ is assigned at both the easy and very-hard extremes, while $K{=}5$ and $K{=}7$ are reserved for intermediate hard-but-solvable examples. The resulting distribution is dataset-dependent, shifting from $K{=}1$-dominated on MMBench to $K{=}5$-dominated on Video-TT.

$0.59\times$ to $0.44\times$. Compared with the Transformer, our predictor also uses fewer rollouts and lower FLOPs on both benchmarks, with comparable downstream accuracy.

These results suggest that higher predictor accuracy translates into a better accuracy–FLOPs trade-off. Less accurate predictors produce less reliable solvability estimates, which causes the binning rule to assign larger K more often. This increases compute without consistent downstream gains. In contrast, our AVIS predictor produces sharper and more useful solvability estimates, avoiding unnecessary rollouts while preserving accuracy.

Predictor overhead. The learned difficulty predictor itself adds negligible inference overhead. It consists of a stack of 1D Conv–GroupNorm–SiLU layers followed by global average pooling and a two-layer MLP. Since it operates on the visual token embeddings already computed by the vision encoder, it is executed only once per input and does not require any additional VLM forward pass. Its average cost is approximately 6 GFLOPs per example, corresponding to less than 0.1% of total inference FLOPs. This overhead is far smaller than the compute saved by visual token pruning or by avoiding unnecessary reasoning rollouts.

C.3 Ablations on KDV

In this section, we evaluate **KDV** as a standalone visual token pruning method. We use Qwen2.5-VL-7B-Instruct and report single-pass inference results ($K{=}1$), without enforcing the explicit `<think>` format used in our main experiments.[4] All experiments follow the VLMEvalKit default image resolution (1280) and use a fixed pruning configuration that retains 320 out of 1280 visual tokens (approximately 75% pruning).

Table 8 summarizes performance on GQA, MMBench-DEV-EN, MME (perception+reasoning sum), POPE, ScienceQA-Image, CV-Bench-2D, SeedBench-Image, and MMStar. Compared to the full-token *Vanilla* baseline, all training-free pruning methods incur only modest degradation when

[4]The model may still generate chain-of-thought text even without enforcing the format.

Table 8: **KDV pruning on Qwen-2.5-VL-Instruct.** We report absolute scores and percentages relative to the full-token (*Vanilla*) baseline. For this ablation, we do not enforce the `<think>` prompting format. Results are shown on GQA [85], MMBench-DEV-EN [63], MME [61] (perception+reasoning sum), POPE [65] (overall accuracy), ScienceQA-Image [82], CV-Bench-2D [64], SeedBench-Image [81], and MMStar [62]. We compare against Vanilla and representative training-free pruning baselines: FastV [10] (K=2), VisionZip [44] (dominant tokens = 22%, context = 3%), PDrop [42] (L=[8, 16, 24], R=0.5), and KeyDiff [54].

Method	GQA	MMB	MME	POPE	SQA	CVBench	SEED	MMStar	Avg.
Upper Bound, 1280 Tokens (100%)									
Vanilla	60.26 100%	78.69 100%	2299 100%	87.87 100%	72.83 100%	74.83 100%	76.91 100%	60.33 100%	100%
Retain 320 Tokens (↓ 75%)									
FastV	56.74 94.16%	77.66 98.69%	2300 100.04%	85.71 97.54%	71.54 98.23%	70.04 93.60%	70.04 91.07%	54.6 90.50%	95.48%
VisionZip	59.07 98.03%	77.92 99.02%	2300 100.04%	87.06 99.08%	73.07 100.33%	73.3 97.96%	73.76 95.90%	55.8 92.49%	97.86%
PDrop	56.91 94.44%	77.41 98.37%	2284 99.35%	87.57 99.66%	72.33 99.31%	74.02 98.92%	74.48 96.84%	55.93 92.71%	97.45%
KeyDiff	58.03 96.30%	77.21 98.12%	2309 100.43%	86.87 98.86%	73.08 100.34%	73.3 97.96%	73.78 95.93%	55.8 92.49%	97.55%
KDV (Ours)	58.76 97.51%	78.53 99.80%	2325 101.13%	86.87 98.86%	73.38 100.76%	73.14 97.74%	73.44 95.49%	55.87 92.61%	**97.99%**

retaining 25% of visual tokens. FastV achieves on average 95.48% of the baseline score, VisionZip 97.86%, PDrop 97.45%, and KeyDiff 97.55%. KDV attains the best average retention at **97.99%**, with small gains on token-heavy benchmarks such as MME (101.13% of Vanilla) and competitive performance elsewhere. These results indicate that the key-diversity-based criterion used by KDV is a strong drop-in alternative to existing training-free pruning strategies when operating under tight visual token budgets.

D Limitations and Future Work

AVIS is intentionally a lightweight, deployment-friendly instantiation of adaptive VCS–VRS allocation, and several of its design choices open natural directions for future work.

