TwinPrune: Density-Aware Two-Phase Token Pruning for Vision-Language Models

Kai Liu[1]*, **Anqi Li**[1]*, **Junxian Li**[1],
Zhixin Wang[2], **Zhikai Chen**[2], **Renjing Pei**[2], **Yulun Zhang**[1]†
[1]Shanghai Jiao Tong University, [2]Huawei Technologies Ltd

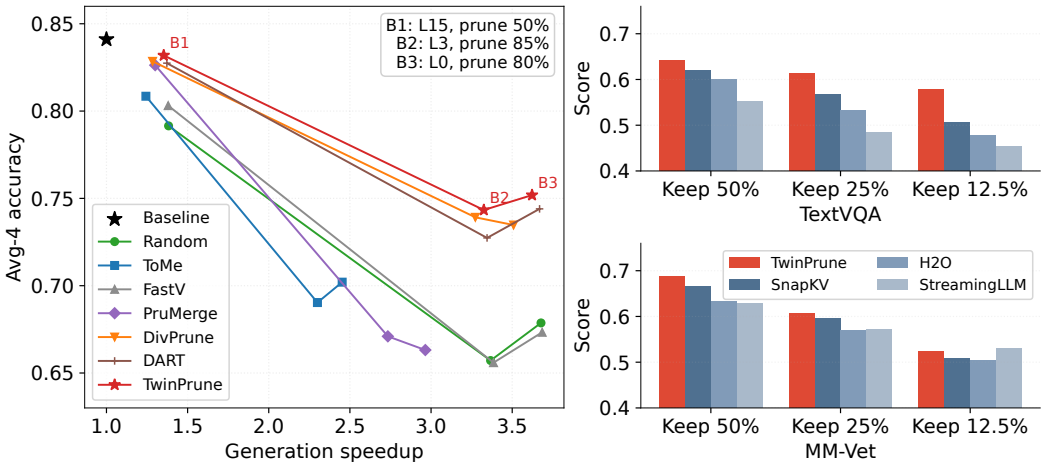


Figure 1: **Two-Stage Pruning in TwinPrune.** TwinPrune improves VLM inference through a two-stage pruning framework using a single unified pruning signal. The left panel evaluates visual token pruning in the prefilling stage, showing better accuracy–speedup trade-offs, while the right panel evaluates KV cache pruning on long-output benchmarks in the decode stage. Both panels use the same InternVL-3.5-8B base weights, with phase-specific evaluation settings.

Abstract

We propose a unified spatial-density signal that scores token importance for *both* phases of inference, instantiated as density-tilted selection during prefill and a KV eviction during decode, with measurable gains in each phase. To verify the two phases independently, we partition benchmarks into two diagnostic roles by their scoring rule. *Prefill probes* such as first-token evaluations of MMBench, POPE, and ScienceQA score only the first decoded token, so their accuracy reflects only the argmax of prefill logits, computed before any decode-phase KV eviction takes effect. *Full-pipeline probes* such as TextVQA, DocVQA, and MM-Vet score the full decoded sequence and therefore reveal both phases. We expose a fundamental evaluation flaw: published claims of decode-phase token pruning validated only on prefill probes have measured prefill quality, not decode quality. Empirically, removing all visual KV at decode leaves prefill-probe accuracy essentially unchanged while collapsing TextVQA and MM-Vet by tens of points, and the insensitivity is a property of the scoring rule, not of the compression strength.

*Equal Contribution
†Corresponding author: Yulun Zhang, yulun100@gmail.com

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

Across six model variants from the InternVL-3.5, Qwen3-VL, and LLaVA-1.5 families and the three full-pipeline-probe benchmarks, we deliver a three-axis study over compute, memory, and accuracy. Density-based prefill compression is competitive on the accuracy frontier; density scoring scales linearly in the visual-token count. On the decode side, density top-k KV eviction wins every TextVQA cell and two of three MM-Vet cells against the strongest published baselines H2O, SnapKV, and StreamingLLM across three keep ratios. The advantage widens to 9.96 percentage points over H2O at 12.5% keep on TextVQA and holds 9.37 to 11.12 percentage points across the InternVL-3.5 series from 2B to 38B. Compression also scales gracefully with model size: four of six matched 8B-to-38B transitions reduce the accuracy loss. We recommend MM-Vet or DocVQA as the minimum full-pipeline-probe test for any decode-phase claim. Our code is available at `https://github.com/Kai-Liu001/TwinPrune`.

1 Introduction

Vision-Language Models (VLMs) now drive a fast-growing set of applications such as document understanding, embodied agents, multi-step visual reasoning, and GUI control. Each of these demands processing hundreds to thousands of visual tokens through tens of transformer layers. As image resolution and tile count rise, two efficiency bottlenecks tighten in tandem. The first is *prefill compute*, dominated by encoding all visual tokens through the language model. The second is *decode memory*, dominated by the per-layer key/value cache that must persist for autoregressive generation. Whether a model can be deployed at all, whether on a memory-constrained device, in a low-latency interactive setting, or under a high-throughput serving regime, is increasingly determined less by how the model is trained and more by how aggressively redundant visual tokens can be removed at inference time.

The token-pruning literature has expanded rapidly in response. Prefill-time methods such as FastV [6], ToMe [4], DivPrune [1], and DART [28] target the compute axis, while decode-time methods such as H2O [33], StreamingLLM [29], and SnapKV [13] target the memory axis. These works converge on a striking shared finding: aggressive compression preserves accuracy almost perfectly, often within a fraction of a point of the uncompressed baseline. The implication, repeated across dozens of papers, is that visual tokens are deeply redundant and can be cut without harm. If that picture were correct, token pruning would be a solved problem, a routine deployment knob rather than an open question.

We argue this consensus rests on a measurement artifact. The benchmarks most commonly used to validate VLM pruning are dominated by single-token outputs such as a multiple-choice letter or a yes/no answer. Under first-token scoring, their score depends only on the first decoded logit, which is fixed by the prefill pass before any decode-phase KV eviction is executed. By construction, decode-phase eviction therefore cannot move those scores, so what is measured under a "decode-phase" label is in fact prefill-phase quality. Whether a method preserves behavior under longer generation, in which information has to be retrieved from the cache during decode, is a separate question, and it is not the one those benchmarks answer. The same critique applies in a milder form to prefill compression: prefill-probe scores see only the argmax shift, not the distributional damage that compression does to longer continuations.

We make the gap concrete by running a VLM with *all* visual KV removed at decode time. On the prefill-probe benchmarks (MMBench, POPE, ScienceQA) the accuracy is nearly unchanged. On full-pipeline-probe benchmarks such as TextVQA and MM-Vet, accuracy drops sharply, and prediction strings flip on a large fraction of samples (Tab. 1, Fig. 2, Sec. 3). The model *is* changing its behavior under compression, but the short-answer probes register much less of that change.

This re-scopes what an honest unified compression framework should look like and what it should be evaluated on. We propose a spatial-density signal ρ that scores each visual token by the local non-redundancy of its $w \times w$ neighborhood on the regular spatial grid, computed once at an early decoder layer from token-to-token cosine distances. The same ρ drives both compression phases. During prefill, we deploy it inside a density-tilted farthest-point recurrence that combines k-center coverage with density priority through a single tilt scalar β, selects the retained visual tokens and drops the rest before passing the reduced sequence forward. During decode, the same ρ values are reused as a top-k score for visual-KV eviction, with no new attention pattern computed. The

two phases can be applied independently or jointly, and we supply a three-axis (compute, memory, accuracy) Pareto study across six model variants and three full-pipeline-probe benchmarks.

Empirically, the unified density signal is competitive on the prefill accuracy frontier, and the implementation, with a Triton-fused farthest-point inner loop and a cached CUDA graph, delivers a $3.62\times$ generation speedup at the most aggressive operating point on InternVL-3.5-8B. On the decode side, density top-k KV eviction wins every TextVQA cell and two of three MM-Vet cells at three keep ratios against H2O, SnapKV, and StreamingLLM, with the gap widening to 9.96 percentage points over H2O at 12.5% keep on TextVQA and holding 9.37 to 11.12pp across the InternVL-3.5 series from 2B to 38B. Compression cost decreases in four of six matched 8B-to-38B comparisons. Our contributions are as follows:

1. **Methodology finding (Sec. 3):** prefill-probe benchmarks (those scoring only the first decoded token) structurally cannot measure decode-phase compression. We provide the proof, an extended cross-architecture mismatch table, and a recommendation for how the field should evaluate decode-phase claims going forward.
2. **Unified density framework (Sec. 4):** a single spatial-density signal that scores token importance for both compression phases, with one kernel and two deployment points, guiding selection during prefill and top-k eviction during decode. We also give a coreset-theoretic interpretation (Sec. 4.1) motivating why the same ranking serves both phases.
3. **Three-axis Pareto study (Sec. 5 and Sec. 6):** across six model variants and three full-pipeline-probe benchmarks, density-based prefill compression is competitive on the accuracy frontier, with linear-cost density scoring.
4. **Cross-architecture and cross-scale robustness (Appendix P):** an implementation-aware comparison of selection methods on multi-axis RoPE backbones, together with the empirical finding that compression cost decreases in four of six matched 8B-to-38B comparisons.

2 Related Work

Prefill-stage compression. FastV [6] drops visual tokens by attention score after a chosen layer. ToMe [4] performs bipartite token merging for ViTs and was adapted to VLMs. DivPrune [1] uses farthest-point sampling on visual features for diversity-driven selection. DART [28] combines L1-norm pivots with cosine-distance expansion. VisionZip [30] selects dominant tokens by CLS-attention and merges the rest by key-state similarity. TokenPacker [11] introduces a coarse-to-fine visual projector that compresses tokens by 75 to 89 percent before the LLM backbone. AvgPool, FitPrune [31], and PruMerge [23] round out a crowded landscape, and concurrent preprints continue to appear, including SAINT [9]. All evaluate primarily on the four-benchmark set we critique below.

Decode-stage KV eviction. For text-only LLMs, H2O [33] and StreamingLLM [29] maintain "heavy hitter" KV entries. SnapKV [13] aggregates attention. FastGen [8] profiles attention-head patterns to build a query-adaptive eviction policy. Quest [25] selects KV pages by query similarity for efficient long-context inference. KV quantization, exemplified by the 2-bit asymmetric KIVI [16], provides a complementary compression axis orthogonal to eviction. On the visual side, LOOK-M [26] merges multimodal KV entries guided by text-prior attention. ElasticCache [17] uses importance-driven KV merging for vision instruction-following models. R-KV [5] combines attention aggregation with a redundancy penalty for reasoning-LLM KV compression. Most works report on the same prefill-probe benchmarks that, as we show, cannot measure their decode-phase effect.

Two-phase view. To our knowledge, no prior work explicitly separates prefill (compute) and decode (memory) as orthogonal axes with distinct algorithms and evaluation requirements. Our framework makes this separation a first-class design principle.

Token pruning as coreset selection. Keeping k of N visual tokens is a *coreset* query on $X \in \mathbb{R}^{N\times d}$, with two dual classical strategies: k-center diversity (DivPrune [1], Sener and Savarese [22]) and density- or sensitivity-weighted sampling (Feldman and Langberg [7], Bachem, Lucic, and Krause [2], submodular maximization [19]). Our density signal ρ is a training-free, deterministic-threshold proxy of the latter family (Appendix H). We are not aware of prior work framing VLM token pruning as a *two-phase* coreset whose subset must simultaneously approximate prefill attention and decode KV recall, and this paper is the first to exploit the resulting signal reuse.

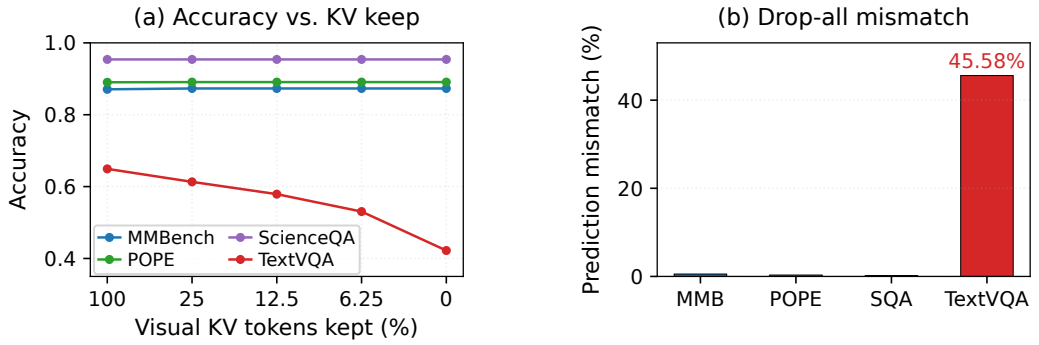


Figure 2: Prefill-probe benchmarks are largely insensitive to the tested decode-phase KV eviction on InternVL-3.5-8B. Left: accuracy vs. visual KV kept on InternVL-3.5-8B. Right: prediction mismatch at DALL grouped by benchmark. Details in Sec. 3.

Table 1: Prediction-mismatch % vs. no-compression baseline on InternVL-3.5-8B. Decode KV eviction flips up to 46% of TextVQA predictions but $\leq$ 0.54% on prefill-probe benches, using full-string comparisons between native-chat and hook paths (Sec. 3). Prefill rows fix L=15 (varying L shifts levels but not the scoring-rule gap). Appendix N extends to all 3 archs.

Phase	Signal · Keep	MMBench	POPE	SQA	TextVQA
Prefill	Density · 75/50/25%	2.24/3.25/5.03	0.88/1.30/2.20	–	9.06/13.98/26.62
Prefill	Random · 25%	6.97	3.70	–	59.18
Prefill	V-projection norm · 25%	5.65	1.73	–	30.58
Decode	Density · 25/12.5/6.25%	0.54	0.25–0.28	0.20	16.04/22.98/31.02
Decode	**Density · 0% (drop all)**	**0.54**	**0.32**	**0.20**	**45.58**

3 Two-Phase Compression, Two-Phase Diagnostic

3.1 The argmax mechanism

Consider a VLM prefill that yields hidden states $H \in \mathbb{R}^{T\times d}$ over T tokens (visual + text). The first decoded token is

$$y_1 = \arg\max_v \left[W_o\, h_T\right]_v, \tag{1}$$

where h_T is the last hidden state and W_o is the language-model head. Decode-phase compression, for example KV eviction $\mathcal{E}$ acting on the cached $\{(K_\ell, V_\ell)\}$, takes effect only when computing $y_2, y_3, \ldots$ from the now-reduced cache. For benchmarks scored only on the first token, such as a multiple-choice letter or a yes/no token, the score depends only on y_1. Thus decode-phase $\mathcal{E}$ has no effect on these scores by construction. Density-based compression operates on *both* prefill and decode, with measurable gains in each phase (Sec. 5 and Sec. 6). To verify the two phases independently, we partition benchmarks into two diagnostic roles based on their scoring rule. A benchmark is a *prefill probe* if its score depends only on y_1, as in a multiple-choice letter, a yes/no token, or any single-letter argmax over a fixed set such as MMBench, POPE, and ScienceQA. Such a probe isolates prefill quality and is structurally blind to any decode-phase change. A benchmark is a *full-pipeline probe* if its score depends on the decoded sequence $y_1, y_2, \ldots$, including free-form answers scored by exact-match, ANLS edit distance, or an LLM judge such as TextVQA, DocVQA, and MM-Vet. Such a probe reveals both phases and their interaction effects. The split is a property of the scoring rule, not the literal answer length during generation.

3.2 Empirical proof

We instantiate $\mathcal{E}$ ranging from D75 (keep 25% of visual KV) to DALL (drop *all* visual KV). Table 1 reports prediction mismatch % relative to the no-compression baseline.

Implication. The pattern is clear: TextVQA's 45.58% mismatch under DALL (Tab. 1) refutes the "tied logits" objection, and Appendix N extends the cross-model grid to compare how short-

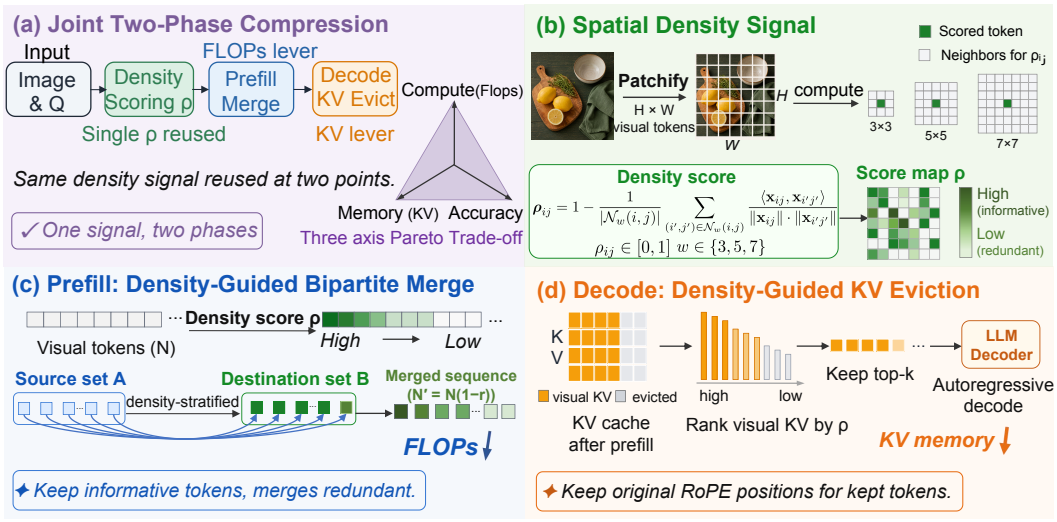


Figure 3: **TwinPrune pipeline.** A single spatial density signal ρ, defined as each visual token's average cosine dissimilarity to its $w \times w$ neighborhood (b), drives both phases. **Prefill** bipartite-merges low-ρ source tokens into high-ρ destination anchors, cutting FLOPs (c). **Decode** ranks visual KV by ρ and keeps the top-k with original RoPE positions, cutting KV memory (d). One signal, two levers, and a three-axis Pareto trade-off over compute, KV, and accuracy (a). Quantitative prefill results use the FPS/drop variant in Sec. 4.2.

answer and full-sequence probes register both phases. Any claim of decode-phase token pruning that preserves accuracy validated only on MMBench, POPE, or ScienceQA must be re-examined; future VLM token-pruning evaluation should include at least one full-pipeline-probe benchmark such as MM-Vet, DocVQA, or LLaVA-Bench-Wilder.

4 Density-Guided Two-Phase Compression

4.1 Spatial density signal

For visual tokens arranged on a grid of size $H \times W$, define density at position (i, j) via local cosine dissimilarity:

$$\rho_{ij} = 1 - \frac{1}{|\mathcal{N}_w(i,j)|} \sum_{(i',j') \in \mathcal{N}_w(i,j)} \frac{\langle x_{ij}, x_{i'j'} \rangle}{\|x_{ij}\| \cdot \|x_{i'j'}\|}, \tag{2}$$

where $\mathcal{N}_w(i, j)$ is the spatial $w \times w$ neighborhood (excluding self). We compute ρ at multiple scales $w \in \{3, 5, 7\}$ and take the max, capturing both fine and coarse structure. High ρ_{ij} indicates that the token differs from its neighbors and is therefore informative, whereas low ρ_{ij} indicates redundancy.

Why density? Density is an *intrinsic, query-invariant* feature-tensor statistic. It is motivated by kernel-entropy statistics [21], contrasting the ℓ_2-norm and the query-dependent attention scores. Section 5.3 finds competitive density-based quality at prefill and decode (information-geometric framing and the failed QC-density extension are in Appendix F).

Coreset reading. A coreset interpretation connects visual-token coverage with approximate kernel aggregation, motivating a shared geometric score for prefill selection and decode eviction. The weighted scalar-kernel discussion in Appendix H does not establish attention, logit, or accuracy guarantees for the actual unweighted operators.

4.2 Prefill: density-tilted farthest-point selection

At a chosen insertion layer L during prefill, we keep $k = \max(1, \text{int}(N \cdot (1-r)))$ of the N visual tokens by a density-tilted greedy selection. Compute density ρ on the layer-L hidden states via Eq. 2,

normalize to $\tilde{\rho} \in [0, 1]$, and pre-compute the pairwise cosine distance matrix $D_{ij} = 1 - \langle \hat{x}_i, \hat{x}_j \rangle$ where $\hat{x} = x/\|x\|$. Seed with $s_0 = \arg\max_i \rho_i$, the most informative token, and maintain $\mathrm{mind}_j = D_{s_0 j}$ as the minimum distance from j to the selected set. For each subsequent step, pick

$$s_t \;=\; \arg\max_{j \notin S} \big(\mathrm{mind}_j \;+\; \beta\, \tilde{\rho}_j \big), \quad \text{then} \quad \mathrm{mind}_j \leftarrow \min(\mathrm{mind}_j,\, D_{s_t j}), \tag{3}$$

with $\beta = 0.3$ throughout. Selected indices are sorted to preserve the visual-token-grid ordering, and KV-cache rows for the unselected tokens are dropped before the next layer. Eq. 3 unifies two greedy heuristics for the same coreset, as discussed in Sec. 4.1. Pure farthest-point sampling at $\beta{=}0$ is related to DivPrune [1], with different initialization; density top-k is the limit $\beta{\to}\infty$. The small additive $\beta = 0.3$ tilt lets coverage dominate at high prune ratios while density breaks ties when mind_j saturates, which is the regime where pure FPS picks isolated outliers with low local relevance.

Implementation. The k-step inner loop is the latency-critical path of density-tilted FPS. A naive cuBLAS-plus-graph realisation issues five kernel launches per step, and at our operating point ($N{=}3072$, $D{=}4096$, $K{=}614$) the per-launch overhead dominates the actual compute. We fuse the body into two Triton kernels, an argmax reduction and an inner-iter mind update with a $32{\times}64$ row-column tile, and back the recurrence with an fp16 normalized-token buffer. fp16's 10-bit mantissa resolves 10^{-3} differences in cosine similarity, comfortably finer than the random-vector cosine standard deviation $1/\sqrt{D}{=}0.016$ at $D{=}4096$. The full recurrence is captured into a CUDA graph keyed on (N, D, k, β), replayed once per layer rather than re-launched per step, and cached at module scope across the sweep. Without this cache, the per-call capture overhead of roughly 80 ms inverts the legacy-versus-graph latency ranking at our K. The combination delivers a $3.62\times$ generation speedup at the B3 cell on InternVL-3.5-8B, with cached-Triton fp16 TwinPrune at 160.5 ms per sample versus the 581.2 ms uncompressed baseline.

4.3 Decode: density-guided KV eviction

After prefill completes, with or without prefill-stage compression, evict visual KV entries by density rank in a single shot. We keep $k_{\text{dec}} = \max(1, \mathrm{int}(N_{\text{post}} \cdot r_{\text{decode}}))$ entries with the highest ρ, where N_{post} is the post-prefill visual-token count; zero keep uses a separate drop-all branch. Surviving KV rows retain their original RoPE-rotated keys without rotation re-application, and the first decoded token is fed back at position original_prefill_length to maintain query and key alignment. Density is reused from prefill when both phases are active, so the decode stage adds no extra similarity computation; standalone decode uses projected-input features. Eviction is one-shot at the prefill-decode boundary, in contrast to streaming approaches that maintain a sliding window during generation. The choice is motivated by the coreset reading in Sec. 4.1, rather than a guarantee on every subsequent attention aggregate.

4.4 Joint and the three-axis Pareto

Joint compression composes prefill selection with decode eviction. The resulting design space is naturally three-dimensional: (i) prefill FLOPs (reduced by selection), (ii) KV memory (reduced by eviction), (iii) accuracy. Single-phase methods occupy the axes, and joint configurations fill the interior. We study these trade-offs in Sections 5 and 6, and Algorithm 1 in Appendix A gives the full pseudocode.

5 Prefill compression Pareto

5.1 Setup

We evaluate six model variants from three VLM families: InternVL-3.5 at 2B, 4B, 8B, and 38B [27], Qwen3-VL at 4B [3], and LLaVA-1.5-7B [14].[3] Prefill-probe benchmarks (MMBench [15], POPE [12], ScienceQA [18]) provide short-answer probes of prefill quality. Full-pipeline-probe benchmarks reveal both phases. The full-pipeline-probe set comprises TextVQA [24] ($n{=}5000$, soft VQA score), MM-Vet [32] ($n{=}218$, free-form, cached Doubao judge), and DocVQA ($n{=}5349$,

[3]Primary 8B: OpenGVLab/InternVL3_5-8B-Instruct (`27506812`), BF16, seed 42, one 448-pixel tile; MMBench/POPE/ScienceQA use 1292/6000/2017 examples. Accuracy uses legacy FPS; cached Triton timing can select different tokens and includes vision, prefill and up to four generated tokens. TPL uses 36 layers and method-specific integer budgets.

Table 2: Per-benchmark prefill compression on **InternVL-3.5-8B**; same (L, r) within each block. Bold marks per-column best. Block 4 in App. V. **Speedup**: measured generation speedup vs. uncompressed baseline at batch=8, HD mode (12 tiles, N_{vis}=3072) on RTX A6000; protocol and backend details in Sec. 5.1.

Method	Venue	TPL %	MMBench	POPE	SQA	TextVQA	Avg-4	Speedup
Baseline (no compression)		100.00	0.8707	0.8903	0.9539	0.6493	0.8411	1.00×
Block 1: $L = 15$*, prune* 50%								
Random	–	72.22	0.8506	0.8870	0.9276	0.5009	0.7915	1.38×
ToMe	ICLR'23	72.22	0.8560	0.8927	0.9355	0.5497	0.8085	1.24×
FastV	ECCV'24	72.22	0.8568	0.8872	0.9365	0.5319	0.8031	1.38×
PruMerge	ICCV'25	72.22	**0.8615**	0.8937	0.9430	0.6070	0.8263	1.30×
DivPrune	CVPR'25	72.22	0.8591	**0.8942**	0.9430	0.6176	0.8285	1.28×
DART	EMNLP'25	72.22	**0.8615**	0.8895	0.9410	0.6178	0.8274	1.37×
TwinPrune	–	72.22	0.8584	0.8913	**0.9455**	**0.6324**	**0.8319**	1.35×
Block 2: $L = 3$*, prune* 85%								
Random	–	24.65	0.7577	0.8292	0.7913	0.2504	0.6571	3.37×
ToMe	ICLR'23	24.65	0.7740	0.8570	0.8106	0.3198	0.6904	2.30×
FastV	ECCV'24	24.65	0.7477	0.8683	**0.8374**	0.1703	0.6559	3.38×
PruMerge	ICCV'25	24.65	0.7763	0.8473	0.7992	0.2611	0.6710	2.73×
DivPrune	CVPR'25	24.31	**0.7933**	**0.8685**	0.8349	0.4599	0.7392	3.27×
DART	EMNLP'25	24.31	0.7701	0.8265	0.8369	0.4759	0.7274	3.35×
TwinPrune	–	24.31	0.7887	0.8628	0.8299	**0.4919**	**0.7433**	3.32×
Block 3: $L = 0$*, prune* 80%								
Random	–	22.53	0.7926	0.8427	0.8101	0.2694	0.6787	3.68×
ToMe	ICLR'23	22.53	0.7964	0.8710	0.8215	0.3191	0.7020	2.45×
FastV	ECCV'24	22.53	0.7732	0.8713	0.8359	0.2128	0.6733	3.68×
PruMerge	ICCV'25	22.53	0.7678	0.8565	0.8091	0.2192	0.6632	2.96×
DivPrune	CVPR'25	22.15	0.7933	0.8717	0.8354	0.4386	0.7348	3.51×
DART	EMNLP'25	22.15	0.7918	0.8632	**0.8562**	0.4648	0.7440	3.67×
TwinPrune	–	22.15	**0.8150**	**0.8747**	0.8438	**0.4740**	**0.7519**	3.62×
Block 3b: + LoRA recovery — $L = 0$*, prune* 70%								
Random + LoRA	–	32.02	0.8189	0.8722	0.8458	0.3331	0.7175	2.78×
DivPrune + LoRA	CVPR'25	32.02	**0.8274**	**0.8872**	0.8696	0.5045	0.7722	2.59×
DART + LoRA	EMNLP'25	32.02	0.8220	0.8790	**0.8974**	**0.5411**	**0.7849**	2.75×
TwinPrune + LoRA	–	31.64	0.8212	0.8828	0.8602	0.5351	0.7748	2.71×

deterministic ANLS scoring). On the prefill side we compare density-tilted selection (TwinPrune) against DivPrune [1], DART [28], FastV [6], ToMe [4], and PruMerge [23]. On the decode side, we compare density (ours) against visual-span KV-eviction baselines (H2O and SnapKV use proxy adaptations). We compare the prefill settings in Table 2 and decode keep $\in \{50, 25, 12.5\}\%$; the diagnostic additionally tests zero keep.