Joint VCS–VRS policies. AVIS factorizes the inference policy into a vision-side pruning step (KDV) and a difficulty-aware rollout selector. This factorization is what makes both decisions runnable before LM prefill and is central to the compute savings we report. A natural extension is a single policy that selects ρ and K jointly under an explicit per-query compute constraint; in principle, such a policy could exploit interactions between visual context and reasoning effort that our staged design does not model. Our preliminary explorations suggest, however, that effective joint policies are non-trivial to design: naive instantiations we tried did not outperform AVIS, indicating that this direction likely requires richer training signals or differentiable surrogates for the discrete pruning and rollout decisions. We view this as an important and open problem rather than a straightforward extension.

Text-aware difficulty prediction. Our difficulty predictor uses only the visual embeddings produced by the vision encoder. This choice is what allows K to be selected before any LM forward pass and is the reason adaptive rollout selection composes cleanly with shared-prefill inference. Incorporating the question text could in principle sharpen difficulty estimates, but the most natural way to do so, via later LM representations [33], would require running part of the LM prefill before K is decided, partially eroding these savings. Lightweight text-aware predictors that operate before the main LM prefill (e.g., over a small auxiliary text encoder, or over very early LM layers) are a promising direction that preserves the pre-prefill property AVIS relies on.

Composition with other test-time scaling axes. We instantiate VRS through self-consistency with majority voting because it composes naturally with shared-prefill inference and is well-supported

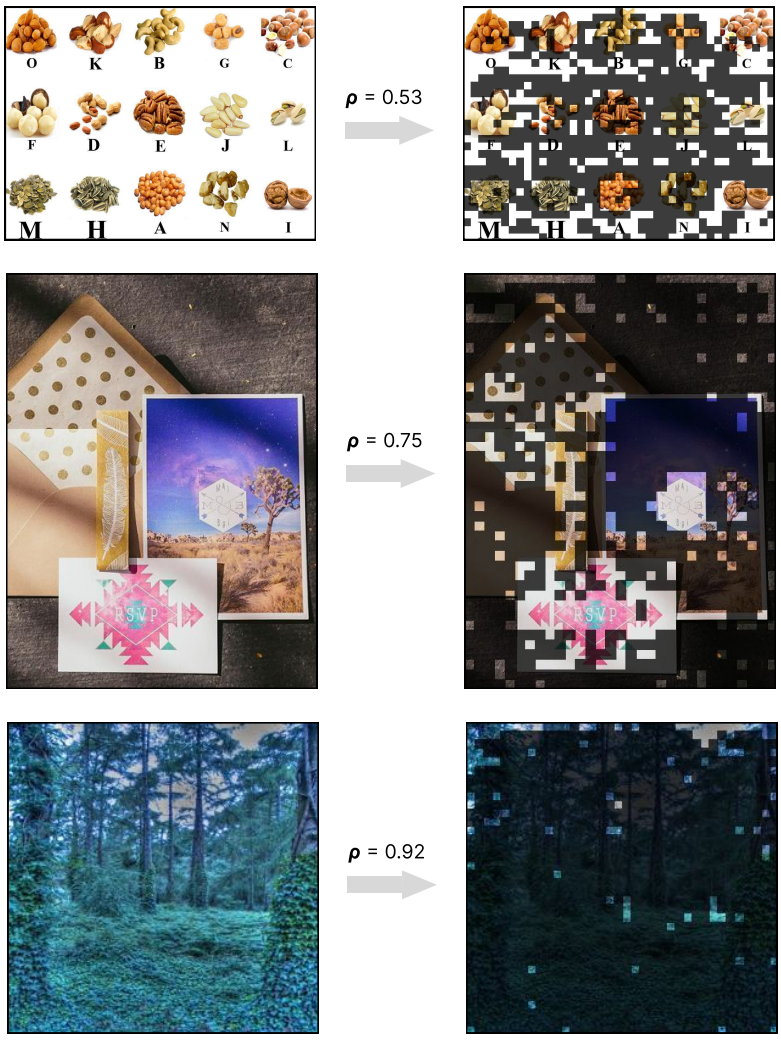


Figure 5: **Adaptive KDV pruning.** Qualitative examples of our adaptive KDV policy selecting different pruning ratios ρ per image. In each pair, the left panel shows the original image and the right panel shows the retained visual tokens after pruning. *Images are from "Are We on the Right Way for Evaluating Large Vision-Language Models?" by Chen et al., published in NeurIPS 2024.*

across model families. Other test-time scaling axes, sequential refinement, longer chain-of-thought, tree search, and Best-of-N with learned verifiers or process reward models, are orthogonal to our allocation framework rather than competing with it, and combining adaptive parallel rollouts with adaptive sequential reasoning is a natural next step.