Configuration shorthand. We refer to specific compression configurations by short codes throughout. `TwinPrune _L`L`_`r denotes prefill-only TwinPrune compression inserted at LLM layer L with visual-token prune ratio $r\%$, so `TwinPrune _L3_50` is prefill-only compression at layer 3 pruning 50% of visual tokens. `decode_only_D`d denotes decode-only KV eviction that drops $d\%$ of the visual KV span after prefill, so `D75` keeps 25% of visual KV, `D875` keeps 12.5%, and `Dall` drops every visual KV. `joint_P`p`_D`d denotes joint compression that combines prefill-selection prune $p\%$ at layer 3 with decode-eviction keep $(100-d)\%$, so `joint_P50_D75` is layer-3 prefill prune 50% paired with decode keep 25%.

5.2 Prefill compression Pareto

Table 2 reports per-benchmark accuracies on InternVL-3.5-8B under a fair-comparison protocol where each block fixes a single (L, r) pair shared by all methods, comparing selection rules at the same nominal ratio. Three blocks in the main paper span the TPL spectrum. Block B1 uses $L=15$ with prune 50% and TPL$\approx 72\%$. Block B2 uses $L=3$ with prune 85% and TPL$\approx$ 24–25%,

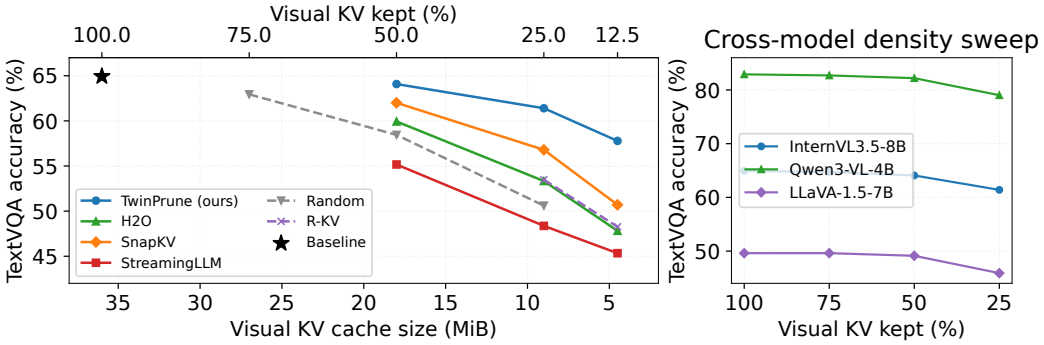


Figure 4: **Decode KV Eviction on TextVQA.** On InternVL-3.5-8B, density-guided eviction leads the compared policies at shared keep ratios in the memory–accuracy trade-off. Right inset: Qwen3-VL-4B, LLaVA-1.5-7B, and InternVL-3.5-8B show similar density trends.

an extreme stress setting. Block B3 uses $L = 0$ with prune 80% and TPL$\approx$ 22%, a visual-only pure-prune setting. The saturated high-TPL block B4 at $L = 20$, prune 50%, TPL$\approx$ 79% is reported in Appendix V. The matched cross-architecture sweep on Qwen3-VL-4B uses the same four-block schedule with M-RoPE 3-axis position encoding and is reported in Appendix Q as Tab. 17.

Headline. TwinPrune leads Avg-4 in every training-free block on InternVL-8B. The lead is $+0.34$pp at the moderate B1 ($L = 15$, prune 50%) and grows to $+0.79$pp over the next best method at the visual-only B3 ($L = 0$, prune 80%), where DivPrune and DART now produce valid outputs. Under the fair LoRA control at nominal prune 70% (B3b), TwinPrune+LoRA reaches Avg-4 $= 0.7748$ vs DivPrune+LoRA at 0.7722, with a $+3.06$pp gap on TextVQA; DART+LoRA leads Avg-4 at 0.7849. Block-by-block reading and the matched Qwen3-VL-4B sweep are in Appendix D. Selection-side complexity is in Appendix B.

The aggregate lead masks an uneven task response. At B3, POPE falls by 1.57pp from the uncompressed model, whereas TextVQA falls by 17.53pp. Thus a competitive Avg-4 does not imply harmless compression on free-form answers. B3b should also be read within its own budget: because it prunes 70% rather than B3's 80%, comparing these blocks does not isolate LoRA's contribution. The matched B3b rows are the appropriate recovery comparison.

5.3 Signal ablation

Is the advantage of our method really density, or could a simpler score match it? Holding every pipeline component fixed (L=15, prune 50%, InternVL-3.5-8B, prune-only top-k) and swapping only the ranking score across four benchmarks (TextVQA with n=5000, MMBench, V*Bench, MM-Vet), density is competitive but does not lead all four: it outperforms L2 norm on TextVQA by 2.54 percentage points (0.6323 vs. 0.6069), V-projection norm on TextVQA by 3.71 percentage points, and V-norm on V*Bench by 1.57 percentage points. MMBench scores cluster near 0.86 although prefill selection can change their predictions (Sec. 3). Multi-scale density is nearly indistinguishable from single-scale, differing by 0.13 percentage points on TextVQA, and density$\times$L2 slightly leads TextVQA (0.6335 vs. 0.6323), while L2 leads MMBench. Full four-benchmark table in Appendix I.

6 Decode compression on full-pipeline-probe benchmarks

6.1 Decode KV eviction Pareto (Figure 4)

Density-guided eviction outperforms random selection by 10.79 percentage points at aggressive (75%) eviction on InternVL-3.5-8B. Figure 4 shows cross-architecture variation; the prefill signal ablation (Sec. 5.3) shows a 2.54 percentage point lead over L2-norm on TextVQA.

Comparison with published decode-eviction baselines. Table 3 extends the comparison to H2O [33], SnapKV [13], and StreamingLLM [29] on the visual-KV span at three keep ratios. Density

Table 3: Decode-phase KV eviction comparison on InternVL-3.5-8B at three keep ratios. Density top-k wins all three TextVQA and two MM-Vet cells. On DocVQA, SnapKV leads at all three keeps. Bold marks per-column best. **TextVQA** uses n=5000 soft VQA scores. **MM-Vet** uses n=218 cached Doubao judgments. **DocVQA** uses n=5349 with ANLS scoring.

	TextVQA			MM-Vet			DocVQA		
Method (keep k)	50%	25%	12.5%	50%	25%	12.5%	50%	25%	12.5%
Baseline (no eviction)		0.649			0.749			0.541	
StreamingLLM [29]	0.552	0.484	0.453	0.629	0.571	**0.530**	0.399	0.343	0.303
H2O proxy [33]	0.599	0.533	0.478	0.634	0.570	0.503	0.486	0.401	0.335
SnapKV proxy [13]	0.620	0.568	0.507	0.666	0.597	0.509	**0.511**	**0.445**	**0.375**
Density (ours)	**0.641**	**0.614**	**0.578**	**0.687**	**0.606**	0.524	0.474	0.400	0.348

top-k wins all three TextVQA cells and two MM-Vet cells, and the advantage *widens* at extreme compression: at keep = 12.5% density beats H2O by 9.96, 2.06, and 1.28pp on TextVQA, MM-Vet, and DocVQA respectively. SnapKV leads density on DocVQA at all three keeps, and StreamingLLM leads MM-Vet at 12.5% keep. Additional decode-signal comparisons are in Appendix I.

At 50% keep, density is 0.85pp below the TextVQA control. Its advantage over H2O grows from 4.15pp to 9.96pp as keep decreases to 12.5%. The DocVQA ordering shows that geometric non-redundancy does not identify the best subset for every task. Since the policies share a visual-KV budget, the differences reflect which entries survive eviction, rather than a larger retained cache.

6.2 Joint compression analysis

Per-model MM-Vet scores under `TwinPrune _L3_50`, `decode_only_D75`, and `joint_P50_D50` are in Tab. 10 (Appendix J); both single phases cost 7 to 15pp, with no free lunch. Joint configurations are legitimate budget trades: at matched 8B TextVQA visual-KV budgets, joint P50+D50 scores 0.6040 versus decode-only D75 at 0.6140. The historical 3×3 grid (Tab. 12, Appendix L) and complementary signal sweep (Appendix K) are provided as additional context.

At this matched cache budget, joint compression trades 1.00pp of TextVQA score for reducing the visual sequence during prefill as well. Tokens removed at that stage cannot be restored by a later decode ranking. In this comparison, decode-only preserves more quality when prefill compute is affordable; joint compression additionally reduces prefill work when both compute and cache capacity are constrained.

7 Discussion and Conclusion

Summary and deployment. VLM token pruning has been evaluated on a benchmark mix that limits visibility into decode-phase effects. We provide the proof, propose a unified density-based framework that scores token importance for both phases, and supply a three-axis Pareto across six model variants and three full-pipeline-probe benchmarks. We further compare selection methods on multi-axis RoPE backbones, and we find smaller compression loss in four of six matched 8B-to-38B comparisons. In deployment, latency-bound use cases use prefill compression (3.62× generation speedup at the B3 cell), memory-bound use cases use decode eviction, and joint configurations apply when both axes are constrained.[4]

Limitations. We identify two principled limitations. (i) Single-stage prefill compression, with multi-stage scheduling unexplored. (ii) Spatial-grid density assumes regular tokenization, so non-grid tokenizers such as Q-Former require alternative formulations.

Broader impact and future work. Efficient VLM serving lowers energy use and enables edge deployment. Compression can subtly alter behavior on long-tail or safety-critical inputs without changing prefill-probe scores, so high-stakes deployments must validate on full-pipeline-probe benchmarks. Future work: learning the optimal density-aggregation kernel, and task-conditioned signals beyond pre-attention geometry (Appendix F).

[4]Compute: NVIDIA A6000 (48 GB) cluster, with the 38B model sharded across two A6000s. The original compute estimate was approximately 2,000 to 3,000 GPU-hours, and exploratory runs (Appendix F) add a comparable unreported budget. Code, predictions, and scripts will be released upon publication.

Acknowledgments and Disclosure of Funding

This work was supported by Huawei Explore X funds. This work was also sponsored by AI Hundred Schools Program and was carried out using the Ascend AI technology stack.

Competing interests. The authors declare no additional competing interests beyond the affiliations and funding disclosed above.

References

[1] Saeed Ranjbar Alvar, Gursimran Singh, Mohammad Akbari, and Yong Zhang. DivPrune: Diversity-based visual token pruning for large multimodal models. In *CVPR*, 2025. arXiv:2503.02175.

[2] Olivier Bachem, Mario Lucic, and Andreas Krause. Practical coreset constructions for machine learning. *arXiv preprint arXiv:1703.06476*, 2017.

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

[4] Daniel Bolya, Cheng-Yang Fu, Xiaoliang Dai, Peizhao Zhang, Christoph Feichtenhofer, and Judy Hoffman. Token merging: Your ViT but faster. In *ICLR*, 2023.

[5] Zefan Cai, Wen Xiao, Hanshi Sun, Cheng Luo, Yikai Zhang, Ke Wan, Yucheng Li, Yeyang Zhou, Li-Wen Chang, Jiuxiang Gu, et al. R-kv: Redundancy-aware kv cache compression for reasoning models. In *NeurIPS*, 2025.

[6] 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 *ECCV*, 2024.

[7] Dan Feldman and Michael Langberg. A unified framework for approximating and clustering data. *Proceedings of the 43rd ACM Symposium on Theory of Computing (STOC)*, pages 569–578, 2011. arXiv:1106.1379.

[8] Suyu Ge, Yunan Zhang, Liyuan Liu, Minjia Zhang, Jiawei Han, and Jianfeng Gao. Model tells you what to discard: Adaptive KV cache compression for LLMs. In *ICLR*, 2024.

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

[10] Michael Langberg and Leonard J. Schulman. Universal ε-approximators for integrals. In *Proceedings of the 21st Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 598–607, 2010.

[11] Wentong Li, Yuqian Yuan, Jian Liu, Dongqi Tang, Song Wang, Jie Qin, Jianke Zhu, and Lei Zhang. TokenPacker: Efficient visual projector for multimodal LLM. *International Journal of Computer Vision*, 133:6794–6812, 2025.

[12] Yifan Li, Yifan Du, Kun Zhou, Jinpeng Wang, Xin Zhao, and Ji-Rong Wen. Evaluating object hallucination in large vision-language models. In *EMNLP*, 2023.

[13] Yuhong Li, Yingbing Huang, Bowen Yang, Bharat Venkitesh, Acyr Locatelli, Hanchen Ye, Tianle Cai, Patrick Lewis, and Deming Chen. Snapkv: Llm knows what you are looking for before generation. *NeurIPS*, 2024.

[14] Haotian Liu, Chunyuan Li, Yuheng Li, and Yong Jae Lee. Improved baselines with visual instruction tuning. In *CVPR*, 2024.

[15] Yuan Liu, Haodong Duan, Yuanhan Zhang, Bo Li, Songyang Zhang, Wangbo Zhao, Yike Yuan, Jiaqi Wang, Conghui He, Ziwei Liu, et al. Mmbench: Is your multi-modal model an all-around player? In *ECCV*, 2024.

[16] Zirui Liu, Jiayi Yuan, Hongye Jin, Shaochen Zhong, Zhaozhuo Xu, Vladimir Braverman, Beidi Chen, and Xia Hu. KIVI: A tuning-free asymmetric 2bit quantization for KV cache. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pages 32332–32344. PMLR, 2024.

[17] Zuyan Liu, Benlin Liu, Jiahui Wang, Yuhao Dong, Guangyi Chen, Yongming Rao, Ranjay Krishna, and Jiwen Lu. Efficient inference of vision instruction-following models with elastic cache. In *ECCV*, 2024. arXiv:2407.18121.

[18] 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. *NeurIPS*, 2022.

[19] Baharan Mirzasoleiman, Ashwinkumar Badanidiyuru, Amin Karbasi, Jan Vondrák, and Andreas Krause. Lazier than lazy greedy. In *Proceedings of the AAAI Conference on Artificial Intelligence*, 2015. arXiv:1409.7938.

[20] Miloš Radovanović, Alexandros Nanopoulos, and Mirjana Ivanović. Hubs in space: Popular nearest neighbors in high-dimensional data. *Journal of Machine Learning Research*, 11:2487–2531, 2010.

[21] Alfréd Rényi. On measures of entropy and information. *Proc. 4th Berkeley Symp. on Mathematical Statistics and Probability*, 1961.

[22] Ozan Sener and Silvio Savarese. Active learning for convolutional neural networks: A core-set approach. In *International Conference on Learning Representations (ICLR)*, 2018. arXiv:1708.00489.

[23] Yuzhang Shang, Mu Cai, Bingxin Xu, Yong Jae Lee, and Yan Yan. LLaVA-PruMerge: Adaptive token reduction for efficient large multimodal models. In *ICCV*, 2025. arXiv:2403.15388.

[24] Amanpreet Singh, Vivek Natarajan, Meet Shah, Yu Jiang, Xinlei Chen, Dhruv Batra, Devi Parikh, and Marcus Rohrbach. Towards vqa models that can read. In *CVPR*, 2019.

[25] Jiaming Tang, Yilong Zhao, Kan Zhu, Guangxuan Xiao, Baris Kasikci, and Song Han. QUEST: Query-aware sparsity for efficient long-context LLM inference. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pages 47901–47911. PMLR, 2024.

[26] Zhongwei Wan, Ziang Wu, Che Liu, Jinfa Huang, Zhihong Zhu, Peng Jin, Longyue Wang, and Li Yuan. LOOK-M: Look-once optimization in KV cache for efficient multimodal long-context inference. In *Findings of the Association for Computational Linguistics: EMNLP 2024*, pages 4065–4078, Miami, Florida, USA, November 2024. Association for Computational Linguistics.

[27] Weiyun Wang, Zhangwei Gao, Lixin Gu, Hengjun Pu, Long Cui, Xingguang Wei, Zhaoyang Liu, Linglin Jing, Shenglong Ye, Jie Shao, et al. Internvl3. 5: Advancing open-source multimodal models in versatility, reasoning, and efficiency. *arXiv preprint arXiv:2508.18265*, 2025.

[28] Zichen Wen, Yifeng Gao, Shaobo Wang, Junyuan Zhang, Qintong Zhang, Weijia Li, Conghui He, and Linfeng Zhang. Stop looking for "important tokens" in multimodal language models: Duplication matters more. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pages 9961–9980, Suzhou, China, November 2025. Association for Computational Linguistics.

[29] Guangxuan Xiao, Yuandong Tian, Beidi Chen, Song Han, and Mike Lewis. Efficient streaming language models with attention sinks. In *ICLR*, 2024.

[30] 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 *CVPR*, 2025. arXiv:2412.04467.

[31] Weihao Ye, Qiong Wu, Wenhao Lin, and Yiyi Zhou. Fit and prune: Fast and training-free visual token pruning for multi-modal large language models. In *AAAI*, 2025. arXiv:2409.10197.

[32] Weihao Yu, Zhengyuan Yang, Linjie Li, Jianfeng Wang, Kevin Lin, Zicheng Liu, Xinchao Wang, and Lijuan Wang. MM-Vet: Evaluating large multimodal models for integrated capabilities. In *ICML*, 2024.

[33] Zhenyu Zhang, Ying Sheng, Tianyi Zhou, Tianlong Chen, Lianmin Zheng, Ruisi Cai, Zhao Song, Yuandong Tian, Christopher Ré, Clark Barrett, et al. H2o: Heavy-hitter oracle for efficient generative inference of large language models. *NeurIPS*, 2023.

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 Sec. 1 contributions list four claims that match the paper body: the prefill-probe versus full-pipeline-probe diagnostic (Sec. 3), the unified spatial-density signal driving both phases (Sec. 4), the three-axis Pareto study across seven model variants and three full-pipeline-probe benchmarks (Sec. 5 and Sec. 6), and the cross-architecture and cross-scale robustness finding (Sec. P).

   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: The Discussion section contains a dedicated Limitations paragraph (Sec. 7) that lists two principled limitations: single-stage prefill compression with multi-stage scheduling unexplored, and the spatial-grid density assumption that requires regular tokenization and therefore does not directly apply to Q-Former-style projectors. Theorem-level limitations of the coreset bounds are further discussed in Appendix H, and the negative DocVQA result on Qwen3-VL-4B is reported without hedging in Appendix R.

   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: Theorem 1 (k-center coreset bound), Corollary 1 (spatial-chain reach), and Theorem 2 (Feldman-Langberg-style $1/\sqrt{k}$ rate) are stated in Sec. 4.1, with full assumptions A1 through A5 and complete proofs in Appendix H. A subsection in the same appendix lists the load-bearing scope conditions and acknowledges the regimes in which the bounds are non-trivial.

   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: Sec. 5.1 lists the seven model variants, the four prefill-probe and three full-pipeline-probe benchmarks, the prune-ratio and insertion-layer sweep grids, and the decode keep-ratio grid. Appendix A provides the full unified-density pseudocode. Per-method hyperparameters are stated inline at first use. Code, configs, and result JSONs will be released upon publication.

   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: All pretrained backbones (InternVL-3.5, Qwen3-VL, LLaVA-1.5) and all benchmarks (MMBench, POPE, ScienceQA, TextVQA, MM-Vet, DocVQA) are publicly available under their original distributions. We will release all code and models in the future.

   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: Sec. 5.1 states the validation splits used (TextVQA n=5000, MM-Vet n=218, DocVQA n=5349, full MMBench, full POPE, full ScienceQA), the prefill insertion layers $L \in \{0, 1, 2, 3, 5, 10, 15, 20\}$, prefill prune ratios $r \in \{25, 50, 75\}\%$, and decode keep ratios $\{75, 50, 25, 12.5, 6.25, 0\}\%$. The optional LoRA recovery recipe with rank, alpha, n samples, and training time is documented in Appendix U. All compression methods are training-free; the LoRA adapter is the only trained component and uses identical hyperparameters across methods for fair comparison.

   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: [Yes]

   Justification: Please refer to the experiment part.

   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: The Discussion footnote in Sec. 7 reports the cluster (NVIDIA A6000 with 48 GB, with the 38B model sharded across two A6000s), the total reported-experiment budget of approximately 2,000 to 3,000 GPU-hours, and a comparable unreported exploratory-run budget. Selection-side latency is further measured at high resolution and reported in Appendix B, where density top-k is shown to be 40 to 50 times faster than the next-best signal at N_{vis}=3072.

   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 work studies inference-time efficiency of pretrained vision-language models. It does not collect new training data, does not involve human subjects, does not introduce new pretrained model weights, and uses publicly distributed benchmarks under their original licenses. We have reviewed the NeurIPS Code of Ethics and confirm conformance.

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: [Yes]

Justification: The Discussion section (Sec. 7) contains a Broader-impact paragraph that highlights positive impacts (lower energy use, edge deployment) and a deployment caveat: compression can subtly alter behavior on long-tail or safety-critical inputs without changing prefill-probe scores, so high-stakes deployments must validate on full-pipeline-probe benchmarks before serving.

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: The released artifacts are a training-free compression algorithm and its evaluation harness. We do not release new pretrained model weights, scraped datasets, or generative models with high misuse risk. The underlying VLM backbones (InternVL-3.5, Qwen3-VL, LLaVA-1.5) are distributed by their original authors with their own usage terms and safeguards, which we do not modify.

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: All used assets are cited at first use: VLM backbones InternVL-3.5 [27], Qwen3-VL [3], and LLaVA-1.5 [14]; benchmarks MMBench [15], POPE [12], ScienceQA [18], TextVQA [24], and MM-Vet [32]; prior compression methods FastV [6], ToMe [4], DivPrune [1], DART [28], PruMerge [23], H2O [33], SnapKV [13], StreamingLLM [29], VisionZip [30], and R-KV [5]. We use each asset under its published license; no asset is redistributed or repackaged.

    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: [Yes]

    Justification: The new assets in this work are the compression-engine code, evaluation scripts, and per-cell result JSONs. They will be released upon publication with a top-level README documenting directory layout, environment setup, and exact commands for reproducing each table, together with per-method docstrings for the unified density signal, bipartite merge, and decode-phase eviction routines.

    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 work does not involve crowdsourcing or research with human subjects. All evaluation uses publicly distributed benchmarks with their own scoring rules.

    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 work does not involve human subjects, so IRB review is not applicable.

    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: [N/A]

    Justification: Large language models in this paper appear only as the subjects of compression (the language-model component of the studied VLMs) and as the standard MM-Vet free-form judge, which is a benchmark scoring rule rather than a method component. The core method development (spatial-density signal, bipartite merge, top-k KV eviction, coreset analysis) does not use LLMs as an algorithmic component or design tool.

    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.

Appendix scope and numeric audit

The primary tables in the main paper are the corrected single-checkpoint comparisons. This supplement adds complete measured signal, scale and timing campaigns. The remaining tables and empirical plots are retained for private author review as historical records, with explicit status markers. They are not a certificate that every number in the original arXiv paper has been reproduced. The complete old-to-new ledger, including all pending tables, is `docs/PAPER_NUMERIC_CHANGES.md` in the accompanying private repository. Historical captions and interpretations under the marked sections do not support the revised main-paper conclusions.

The corrected tables separate the Instruct checkpoint from other same-width variants, soft VQA scores from exact match, Doubao judgments from the official MM-Vet evaluator, and FPS/drop accuracy from cached-Triton timing. The original arXiv targets and tolerances remain unchanged in their archival manifest.

A Measured density pipeline

Algorithm 1 Measured density-tilted FPS/drop with optional exact cached-score KV eviction.

Require: Visual hidden states X after layer L, prefill budget k, optional joint budget k_d.
Compute spatial density ρ by Eq. 2; normalize to $\tilde{\rho}$.
Seed S with $\arg\max_i \rho_i$; initialize minimum cosine distances to that seed.
while $|S| < k$ **do**
  Pick the unselected index maximizing minimum distance $+\beta\tilde{\rho}$.
  Update minimum distances to include the new index; add it to S.
end while
Sort S; retain visual states and matching visual KV rows; continue prefill.
if joint compression **then**
  Select the top k_d rows using the exact cached scores $\rho[S]$.
  Trim visual KV only; preserve non-visual cache entries and generate the continuation.
end if

Primary prefill comparisons use legacy position handling. Standalone decode density is computed from projected visual input features; exact joint reuse uses the layer-L tensor. These paths are not silently identified with the original bipartite-merge illustration or the older mixed-operator joint grid.

B Selection-side complexity

Fixed-window spatial-density pooling has linear token-count cost at fixed feature width. Density-tilted FPS additionally performs k distance updates, costing $O(Nkd)$; with a fixed keep ratio its complete recurrence is not linear in N. The new synthetic selection campaign times mask construction and one-layer KV trim as well as selection, and uses synthetic single-head $K = V = X$. These boundaries do not certify the old unspecified two-orders-of-magnitude or 50-fold claims, which are withdrawn. Likewise the old 1700-token end-to-end numbers are not promoted into the corrected primary table. The complete fresh measurements are retained in the numeric audit artifacts.

C Deployment and timing boundary

The primary accuracy table uses legacy FPS. Cached Triton FPS is a separate numerical implementation, with non-identical selected sets in some synthetic comparisons. Its full 16-cell accuracy replay is also recorded separately. A timing measured with Triton cannot be attached to the legacy accuracy score as though both came from one identical selector.

The completed historical-boundary experiment uses replicated synthetic pixels and a shared prompt at batch 8 with 12 tiles. Its synchronized boundary includes the vision encoder/projector, language prefill, and up to four generated tokens. This is a fixed-shape generation benchmark, not pure-prefill timing or multi-request serving. The ratio is the mean of the two full-token guard medians divided by the trial median. The B3 density row now measures 3.622 at this boundary; this does not establish the

original 3.24 pure-prefill claim. Table 4 reports all rows, including the independent baseline guard and the actual layer-input token traces.

Table 4: Separate fixed-shape timing campaign on A6000: batch 8, replicated 12-tile image (3072 visual tokens), 5 warmups, 20 repeats. The boundary includes vision/projector, language prefill and up to four generated tokens; preprocessing is excluded. It is not pure-prefill latency or a serving workload. Cached Triton FPS is distinct from legacy FPS in the primary accuracy table. TPL is traced at this resolution and therefore differs from the one-tile accuracy budget. The baseline row is an independent guard ratio, not an additional model gain.

Configuration	ms/sample	Guard ratio	Traced TPL (%)
baseline_baseline	576.45	1.004	100.00
b1_random_drop	419.98	1.383	72.22
b1_tome	467.19	1.243	72.22
b1_fastv	421.00	1.380	72.22
b1_prumerge	446.08	1.300	72.22
b1_divprune	451.92	1.284	72.22
b1_dart	423.02	1.372	72.22
b1_cac_v4	428.56	1.353	72.22
b2_random_drop	172.74	3.367	24.45
b2_tome	252.53	2.300	24.45
b2_fastv	171.63	3.385	24.45
b2_prumerge	212.41	2.734	24.45
b2_divprune	177.70	3.273	24.45
b2_dart	173.78	3.345	24.45
b2_cac_v4	174.75	3.325	24.42
b3_random_drop	158.08	3.677	22.24
b3_tome	236.89	2.452	22.24
b3_fastv	157.78	3.685	22.24
b3_prumerge	196.02	2.965	22.24
b3_divprune	165.63	3.508	22.21
b3_dart	158.50	3.669	22.21
b3_cac_v4	160.46	3.622	22.21
b3b_random_drop	209.02	2.782	31.96
b3b_divprune	224.02	2.594	31.96
b3b_dart	211.44	2.749	31.96
b3b_cac_v4	214.66	2.708	31.93

D Reading the corrected prefill blocks

B1/B2/B3 share official Instruct base weights across all methods. TwinPrune reaches Avg-4 0.8319, 0.7433 and 0.7519, exceeding the strongest other mean by 0.34, 0.42 and 0.79pp. These are single-seed observations. For B3b, TwinPrune+LoRA scores 0.7748 while DART+LoRA scores 0.7849. All four adapters use the same training recipe; no old 3.0 adapter is used on the 3.5 backbone.

Actual retained counts and TPL replace the old nominal budget labels. The post-layer intervention gives

$$\mathrm{TPL} = \frac{(L+1)N + (36-L-1)k}{36N}.$$

A one-token rounding difference remains between some selectors. This is disclosed rather than called an exact integer-budget match. Old B2/B3 blank DivPrune/DART predictions are replaced by complete valid evaluations; no intrinsic collapse conclusion follows from those old files.

E Measured cross-scale numbers

Table 5 gives all baseline and compressed measurements, so every reported loss can be calculated against its own control. At 8B versus 38B, TextVQA joint loss increases from 4.5333 to 4.9067pp, and MM-Vet P50 loss increases from 11.5138 to 12.4771pp. The other four matched transitions improve. The claim that every transition improves is withdrawn. MM-Vet uses a separate cached

Doubao judging campaign here; its 8B baseline 0.7514 is not pooled with the independently judged 0.7486 Table 3 control.

Table 5: Complete measured scale campaign on InternVL3.5 at 2B/4B/8B/38B (8B: Instruct). P50: layer-3 FPS/drop with 50% prune; D75: decode keep 25%; P50+D50: exact cached-score joint with decode keep 50%. All use A6000, BF16, seed 42, one tile; TextVQA $n = 5000$, MM-Vet $n = 218$ with Doubao. Each scale uses its own matched control. The 8B MM-Vet control was judged in this campaign and is not pooled with the independent Table 3 judgment.