E Societal Impact

AVIS is a test-time inference policy for existing vision-language models; it does not introduce new model capabilities, training data, or model weights. Its primary effect is to reduce the compute, latency, and energy required for VLM inference at comparable or improved accuracy, which can lower the cost and carbon footprint of multimodal model deployment. The qualitative capabilities and failure modes of the underlying VLMs, including hallucination, biased outputs, and potential for misuse such as automated surveillance or multimodal disinformation, are inherited from the base model and should be addressed with the same safeguards as for the underlying VLM.

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: The abstract and Section 1 clearly outline our core contributions: characterizing the trade-offs of Visual Context Scaling (VCS) and Visual Reasoning Scaling (VRS), and introducing the AVIS policy and Key Diversity Visual (KDV) pruning rule.

   Guidelines:

   - The answer [N/A] means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A [No] or [N/A] answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: We discuss the limitations of our decoupled policy design for visual pruning and reasoning scaling in Section 6 (Conclusion) and suggest directions for future unified approaches.

   Guidelines:

   - The answer [N/A] means that the paper has no limitation while the answer [No] means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

Answer: [Yes]

Justification: We present our FLOPs model and theoretical estimations in Section 3, with complete derivations, formal proofs, and assumptions fully detailed in Appendix A.

Guidelines:

- The answer [N/A] means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

Answer: [Yes]

Justification: We thoroughly describe our baseline configurations, dataset curation process, hyperparameter settings (e.g., pruning threshold $\tau = \pi/4$), and evaluation frameworks in Section 5.1 and Appendix B.

Guidelines:

- The answer [N/A] means that the paper does not include experiments.
- If the paper includes experiments, a [No] answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
- If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
- Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
- While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
  (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
  (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
  (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
  (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in

some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

   Answer: [No]

   Justification: The code and full calibration dataset are not included in the current anonymous submission to preserve double-blind review, but will be open-sourced upon acceptance.

   Guidelines:

   - The answer [N/A] means that paper does not include experiments requiring code.
   - Please see the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - While we encourage the release of code and data, we understand that this might not be possible, so [No] is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
   - The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
   - The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
   - At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
   - Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer) necessary to understand the results?

   Answer: [Yes]

   Justification: We detail the evaluation frameworks, baseline settings, temperature and top-p sampling hyper-parameters in Section 5.1, as well as the calibration dataset generation process in Appendix B.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [No]

   Justification: Error bars are not reported due to the prohibitively high computational cost of running multiple random seeds for large vision-language models evaluated across dozens of extensive image and video benchmarks.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.

- The authors should answer [Yes] if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
- The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
- The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
- The assumptions made should be given (e.g., Normally distributed errors).
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g., negative error rates).
- If error bars are reported in tables or plots, the authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: Hardware details (a single NVIDIA L40 48GB GPU) and specific execution frameworks (vLLM v0.10.0) are documented alongside exact wall-clock latency measurements in Section 5.3.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification: The research utilizes open-source models and benchmark datasets for algorithmic efficiency analysis, conforming to the Code of Ethics.

   Guidelines:

   - The answer [N/A] means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer [No], they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

Answer: [No]

Justification: This paper focuses on foundational algorithm design (inference efficiency). The societal impacts of our method align entirely with those of general VLM development, thus we do not include a dedicated broader impact section.

Guidelines:

- The answer [N/A] means that there is no societal impact of the work performed.
- If the authors answer [N/A] or [No], they should explain why their work has no societal impact or why the paper does not address societal impact.
- Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate Deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pre-trained language models, image generators, or scraped datasets)?

Answer: [N/A]

Justification: Our work proposes an inference-time acceleration technique and does not release any new high-risk pre-trained models or scraped datasets.

Guidelines:

- The answer [N/A] means that the paper poses no such risks.
- Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
- Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
- We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

Answer: [Yes]

Justification: Existing models (Qwen2.5-VL series) and standard evaluation benchmarks are accurately referenced throughout Section 5 and Appendix B.

Guidelines:

- The answer [N/A] means that the paper does not use existing assets.
- The authors should cite the original paper that produced the code package or dataset.
- The authors should state which version of the asset is used and, if possible, include a URL.
- The name of the license (e.g., CC-BY 4.0) should be included for each asset.
- For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
- If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
- For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [N/A]

    Justification: We do not introduce newly collected datasets or foundational model architectures; our work evaluates an inference strategy over existing open-source benchmarks.

    Guidelines:

    - The answer [N/A] means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [N/A]

    Justification: The research does not involve crowdsourcing or human subjects.

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
    - According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [N/A]

Justification: The research does not involve human subjects, rendering IRB approval inapplicable.

Guidelines:

- The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
- Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
- We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
- For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does *not* impact the core methodology, scientific rigor, or originality of the research, declaration is not required.

Answer: [Yes]

Justification: We explicitly document our use of VLM/LLM models (e.g., Qwen2.5-VL-32B and Qwen2.5-VL-7B) as automatic judges to filter data and estimate difficulty during the curation of our calibration dataset, as outlined in Appendix B.

Guidelines:

- The answer [N/A] means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
- Please refer to our LLM policy in the NeurIPS handbook for what should or should not be described.