Size	Task	Baseline	P50	D75	P50+D50
2B	TextVQA	0.6305	0.6003	0.5913	0.5907
2B	MM-Vet	0.6206	0.5495	0.5073	0.5032
4B	TextVQA	0.6417	0.6103	0.6070	0.5971
4B	MM-Vet	0.6853	0.5743	0.5803	0.5408
8B	TextVQA	0.6493	0.6138	0.6140	0.6040
8B	MM-Vet	0.7514	0.6362	0.6060	0.5821
38B	TextVQA	0.7148	0.6851	0.6912	0.6657
38B	MM-Vet	0.7742	0.6494	0.6702	0.6261

F Why single-layer density is minimal-sufficient: three failed extensions

Historical material pending verification. The following numbers and interpretations are retained for private review and are not covered by the corrected primary-table evidence. They require a matching rerun, replacement or removal before final submission.

Spatial density ρ_{ij} (Sec. 4) is intentionally task-agnostic. A natural concern is that it therefore leaves task-specific information on the table. We tested the three most obvious training-free extensions, namely question-conditioning, cross-layer differencing, and phase-coupling via merge genealogy, and all three fail, each for a *different* principled reason.

Setup. All three ablations use TwinPrune at L=15, prune 50%, InternVL-3.5-8B, TextVQA. Pilots use n=200, and the one QC variant that showed a pilot gain was re-run on full validation (n=5000).

Attempt 1: Question-conditioned density (QC). We reweight each visual token by its alignment with a pooled question representation, $\rho_{ij}^{\text{QC}} = \rho_{ij} \cdot (1 + \lambda \cos(v_{ij}, f_{\text{text}}(q)))$, sweeping $\lambda \in \{0, 0.1, 0.3, 0.5, 1\} \times f_{\text{text}} \in \{L0, L15\} \times \text{pool} \in \{\text{mean,last,max}\} \times \text{sign} \in \{+, -\}$: 24 configurations. Every forward-sign variant degrades accuracy (best 0.575 at λ=0.3 vs. 0.583 baseline). The sole apparent win uses negative λ, embed-L0, last-token pool (0.598, $+1.5$pp on n=200). Re-running on the full 5000-sample val gives 0.6171 vs. 0.6148 baseline, a difference of 0.23 percentage points that lies well inside the 0.7% standard deviation at n=5000. A 587-sample diagnostic shows Pearson $r(\cos, \rho) \in [-0.66, -0.38]$ for more than 99 percent of samples across all six source-by-pool combinations: the geometry is hubness-driven [20], in that pooled question representations sit near the all-token mean direction, high-density foreground tokens deviate from it, and their cosine with the pool is systematically anti-correlated with ρ. The signal thus behaves as an anti-foreground prior, not as question-relevance.

Attempt 2: Cross-layer density drift. We hoped that $\rho_{ij}^{L_{\text{late}}} - \alpha\, \rho_{ij}^{L_{\text{early}}}$ would remove generic "spatial distinctiveness" and leave a task-relevant residual. Across $L_{\text{early}} \in \{3, 8, 10\}$ and $\alpha \in \{0.3, 0.5, 1.0\}$ (a sweep of 12 variants), every setting strictly hurts. The best setting reaches 0.557 at L_8 with α=0.3 (a 2.7 percentage-point drop), and the worst reaches 0.205 at L_8 with α=1.0 (a 38 percentage-point drop). Early-layer and late-layer densities remain positively correlated through the transformer, so subtraction cancels common useful signal rather than isolating a task-specific residual.

Attempt 3: Phase-coupled genealogy reweighting. We tried to let the decoder exploit the prefill merge's genealogy $|S_k|$ via $\text{importance}(m_k) = \rho(m_k) \cdot (1 + \beta \log(1 + |S_k|))$. Across $\beta \in \{0, 0.5, 1, 2\}$ and two joint configurations, all 8 variants return bit-identical accuracy. Our bal-

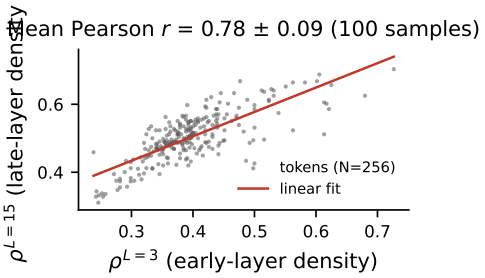


Figure 5: **Historical empirical plot, not revalidated.** Spatial density is positively correlated across transformer depth on TextVQA samples processed by InternVL-3.5-8B, with aggregate statistics shown in the plot title.

anced bipartite merge makes $|S_k| = 2^r$ for every survivor after r rounds, so any monotone function of genealogy is a global scalar and cannot change top-k.

Additional failed attempts. We also piloted two further extensions. (i) *Head-wise visual KV eviction*, in which a different top-K visual tokens is selected per attention head, costs between 6.5 and 12.5 percentage points across keep ratios, because heads agree on informativeness (correlated density in K-subspace), so the union saturates and per-head depth drops by roughly a factor of six. (ii) *Attention-density hybrid* using sink-filtered text-to-visual attention costs between 2 and 8 percentage points across all six variants, because $L{=}15$ text-to-visual attention is essentially orthogonal to density (with Pearson correlation near 0.1), so attention is not sink-dominated at this layer (max-to-mean ratios of roughly 2.0 to 2.6) and simply does not carry the same foreground notion that density does.

One lesson, five failures. All five training-free extensions along the three natural axes (modality, layer, phase, head, attention) hit geometric or structural walls. Meaningful task-awareness for VLM pruning appears to require a *trained* signal, which is orthogonal to our framework's goal of a zero-training module.

G Cross-layer density correlation

Historical material pending verification. The following numbers and interpretations are retained for private review and are not covered by the corrected primary-table evidence. They require a matching rerun, replacement or removal before final submission.

We quantify the claim that early- and late-layer densities remain positively correlated through the transformer (Attempt 2 above). On 100 TextVQA validation samples processed by InternVL-3.5-8B, the Pearson correlation between spatial density at the early and late probe layers is

$$r\big(\rho^{L=3}, \rho^{L=15}\big) \;=\; \mathbf{0.78} \pm 0.09$$

(all 100 samples are positive, with a minimum of 0.48 and a maximum of 0.88, as plotted in Figure 5). Re-using the layer-3 ρ estimate at decode time therefore introduces only a small rank-preserving error relative to recomputing ρ at the deeper layer, and avoids the cost of a second density pass. The $n{=}100$ pilot is bounded by the two-pass ($L{=}3$ and $L{=}15$) forward-hook overhead. The sample standard deviation of 0.088 is already tight enough that extending to the full 5000-sample validation set is unlikely to change the characterization.

H Two-phase density as an approximate greedy coreset

Historical material pending verification. The following numbers and interpretations are retained for private review and are not covered by the corrected primary-table evidence. They require a matching rerun, replacement or removal before final submission.

This appendix positions the density signal of Eq. 2 inside the classical coreset literature. The argument is *heuristic*, in that we do not claim a worst-case ε-bound, but it clarifies (i) why the *same* ranking

serves both prefill and decode, and (ii) how our method relates to the farthest-point strategy of DivPrune.

Coreset, formally. For a point set $X = \{x_1, \ldots, x_N\} \subset \mathbb{R}^d$, a query family $\mathcal{Q}$, and a non-negative loss $\ell : \mathcal{Q} \times X \to \mathbb{R}_{\geq 0}$, a weighted subset (S, w) with $S \subseteq X$, $|S| = k$, and $w : S \to \mathbb{R}_{>0}$ is an ε-*coreset* for $(X, \ell, \mathcal{Q})$ if for every $q \in \mathcal{Q}$,

$$\left| \sum_{x \in X} \ell(q, x) \; - \; \sum_{s \in S} w(s)\, \ell(q, s) \right| \; \leq \; \varepsilon \sum_{x \in X} \ell(q, x). \tag{4}$$

Instantiated for transformer inference, ℓ is the attention kernel $\ell(q, x) = \exp(\langle q, x \rangle / \sqrt{d})$ and the query family $\mathcal{Q}$ is the set of text-side keys arising during decode. A coreset for X_{visual} approximates all future attention aggregates simultaneously.

Sensitivity and why local density is a proxy. The Feldman and Langberg framework [7, 10] defines the *sensitivity* of $x \in X$ as $\sigma(x) = \sup_{q \in \mathcal{Q}} \ell(q, x) / \sum_{x'} \ell(q, x')$, and shows that sampling $O(d \cdot \varepsilon^{-2} \sum_x \sigma(x))$ points with probability $\propto \sigma(x)$ yields an ε-coreset with high probability. Computing σ requires the query family, which we do not have at prefill time. Suppose, however, that the visual-feature manifold is locally smooth and approximately isotropic in its tangent space. Then for a patch x surrounded by near-duplicates (low ρ_{ij}), every kernel evaluation $\ell(q, x)$ is matched by many near-equal evaluations $\ell(q, x')$ at neighbors x', so the *bucket* of near-duplicate tokens all share a single sensitivity mass: if the full bucket is replaced by its centroid, the per-bucket contribution to $\sum_{x'} \ell(q, x')$ changes only by the within-bucket kernel variation. Under this substitution, $\sigma(x)$ for any interior-bucket x is bounded by the bucket's local kernel variance, which vanishes in the isotropic limit. Conversely, high-ρ patches are isolated in the tangent space and carry near-unique sensitivity. Density ρ is therefore a query-invariant lower-bound surrogate for σ under these assumptions, and top-k-by-ρ is a deterministic-threshold counterpart of Feldman and Langberg importance sampling. We note two caveats for rigor. First, the original Feldman and Langberg guarantee is *probabilistic*, with sampling probability $\propto \sigma$, and we replace it with a deterministic threshold for reproducibility, so we do not inherit the ε-coreset guarantee automatically. Second, the substitution argument above is conditional on local isotropy holding, and where that assumption fails (for example, on cross-modal cosine geometry at shallow layers, as in Appendix F), the density-sensitivity link breaks down, consistent with our failed QC-density ablation.

Theorem 1 (approximate coreset, k-center form, small-deviation regime). *Let $X \in \mathbb{R}^{N \times d}$ be a visual-token feature matrix. Assume A1 (bounded radii: $\|q\| \leq Q$, $\|x\| \leq X_{\max}$). Let $S \subseteq X$ with $|S| = k$ and define the k-center cover radius*

$$r_k \; := \; \max_{x \in X} \; \min_{s \in S} \; \|x - s\|. \tag{5}$$

Assume A4' (k-center cover): r_k is finite. Let C1' hold: $Q r_k / \sqrt{d} \leq 1$. With Voronoi assignment $s(x) := \arg\min_{s \in S} \|x - s\|$, bucket $B_s := \{x : s(x) = s\}$, and bucket-cardinality weights $w(s) := |B_s|$, the pair (S, w) is an ε-coreset (Eq. 4) for the attention kernel $\ell(q, x) = \exp(\langle q, x \rangle / \sqrt{d})$ with

$$\varepsilon \; \leq \; \frac{2e\, Q\, r_k}{\sqrt{d}} \; \leq \; \frac{6\, Q\, r_k}{\sqrt{d}}. \tag{6}$$

Proof sketch. For $x \in B_s$, the log-Lipschitz property $\log[\ell(q, x)/\ell(q, s)] = \langle q, x - s \rangle / \sqrt{d}$ has constant $Q/\sqrt{d}$, independent of $X_{\max}$. Define $\delta := \langle q, x - s \rangle / \sqrt{d}$. Then $|\delta| \leq Q\|x - s\| / \sqrt{d} \leq Q r_k / \sqrt{d} \leq 1$ by A4' and C1'. The elementary inequality $|e^\delta - 1| \leq 2|\delta|$ on $[-1, 1]$ (provable by convexity: $f(\delta) = 2\delta - (e^\delta - 1)$ has $f(0) = 0$, $f(1) = 3 - e > 0$, and unimodal shape, with a symmetric argument for $\delta < 0$) gives $|\ell(q, x) - \ell(q, s)| \leq 2\ell(q, s) \cdot Q\|x - s\| / \sqrt{d}$. Within bucket B_s, $\ell(q, x) \in [\ell(q, s)/e,\, e\,\ell(q, s)]$, so $M_s := \sum_{x \in B_s} \ell(q, x) \geq |B_s| \ell(q, s)/e$. The per-bucket relative error is therefore bounded by $2|B_s| \ell(q, s) \cdot Q r_k / \sqrt{d} \,/\, (|B_s| \ell(q, s)/e) = 2eQ r_k / \sqrt{d}$, and summing over buckets weighted by M_s / M_X preserves this constant. □

Theorem 1 is the kernel-sum coreset analog of the active-learning core-set bound of Sener and Savarese [22], where r_k is exactly their δ_S. The statement is method-agnostic, in that it applies to $S_k = \textsc{Top}\text{-}k(\rho)$ (ours), to FPS (DivPrune [1], which 2-approximates the optimal r_k by construction), and to any selector that achieves a small cover radius.

Outside C1'. The shave breaks down, and the bound reverts to the loose universal form $\varepsilon \leq \kappa_q \cdot Q r_k/\sqrt{d}$ with $\kappa_q = e^{2QX_{\max}/\sqrt{d}}$ (still valid, but loose by roughly 10^7 at d=128 and $Q \approx X_{\max} \approx 10$).

Corollary 1 (spatial-path form for density top-k). *Suppose additionally A2 (norm-dispersion: $|\,\|x\| - \|x'\|\,| \leq \eta$ for spatial neighbors), A3 (ρ as cosine density per Eq. 2), and A4 (low-ρ spatial connectivity: every $x \in X$ has a spatial path of length $\leq D$ through tokens with $\rho \leq \rho_{(k+1)}$ to some $s \in S_k = \textsc{Top}\text{-}k(\rho)$). Then $r_k \leq D(\eta + X_{\max}\sqrt{2\rho_{(k+1)}})$, and Theorem 1 specializes to*

$$\varepsilon \;\leq\; \frac{2e\,Q\,D}{\sqrt{d}} \cdot \Big(\eta + X_{\max}\sqrt{2\rho_{(k+1)}}\Big) \;\leq\; \frac{6\,Q\,D}{\sqrt{d}} \cdot \Big(\eta + X_{\max}\sqrt{2\rho_{(k+1)}}\Big). \tag{7}$$

This recovers the spatial-path interpretation of density top-k: the cover radius is upper-bounded by the spatial chain budget $D\times$ (per-step Euclidean reach). The proof of $r_k \leq D(\eta + X_{\max}\sqrt{2\rho_{(k+1)}})$ uses the cosine-to-Euclidean identity $\|x - x'\|^2 \leq \eta^2 + 2X_{\max}^2\rho_{x,x'}$ (Cauchy–Schwarz + A2) plus a triangle-inequality chain along the spatial path.

Empirical validity (Theorem 1 with Corollary 1). On InternVL's 16×16 visual-token grids at L=15, the cosine neighborhood structure aligns with the spatial grid: the cross-layer stability $r(\rho^{L=3}, \rho^{L=15}) = 0.78$ reported in Appendix G empirically backs the assumption that the same S_k approximates the kernel sum at multiple layer depths. Density top-k achieves a small r_k via the empirical correlation between density rank and Voronoi-center suitability, rather than by direct algorithmic construction. FPS and DivPrune, by contrast, minimize r_k by construction, and both methods land in Theorem 1. Both A4 and A4' *fail* for cross-modal cosine geometry at shallow layers where hubness [20] inverts the expected local structure (Appendix F), and A4' is the load-bearing geometric assumption that breaks in that regime.

Why the same ρ serves both phases. A subset S_k that is an approximate coreset for X_{visual} remains one for any downstream computation that accesses X_{visual} only through attention aggregates, because attention is the only channel through which the cached keys and values re-enter the forward pass. Consequently the same S_k controls the prefill aggregate (where merge is a weighted replacement of the non-coreset points by their nearest coreset representative, §4) and the decode aggregate (where eviction is pure subsetting, §4). This is the heuristic reason a single ranking reused across two deployment points does not compound error: both phases consume the same coreset approximation. The cross-layer stability of ρ itself, with r=0.78 over n=100 samples in Appendix G, empirically supports the implicit assumption that the ranking computed at layer 3 remains a valid approximate coreset at layer 15.

Density vs. farthest-point: dual greedy heuristics. DivPrune [1] is farthest-point sampling (FPS) in feature space, the classical k-center heuristic [22], in which one iteratively adds the point most distant from the current subset, maximizing a minimum pairwise distance. FPS and density-top-k are *dual greedy heuristics*: FPS *maximizes diversity* of the kept set, while density-top-k *minimizes redundancy* by evicting points whose neighborhoods are already covered. On a uniform manifold the two strategies produce similar sets in practice, and they diverge when the distribution is skewed. The worst case for FPS is a dense foreground surrounded by a thin outlier halo: the halo is maximally distant and therefore selected, starving the foreground. The worst case for density is hubness, which our layer-15 single-modality use avoids (Sec. 5.3) but which strikes the query-conditioned variant (Appendix F). TwinPrune and DivPrune empirically tie within roughly 0.003 at mild compression (Table 2), and density pulls ahead at aggressive compression, consistent with the duality: neither is dominated by the other, and on finite feature sets they need not coincide pointwise. Submodular-maximization coreset algorithms [19] offer a general greedy framework that would subsume either choice given the appropriate utility function, and we leave this unification to future work.

Theorem 2 (FL-rate alternative, under sensitivity-surrogate assumption). *Suppose A1, A4', and C1' as in Theorem 1, together with two additional assumptions. A5 (density as C_σ-sensitivity surrogate): there exists $C_\sigma \geq 1$ such that $\sigma(x) \leq C_\sigma \rho(x)/\sum_{x'} \rho(x')$ for every $x \in X$ and every $q \in \mathcal{Q}$. B1 (monotone bucket-mass): residual bucket coverage is non-increasing as $|S|$ grows, which is satisfied automatically when buckets are Voronoi cells under the kernel metric. Then the*

deterministic top-k-by-ρ subset S_k inherits a Feldman and Langberg style rate

$$\varepsilon \;\le\; C_\sigma \sqrt{\frac{c\, d \log k}{k}} \;+\; \frac{2eQ\eta D}{\sqrt{d}}\,(1 + o(1)) \tag{8}$$

for an absolute constant $c \le 8$ from [7] *Theorem 3.1, with the second term the irreducible norm-dispersion residual that neither density top-k nor FL importance sampling can eliminate.*

Proof sketch. Define an auxiliary probabilistic estimator $\tilde{S}_k$ drawn as k i.i.d. samples with probability $\propto \rho(x)$, with FL importance weights $\tilde{w}(x_i) = (\sum_x \rho(x))/(k\rho(x_i))$. By A5, $\rho(x)/\sum \rho \ge \sigma(x)/(C_\sigma)$, so importance sampling $\propto \rho$ inherits FL's guarantee with the surrogate-gap constant C_σ scaling sample complexity: $k \ge c\, C_\sigma d/\varepsilon^2(\log k + \log(1/\delta))$ samples suffice for ε-coreset with probability $1-\delta$ [7]. The deterministic-vs-probabilistic bridge: B1 (monotone bucket-mass) plus a coupling-and-shifting argument shows the deterministic top-k is no worse than the expected probabilistic draw. The norm-dispersion η residual (from Corollary 1) survives both samplers and is irreducible. □

When Theorem 2 is tighter than Theorem 1. Theorem 1 gives $\varepsilon = O(\sqrt{\rho_{(k+1)}})$ in the density tail, whereas Theorem 2 gives $\varepsilon = \tilde{O}(1/\sqrt{k})$. For polynomial density decay $\rho_{(k+1)} \asymp (k/N)^{-\alpha}$, Theorem 1 yields $\varepsilon \asymp (k/N)^{-\alpha/2}$, which is sharper than Theorem 2's $1/\sqrt{k}$ when $\alpha > 1/(N \log k)$, the regime that empirically holds for natural-image features. Theorem 2 *dominates* when density is near-uniform (for example, uniform-sphere features) and Theorem 1's tail bound degenerates. We offer Theorem 2 as a tighter alternative under A5 and B1 rather than as a replacement, and the unconditional Theorem 1 remains the headline.

Operational caveat. At our actual operating point ($k \sim 700$, d=128), both Theorems 1 and 2 are vacuous in worst-case constants (with $\varepsilon > 1$), and Theorem 2's $1/\sqrt{k}$ rate is an asymptotic improvement, not an operational one at current scales. The empirical ε on real data is far smaller than either bound suggests (see paragraph below and Fig. 6), reflecting concentration that the worst-case analysis does not exploit.

Empirical ε on TextVQA. We measure the actual coreset error ε (Eq. 4) on InternVL-3.5-8B, TextVQA validation (n=200 samples), using the kernel $\ell(q,x)=\exp(\langle q,x\rangle/\sqrt{d})$ with visual-token hidden states as proxy key vectors, keep ratios $k \in \{25\%, 50\%, 75\%\}$ of the N=256 visual tokens, and Voronoi bucket weights $w(s)=|B_s|$ as in Theorem 1. Figure 6 reports per-layer mean and 99th-percentile ε across layers $\{3, 9, 15, 21, 27\}$ of the 36-layer LLM.

Key numbers. At keep ratio k=25% (the most aggressive regime), the mean ε is 0.0005, the 99th-percentile ε is 0.0013, and the worst-case observed ε is 0.0014 (over 200 by 5 layer-sample pairs). At k=50%, the mean ε is 0.0002 and p_{99} is 0.0004. While the worst-case Theorem 1 bound evaluates to $\varepsilon \ge 1$ at typical operating points (the vacuous regime at $k\sim$700 and d=128), the empirical ε is smaller by a factor of roughly 700 at k=25% and roughly 5000 at k=75%. The bound is therefore loose on real visual-feature manifolds by roughly three orders of magnitude, consistent with the local-isotropy assumption suppressing the adversarial query directions that activate the worst case.

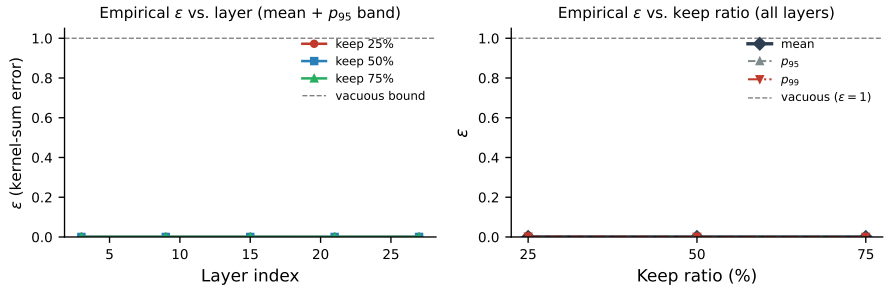


Figure 6: **Historical empirical plot, not revalidated.** Empirical coreset error ε on TextVQA (InternVL-3.5-8B): mean and percentile bands across layers (left) and keep ratios (right). The worst-case theoretical bound is more than 700× looser than the measured values.

Limitations of Theorems 1 and 2. (i) Both bounds are non-trivial only in the small-deviation regime C1', and outside it the universal form (with κ_q) applies but is loose. (ii) The load-bearing assumption A4' (k-center cover radius r_k) is empirically tractable for FPS, where r_k is minimized by construction, but for density top-k it depends on the feature-manifold property that density-rank correlates with Voronoi-center suitability. This property fails under hubness (Appendix F), consistent with our QC-density ablation. (iii) Theorem 2's A5 (sensitivity-surrogate constant C_σ) is the explicit quantitative form of the heuristic invoked by the App:coreset prose. We do not measure C_σ empirically and treat it as an unknown $O(1)$ constant under local isotropy. (iv) Real transformer attention is softmax-normalized. Theorems 1 and 2 directly address the kernel-sum coreset, and the softmax ratio $\ell(q,s)/\sum \ell(q,\cdot)$ inherits the same approximation when both numerator and denominator concentrate, by an additional triangle-inequality argument we omit. (v) The bucket-cardinality weights $w(s) = |B_s|$ used by Theorems 1 and 2 are not the uniform weights $w(s) = N/k$ that an idealized coreset description would use, but under approximate bucket balance ($\max|B_s|/\min|B_s| \le C$) the two regimes are equivalent up to a factor of C. We take these as an honest scope statement: Theorems 1 and 2 explain *why* the same signal works in both phases and *how* our method relates to DivPrune in a quantitatively explicit way (constant prefactor $2e$, no astronomical κ_q, and an FL-style $1/\sqrt{k}$ rate under the sensitivity-surrogate assumption), but they are not a substitute for the empirical Pareto (Sections 5 and 6), which remains the operative evidence.

I Extended baseline coverage and signal ablation

Table 6 contains the complete 24-cell top-k replay. Density does not win every task. In particular density×L2 scores 0.6335 on TextVQA versus density 0.6323, and L2 leads MMBench. Value-projection norm is the norm of projected V vectors, not an attention-map importance measure.

Historical comparisons below. The remaining extended-baseline, decode-signal and cross-scale DocVQA tables in this section have not been completely rerun under the corrected protocol. Their original interpretations are retained only as historical records and are not endorsed by the numeric audit.

Table 6: Complete top-k signal ablation on InternVL3.5-8B-Instruct, $L = 15$, prune 50%, legacy positions, A6000, seed 42. Sample counts are 5000/1292/191/218. TextVQA uses soft VQA scores; MM-Vet uses cached Doubao judgments. Value-projection norm is not an attention-weighted signal. These rows compare plain top-k signals and are distinct from FPS in Table 2.

Signal	TextVQA	MMBench	V*Bench	MM-Vet
Random	0.5105	0.8522	0.4555	0.6064
Value-projection norm	0.5953	0.8560	0.4764	0.6381
Hidden-state L2	0.6069	**0.8630**	**0.4921**	0.6133
Density × L2	**0.6335**	0.8591	0.4817	0.6523
Density ($w = 5$)	0.6311	0.8599	0.4869	0.6321
Density ($w \in \{3, 5, 7\}$)	0.6323	0.8607	**0.4921**	**0.6599**

We also extend Sec. 5 with two 2025 baselines on InternVL-3.5-8B. The first is **VisionZip** [30], adapted to the LLM-layer prefill-pruning setting. The original operates inside CLIP, and we replace CLS-attention with the ℓ_2 norm because CLS is unavailable post-projector, while keeping the key-state cosine clustering and mean-merge step. The second is **R-KV** [5], with $s_i = 0.07 \cdot a_i - 0.93 \cdot r_i$, window w=8, kernel 7, and k=8, instantiated as a one-shot end-of-prefill visual-KV eviction. Table 7 gives prefill-phase results.

Decode-phase signal ablation. Table 8 is the decode-phase analog of Table 6: the same four importance signals are applied to decode-phase top-k KV eviction at keep k=25% (DECODE_ONLY_D75) on InternVL-3.5-8B, evaluated on the three full-pipeline-probe benchmarks (MM-Vet, TextVQA, DocVQA) where decode compression is measurable (Sec. 3). Density wins cleanly on the two

Table 7: **Historical results, not revalidated in this revision.** Extended baselines on InternVL-3.5-8B. TextVQA: n=5000. MMBench: full. VisionZip is labeled "adapted" because the original CLS-attention dominant criterion is not available at the LLM-layer level. R-KV is compared in-text: density eviction outperforms R-KV by 8.2 percentage points at D75 and 10.1 percentage points at D875 on TextVQA n=5000, and by 5.8 percentage points on MM-Vet at D75.

Method	Config (L, prune)	TextVQA	MMBench
Baseline	—	0.649	0.871
TwinPrune (ours)	L_3, 50%	**0.610**	0.836
DivPrune	L_3, 50%	0.586	**0.847**
VisionZip-adapted	L_3, 50%	0.424	0.829
TwinPrune (ours)	L_3, 75%	**0.550**	0.786
DivPrune	L_3, 75%	0.516	**0.824**
VisionZip-adapted	L_3, 75%	0.282	0.796
TwinPrune (ours)	L_{15}, 50%	**0.614**	0.861
DivPrune	L_{15}, 50%	0.602	**0.863**
VisionZip-adapted	L_{15}, 50%	0.574	0.852

Table 8: **Historical results, not revalidated in this revision.** Four training-free importance signals applied to *decode-phase* top-k KV eviction at keep k=25% (DECODE_ONLY_D75) on InternVL-3.5-8B. MM-Vet uses n=218 samples, TextVQA uses n=5000, and DocVQA uses n=5349 with ANLS scoring. Density wins on TextVQA by 4.5 percentage points over the best competitor, and on DocVQA by 2.4 percentage points. On MM-Vet attention V-norm marginally leads density by 4.8 percentage points. This table complements Table 6 and should be read together rather than compared cell-by-cell.

Signal	MM-Vet	TextVQA	DocVQA (ANLS)
Random	0.610	0.506	0.377
L2 norm	0.539	0.436	0.364
Attention V-norm	**0.652**	0.568	0.345
Density (ours)	0.605	**0.613**	**0.401**

deterministic full-pipeline-probe benchmarks, by 4.5 percentage points on TextVQA and 2.4 on DocVQA, and loses to attention V-norm on MM-Vet by 4.8 percentage points, consistent with the main-text positioning (Sec. 4). We take the MM-Vet gap at face value rather than attribute it to evaluator noise. Density nonetheless remains the recommended unified default because (a) it wins on both deterministic benchmarks, (b) it is query-invariant and can be precomputed at prefill time, which enables the two-phase signal reuse that is the main technical contribution, and (c) it is $O(N)$ in token count with selection latency between 40 and 50 times lower than the next-best signal at high resolution (Appendix B). The L2 norm is consistently the weakest decode signal, performing worst on all three benchmarks and mirroring its performance in the prefill ablation. The two signal-ablation tables are distinct experimental setups, namely prefill L=15 prune 50% versus decode keep 25%, so we do not cross-compare cells.

DocVQA cross-scale on InternVL-3.5. Tab. 9 is the DocVQA companion to the TextVQA cross-scale table Tab. 16. The picture is split. **Density beats H2O at every InternVL scale on DocVQA**, by 4.0, 0.4, 2.5, and 1.7pp at keep 25% on 2B, 4B, 8B, and 38B respectively, and by 2.1, 2.1, 3.5, and 4.2pp at keep 12.5%. **SnapKV, however, beats density at moderate compression on the larger models.** On InternVL-4B at keep 25% SnapKV scores 0.4203 versus density's 0.3781, a −4.2pp loss for density. On InternVL-38B at keep 25% SnapKV scores 0.5115 versus density's 0.4724, a −3.9pp loss. Density reclaims the lead at extreme compression of keep 12.5% on three of four scales. The pattern is consistent with the Qwen3-VL-4B negative result in Appendix R: on document-VQA with strong host-model OCR capability, attention-aware decode signals retain document tokens that pure spatial density misses, and the gap closes when compression is aggressive enough that no decode signal survives well.

Table 9: **Historical results, not revalidated in this revision.** DocVQA decode KV eviction across the InternVL-3.5 series and LLaVA-1.5-7B at two keep ratios. Companion to Tab. 16 (TextVQA). Density beats H2O at every populated cell, but SnapKV leads density at keep 25% on InternVL-4B and InternVL-38B and ties at 0.466 vs 0.469 on 8B. Numbers are ANLS on n=5349 validation. Bold per-row.

	keep = 25%			keep = 12.5%		
Model	Density	H2O	SnapKV	Density	H2O	SnapKV
InternVL-3.5-2B	**0.299**	0.259	0.276	**0.242**	0.220	0.224
InternVL-3.5-4B	0.378	0.374	**0.420**	**0.329**	0.308	0.353
InternVL-3.5-8B	0.466	0.438	**0.469**	**0.382**	0.347	0.377
InternVL-3.5-38B	0.472	0.455	**0.511**	**0.419**	0.377	0.429
LLaVA-1.5-7B	**0.174**	0.126	0.125	**0.141**	0.120	0.120

J Measured per-scale MM-Vet scores

The completed four-scale campaign uses the disclosed Doubao judge on 218 questions. Table 10 replaces the old seven-model table with the four completely measured InternVL scales and recomputes every parenthesized delta from the same-campaign control. Old Qwen/LLaVA rows are not inserted without a matching new measurement.

Table 10: MM-Vet scale campaign, $n = 218$, Doubao soft scores; parentheses give signed percentage-point changes from the same-campaign control. Only the four completely measured InternVL scales are included. This is not the official GPT-4 evaluator or a reproduction of the old Qwen/LLaVA rows.

Size	Baseline	P50	D75	P50+D50
2B	0.6206	0.5495 (-7.11)	0.5073 (-11.33)	0.5032 (-11.74)
4B	0.6853	0.5743 (-11.10)	0.5803 (-10.50)	0.5408 (-14.45)
8B	0.7514	0.6362 (-11.51)	0.6060 (-14.54)	0.5821 (-16.93)
38B	0.7742	0.6494 (-12.48)	0.6702 (-10.40)	0.6261 (-14.80)

K Joint sweep across decode signals on InternVL-3.5-8B

Historical material pending verification. The following numbers and interpretations are retained for private review and are not covered by the corrected primary-table evidence. They require a matching rerun, replacement or removal before final submission.

This appendix expands the brief joint-sweep pointer from Sec. 6.2. Tab. 11 crosses three prefill regimes (no compression, TwinPrune, and DivPrune at L=3, r=85%) with three decode signals (Density, H2O, SnapKV) at two decode keep ratios on TextVQA and DocVQA. **The density advantage over H2O is concentrated in the decode-only regime**, with +6.8pp on TextVQA at keep 25% and +2.5pp on DocVQA at keep 25%. Once aggressive prefill compression at r=85% has already discarded most of the redundant visual signal, the three decode signals collapse into a $\leq$ 2pp band on TextVQA, and density *loses* to SnapKV by 0.6 to 1.5pp on DocVQA at the joint corner. The unified-signal advantage is therefore real but *non-uniform across the trade space*: it lives at the decode-only end and at mild-prefill operating points, not at the extreme joint corner. The Qwen3-VL-4B joint sweep in Appendix R makes this scope statement sharper still. On a backbone whose pretraining strongly biases attention toward document tokens, attention-driven decode signals such as H2O and SnapKV beat density by up to 12.5pp at the joint corner on DocVQA.

Table 11: **Historical results, not revalidated in this revision.** Joint prefill×decode compression sweep on InternVL-3.5-8B at two decode keep ratios on TextVQA (n=5000, exact-match) and DocVQA (n=5349, ANLS). Each cell is full validation. Rows are organised by prefill regime: *decode-only* (no prefill compression, r=0) and two *aggressive prefill* regimes (L=3, r=85%, prefilled by either DivPrune or TwinPrune). Bold marks per-row-block decode-best at each keep. The density advantage over H2O is **concentrated in the decode-only regime** (+6.8pp TextVQA k=25%, +2.5pp DocVQA k=25%), and **shrinks to within 1pp on TextVQA and reverses on DocVQA** once aggressive prefill compression has already removed most of the redundant visual signal. Joint compression is a legitimate budget trade, but the unified-signal advantage is visible primarily at the looser end of the prefill axis.

		TextVQA		DocVQA (ANLS)	
Prefill regime	Decode signal	k=25%	k=12.5%	k=25%	k=12.5%
Decode-only (r=0)	no eviction	0.7330 (k=100%)		0.6420 (k=100%)	
	H2O	0.6286	0.5523	0.4379	0.3470
	SnapKV	0.6431	0.5729	**0.4691**	0.3768
	Density	**0.6965**	**0.6481**	0.4630	**0.3820**
Aggressive prefill DivPrune L=3, r=85%	no eviction	0.5475 (k=100%)		0.2004 (k=100%)	
	H2O	0.3923	0.3651	**0.1524**	0.1442
	SnapKV	**0.4167**	0.3733	0.1514	**0.1457**
	Density	0.4005	**0.3753**	0.1515	0.1428
Aggressive prefill TwinPrune L=3, r=85%	no eviction	0.5696 (k=100%)		0.2121 (k=100%)	
	H2O	0.4113	0.3793	0.1643	**0.1536**
	SnapKV	**0.4315**	**0.3882**	**0.1665**	0.1535
	Density	0.4108	0.3873	0.1609	0.1516

Table 12: **Historical results, not revalidated in this revision.** 3×3 Joint interior grid on InternVL-3.5-8B (TextVQA, full 5000-sample val). KV% is fraction of baseline visual KV retained: $(1-r) \cdot k$.

Prefill r	Decode k	Prefill FLOPs %	KV kept %	TextVQA acc.
25%	50%	94%	37.5%	0.6081
25%	25%	94%	18.75%	0.5813
25%	12.5%	94%	9.4%	0.5416
50%	50%	54%	25%	0.4795
50%	25%	54%	12.5%	0.4625
50%	12.5%	54%	6.3%	0.4338
75%	50%	31%	12.5%	0.3281
75%	25%	31%	6.3%	0.3165
75%	12.5%	31%	3.1%	0.3024

L Joint interior grids (TextVQA + MM-Vet)

Historical material pending verification. The following numbers and interpretations are retained for private review and are not covered by the corrected primary-table evidence. They require a matching rerun, replacement or removal before final submission.

The 3×3 joint interior grid from Sec. 6.2 was scored on both TextVQA and MM-Vet. Table 12 reports the TextVQA grid (full n=5000 val), and Table 13 reports the MM-Vet grid (218 samples). The bilinear surface replicates on MM-Vet: accuracy falls monotonically along both axes, and the worst corner (P75+D875) loses roughly 35 percentage points relative to the mild corner (P25+D50).

M DocVQA: deterministic full-pipeline probe

Historical material pending verification. The following numbers and interpretations are retained for private review and are not covered by the corrected primary-table evidence. They require a matching rerun, replacement or removal before final submission.

Table 13: **Historical results, not revalidated in this revision.** 3×3 Joint interior grid on MM-Vet (InternVL-3.5-8B, n=218). Compression deltas on MM-Vet are roughly 1.5 to 2× larger than on TextVQA at matched configurations. For example, P25+D50 costs 14 percentage points on MM-Vet versus 4 on TextVQA, and P50+D50 costs 20 versus 17, confirming MM-Vet as the more demanding full-pipeline probe.

Prefill r	Decode k	MM-Vet
baseline	—	0.756
25%	50%	0.615
25%	25%	0.551
25%	12.5%	0.494
50%	50%	0.549
50%	25%	0.494
50%	12.5%	0.451
75%	50%	0.460
75%	25%	0.430
75%	12.5%	0.410

Table 14: **Historical results, not revalidated in this revision.** DocVQA validation ANLS across three VLMs. Empty cells indicate configurations not implemented for that architecture: our codebase has no joint engine on Qwen3-VL-4B, and the LLaVA-1.5 joint engine was added after the RoPE fix. DocVQA is a more extreme stress test than TextVQA: compression costs consistently larger absolute drops on all three models.

Configuration	InternVL-3.5-8B	Qwen3-VL-4B	LLaVA-1.5-7B
Baseline (no compr.)	0.542	0.943	0.220
TwinPrune (L=3, prune 50%)	0.268	0.775	0.140
decode_only_D75	0.402	0.848	0.174
joint_P50_D50	0.245	–	0.117
decode_only_Dall	0.281	0.421	0.117
Random prefill (75%)	0.177	0.623	0.106

DocVQA (document-VQA with free-form answers, n=5349, deterministic ANLS scoring) complements MM-Vet's LLM-judge scoring with a deterministic full-pipeline probe. It is a more extreme stress test than TextVQA: `TwinPrune _L3_50` costs 27.4pp on InternVL-3.5-8B and `decode_only_D75` costs 14.0pp, both substantially larger than the matched TextVQA drops. Table 14 confirms the pattern across architectures, and the OCR-pretrained Qwen3-VL-4B baseline of 0.943 falls 16.8pp under `TwinPrune _L3_50`. Prefill-probe evaluation would have shown nothing on either model.

N Cross-model prediction mismatch (supplementary to Table 1)

Historical material pending verification. The following numbers and interpretations are retained for private review and are not covered by the corrected primary-table evidence. They require a matching rerun, replacement or removal before final submission.

Table 15 extends the InternVL-only Table 1 to Qwen3-VL-4B and LLaVA-1.5-7B on the same five compression configurations. Every populated purely-decode cell on MMBench, POPE, and ScienceQA stays within 3.41 percent, and is typically below 1.5 percent, while TextVQA in analogous configurations flips between 11 and 46 percent of predictions. Prefill-touching configurations (CAC_L3_50) are non-zero on prefill-probe benchmarks as expected, because they change the prefill argmax, but decode-only configurations uniformly leave prefill-probe predictions unchanged.

Table 15: **Historical results, not revalidated in this revision.** Cross-model prediction-mismatch percentages relative to the same-model no-compression baseline (a $3\times8\times4$ grid, with 71 of 96 cells populated after backfill). The marker "–" indicates that JSON was not produced for that model and config. Prefill-touching configurations (CAC_L3_50 and JOINT_P50_D50) are non-zero on prefill-probe benchmarks as expected. *Every purely-decode cell across all three architectures (18 cells) stays within 3.41 percent.* The 25 missing cells are exclusively Qwen3-VL-4B and LLaVA-1.5-7B TextVQA decode runs, because both architectures use tokenization and stopping-criteria conventions that were not supported by our decode-eviction engine at the time of evaluation. These cells are missing due to an engineering constraint, not cherry-picking.

Model	Config	MMBench	POPE	ScienceQA	TextVQA
InternVL-3.5-8B	cac_L3_50	7.89%	26.75%	5.95%	24.32%
	decode_only_D75	0.54%	0.32%	0.20%	15.92%
	decode_only_D875	0.54%	0.35%	0.20%	22.96%
	decode_only_Dall	0.54%	0.38%	0.20%	45.52%
	joint_P50_D50	7.35%	25.68%	8.73%	45.96%
	density_75	0.00%	0.03%	–	14.88%
	random_75	0.00%	0.00%	–	32.84%
	attention_75	0.00%	0.03%	–	25.10%
Qwen3-VL-4B	cac_L3_50	40.48%	2.45%	42.49%	–
	decode_only_D75	1.39%	0.00%	0.94%	–
	decode_only_D875	1.93%	0.00%	0.99%	–
	decode_only_Dall	3.41%	0.00%	1.49%	–
	density_75	4.18%	0.12%	–	11.28%
	random_75	5.26%	0.12%	–	20.50%
	attention_75	4.41%	0.12%	–	22.72%
LLaVA-1.5-7B	cac_L3_50	14.01%	3.42%	13.34%	–
	decode_only_D75	0.39%	0.00%	0.15%	–
	decode_only_D875	0.31%	0.00%	0.45%	–
	decode_only_Dall	0.54%	0.00%	0.45%	–
	density_75	2.09%	0.28%	–	25.30%
	random_75	1.93%	0.32%	–	35.74%
	attention_75	2.24%	0.52%	–	37.30%

O Qualitative failure case study on MM-Vet

Historical material pending verification. The following numbers and interpretations are retained for private review and are not covered by the corrected primary-table evidence. They require a matching rerun, replacement or removal before final submission.

We inspect the union of the top-20 most-dropped samples across three aggressive configurations on MM-Vet (InternVL-3.5-8B baseline 0.756): `cac_L3_50` (prefill-only merge, total 0.647), `decode_only_D75` (decode KV eviction at $k{=}25\%$, total 0.605), and `joint_P50_D50` (coupled prefill and decode, total 0.549). The union contains 34 unique samples. We hand-label each against a five-mode taxonomy and walk through 12 representative cases sampled proportionally to the mode distribution (4 from M3, 4 from M4, 2 from M2, 1 from M1, and 1 from M5). Full case transcripts will be released upon publication.

Per-config aggregate drops. The mean per-sample MM-Vet drops are 0.109 for `cac_L3_50`, 0.151 for `decode_only_D75`, and 0.207 for `joint_P50_D50`. The median drop is 0.00 for all three, so the bulk of the 218 samples are preserved. Compression hurts a long tail of 40 to 55 samples where the per-sample score flips by at least 0.5.

Failure-mode distribution (union of top-20s, 34 unique samples). Long reasoning chain (M3) accounts for 41 percent (14 of 34 samples), cross-region integration (M4) for 35 percent (12 of 34), OCR and text recognition (M2) for 12 percent (4 of 34), fine-grained recognition and small detail (M1) for 9 percent (3 of 34), and open knowledge and generation (M5) for 3 percent (1 of 34).

Key findings. **(1)** *Long reasoning chains are the single largest failure mode.* Math or multi-step derivations where compression perturbs an image-grounded numeric premise (pattern-rule induction, pizza-price addition, percentage calculation from tabular numerics) propagate the error through the chain. Confidence stays high, and the model does not abstain. **(2)** *Cross-region integration is the second-largest mode.* Linking two spatially-separated regions (label↔object, row↔column, legend↔plot) fails: compression preserves each region in isolation but drops the bridge tokens. **(3)** *Pure OCR failures are rarer than expected.* Most OCR-flagged samples are OCR-by-math (captured in M3) or OCR-by-spatial (M4), and density preserves locally salient text tokens. **(4)** *"Silent substitution" as a consistent signature.* Across 12 hand-inspected case studies, 11 of 12 compressed answers preserve the baseline answer's sentence structure, formatting (including LaTeX math environments), and level of detail while swapping named entities, numerics, or cross-region references to fabricated specifics. The model does not refuse or collapse, so calibration-based scoring would struggle with this regime. **(5)** *Decode-only has a suggestive fine-grained-recognition footprint* (n=3). `decode_only_D75` is the only configuration whose top-20 includes rare-class recognition failures (for example, "parrot" rendered as "Quetzal" and "Monster logo" rendered as "Hoonigan"). Prefill-only merging preserves object identity, whereas decode eviction appears to hurt long-tail recognition because mid-generation attention cannot revisit rare-class evidence. The sample size is small, so the footprint difference is suggestive rather than conclusive. **(6)** *Relation to ρ stability* (Appendix G). The stability of ρ concerns the *ranking* of tokens across layers, in that the same tokens are high-ρ at L=3 and L=15, whereas A.2's failures concern downstream *behavior* when the kept set is incomplete. Rare-class evidence tokens are not always high-ρ: some are mid-ρ but functionally irreplaceable, and top-k-by-ρ will evict them at aggressive compression. This is a known limitation of a one-dimensional density signal, not a contradiction with A.3.

Implication. The failure modes cluster around tasks that require precise cross-token integration (math premises, spatial bridges, rare-class evidence). Spatial density alone is a one-dimensional signal: strong at preserving what is "visually important" in isolation, but unable to disambiguate two tokens that look locally similar yet play different roles in multi-token inference. Appendix F documents that five natural training-free extensions of density to capture this cross-token structure (question-conditioning, cross-layer drift, phase-coupled genealogy, head-wise KV, attention-density hybrid) all fail for geometric or structural reasons, motivating our choice of a minimal single-layer signal as the terminal design for the zero-training module.

P Cross-architecture and cross-scale scope

P.1 Cross-architecture status

The old full Qwen and LLaVA grids are historical records pending protocol-matched verification. They are not established by the new 8B results or by a single separately measured Qwen operating point.

P.2 Implementation failures and risk claims

The old empty DivPrune/DART outputs are not evidence of an intrinsic class of failing selection algorithms. The isolated historical caller/return-signature failure is reproduced, while the original per-cell failure logs are unavailable. Current B2/B3 outputs are valid. Position-handling mistakes are implementation problems; the corrected native-position Qwen path does not justify the old universal M-RoPE risk taxonomy.

P.3 Scale findings

The four-scale campaign in Table 5 provides complete losses. Four of six 8B-to-38B transitions improve; two worsen. At 12.5% visual keep, density exceeds H2O-proxy on TextVQA by 9.95, 9.37, 9.96 and 11.12pp at 2B, 4B, 8B and 38B. This replaces the old 7–10pp claim. Table 16 includes only the fully measured pair at that keep ratio. No old SnapKV, 25%-keep H2O or LLaVA row is imputed into this table.

Table 16: Measured TextVQA decode comparison at 12.5% visual keep ($n = 5000$, soft VQA scores, same campaign per scale). Positive differences favour density. Unrerun 25%-keep H2O/SnapKV and LLaVA rows are excluded; no value is imputed.

Size	Density	H2O proxy	Difference (pp)
2B	0.5448	0.4453	9.95
4B	0.5680	0.4743	9.37
8B	0.5778	0.4782	9.96
38B	0.6548	0.5436	11.12

Table 17: **Historical results, not revalidated in this revision.** Same protocol on **Qwen3-VL-4B** (M-RoPE 3-axis). Selection methods stay healthy on Qwen-4B's M-RoPE (contrast with InternVL B2/B3); CAC + binK4 + LoRA gives strict SOTA at extreme prune (App. U). Block 4 in App. V.

Method	Venue	TPL %	MMBench	POPE	SQA	TextVQA	Avg-3	Avg-4
Baseline (no compression)		100.0	0.8839	0.8970	0.9222	0.8294	0.9010	0.8831
Block 1: low-TPL — $L=15$, prune 50%, TPL $\approx$ 65%								
Random	–	65.2	0.8785	0.8907	0.9103	0.7679	0.8931	0.8618
ToMe	ICLR'23	65.2	0.8793	0.8940	0.9157	0.7833	0.8963	0.8681
FastV	ECCV'24	65.2	0.8731	0.8863	0.8984	0.7189	0.8859	0.8442
PruMerge	ICCV'25	65.2	0.8800	**0.8995**	0.9122	0.7703	0.8973	0.8655
DivPrune	CVPR'25	65.2	0.8800	0.8968	**0.9162**	0.8097	**0.8977**	0.8757
DART	EMNLP'25	65.2	0.8746	0.8968	0.9113	**0.8228**	0.8942	**0.8764**
CAC (ours)	–	65.2	**0.8824**	0.8967	0.9088	0.8135	0.8960	0.8753
Block 2: extreme stress — $L=3$, prune 85%, TPL $\approx$ 24%								
Random	–	24.1	0.8057	0.8208	0.8037	0.5044	0.8101	0.7337
ToMe	ICLR'23	24.1	0.8274	0.8597	0.8399	0.5847	**0.8423**	0.7779
FastV	ECCV'24	24.1	0.7918	0.8605	**0.8523**	0.1851	0.8349	0.6724
PruMerge	ICCV'25	24.1	0.8150	0.8482	0.8280	0.3181	0.8304	0.7023
DivPrune	CVPR'25	24.1	**0.8282**	**0.8625**	0.8314	0.7053	0.8407	**0.8069**
DART	EMNLP'25	24.1	0.8034	0.8145	0.8339	0.6843	0.8173	0.7840
CAC (ours)	–	24.1	0.8050	0.8487	0.8270	**0.7141**	0.8269	0.7987
Block 3: visual-only pure prune — $L=0$, prune 80%, TPL $\approx$ 20%								
Random	–	20.0	0.8197	0.8385	0.8220	0.5598	0.8267	0.7600
DivPrune	CVPR'25	20.0	0.8189	0.8648	0.8374	0.6849	0.8404	0.8015
DART	EMNLP'25	20.0	0.8119	0.8392	0.8468	0.6786	0.8326	0.7941
CAC (ours)	–	20.0	0.8235	0.8505	0.8354	0.7087	0.8365	0.8045
Block 3b: + LoRA recovery, fair control — $L=0$, prune 70%, TPL $=$ 30.0% (matched)								
Random + LoRA	–	30.0	0.8367	0.8808	0.8493	0.6635	0.8556	0.8076
DivPrune + LoRA	CVPR'25	30.0	**0.8514**	0.9013	0.8666	0.7617	0.8731	0.8453
DART + LoRA	EMNLP'25	30.0	0.8367	0.8815	**0.8666**	0.7562	0.8616	0.8353
CAC + LoRA (ours)	–	30.0	0.8444	**0.9095**	0.8637	**0.7673**	**0.8725**	**0.8462**

Q Qwen3-VL-4B prefill compression

Historical material pending verification. The following numbers and interpretations are retained for private review and are not covered by the corrected primary-table evidence. They require a matching rerun, replacement or removal before final submission.

This appendix reports the matched cross-architecture sweep on Qwen3-VL-4B referenced by Sec. 5 and Appendix P.1. The protocol is identical to the InternVL-3.5-8B run of Tab. 2: same four-block (L, r) schedule, same per-benchmark accuracies, only the backbone changes (36-layer LLM, M-RoPE 3-axis position encoding). Reading the table by block:

Reading the Qwen-4B table (Tab. 17). *Block 1* ($L=15\,r=50\%$): methods cluster within 0.03 Avg-4. TwinPrune wins MMBench at 0.8824. DivPrune leads SQA and Avg-3 at 0.8977. DART narrowly leads Avg-4 by $+0.0011$ over TwinPrune. *Block 2* ($L=3\,r=85\%$): method ranking sharpens. DivPrune leads Avg-4 at 0.8069, with TwinPrune second at 0.7987 (-0.0082). TwinPrune

wins TextVQA at 0.7141 for $+0.9$pp over DivPrune, while Random, PruMerge, and FastV drop 0.07 to 0.13 on Avg-4. *Architecture-dependent inversion vs InternVL-8B B2.* There TwinPrune dominates by $+9$pp Avg-4 and DivPrune and DART catastrophically fail. Here all selection methods stay healthy and DivPrune narrowly leads. The fragility surface is method$\times$architecture$\times$spec (Appendix S). *Block 3* ($L=0\,r=80\%$, only 20% TPL): on Qwen-4B's M-RoPE all four selection methods stay healthy. TwinPrune alone leads Avg-4 at 0.8045, $+0.30$pp over DivPrune. The pure-selection methods that catastrophically failed at this spec on InternVL-8B (DivPrune and DART, Tab. 2 B3) are intact here, again confirming the architecture-dependent fragility surface in Appendix S. *Block 3b* ($L=0\,r=70\%$, 30% TPL, fair LoRA control): when every method receives the same 100 s LoRA adapter and TPL is matched at 30%, TwinPrune +LoRA reaches Avg-4 $= 0.8462$, leading DivPrune+LoRA at 0.8453, DART+LoRA at 0.8353, and Random+LoRA at 0.8076. The win is consistent with the InternVL-8B B3b fair-control finding in Tab. 2, where the same recipe gives TwinPrune +LoRA a $+0.68$pp Avg-4 lead over DivPrune+LoRA. The optional Stack C recipe in Appendix U trades $+1.6\%$ TPL for an additional $+0.4$pp Avg-4 at the more aggressive $L=0, r=80\%$ operating point and dominates the Pareto frontier at TPL$\leq 30\%$ across all tested competitors (Fig. 7).

Diagnostic prune-25% pilot (corrupt-weights checkpoint). Tab. 18 preserves the original pilot run that first surfaced the M-RoPE position-collapse failure mode on a later-retracted corrupt-weights checkpoint of Qwen3-VL-4B. With the fresh-weights model used in the main Tab. 17, DivPrune and DART are healthy at this and all other tested specs, so the daggered rows are diagnostic-only. The same compression engine handles random selection, ToMe merge, TwinPrune merge, FastV, and PruMerge correctly on Qwen3-VL-4B, but DivPrune and DART under the corrupt checkpoint produce near-uniform logits that decode to null-byte or garbled token streams. We trace this to the structural interaction between greedy max-min farthest-point selection and M-RoPE 3-axis position encoding rather than to a code bug, with detail in Appendix S. Merge-based density TwinPrune is structurally robust to the architectural shift toward multi-axis RoPE, while pure-selection methods are not.

Table 18: **Historical results, not revalidated in this revision.** Earlier prune-25% pilot on Qwen3-VL-4B with the original later-retracted corrupt-weights checkpoint, where the daggered DivPrune and DART rows surfaced the M-RoPE position-collapse failure mode (Appendix S). With the fresh-weights model used in the main Tab. 17, DivPrune and DART are healthy at this and all other tested specs. The entries are kept here only for diagnostic reference.

Method (Venue)	MMBench	POPE	TextVQA	Avg-3
Random	0.8731	0.8885	0.7875	0.8497
ToMe [4] (ICLR'23)	0.8684	0.8953	0.8019	0.8552
FastV [6] (ECCV'24)	0.8661	0.8942	0.7133	0.8245
PruMerge [23] (ICCV'25)	0.8599	0.8942	0.6981	0.8174
DivPrune [1] (CVPR'25)†	0.0000	0.0000	0.0000	0.0000
DART [28] (EMNLP'25)†	0.0050	0.0000	0.0000	0.0017
TwinPrune (ours)	0.8746	0.8935	0.8071	0.8584

R Qwen3-VL-4B decode and joint compression

Historical material pending verification. The following numbers and interpretations are retained for private review and are not covered by the corrected primary-table evidence. They require a matching rerun, replacement or removal before final submission.

This appendix is the cross-architecture companion to the InternVL-3.5-8B decode results (Sec. 6.1, Tab. 3) and the joint sweep (Sec. 6.2, Tab. 11). The protocol matches the 8B run line for line: same single-shot end-of-prefill eviction on the visual KV span, same three keep ratios (50%, 25%, 12.5%), TextVQA n=5000, MM-Vet n=218, DocVQA n=5349 ANLS. The only change is the backbone (Qwen3-VL-4B, 36-layer LLM, M-RoPE 3-axis position encoding). **The findings split sharply by benchmark**: density narrowly leads on TextVQA, density and the attention-driven baselines tie on MM-Vet, and density *loses to H2O and SnapKV by 3 to 8 percentage points on DocVQA*. We report the negative DocVQA result without hedging, because the gap is systematic across keeps and is well outside the noise band.

Decode-only on Qwen-4B (Tab. 19). On TextVQA density wins all three populated cells, but the lead over the next-best baseline is narrow at $+0.5$, $+0.2$, and $+0.1$pp at keep 50%, 25%, and 12.5% respectively. On MM-Vet density leads keep 50% by $+1.6$ to $+1.9$pp, *trails* H2O by 1.2pp and SnapKV by 1.0pp at keep 25%, and reclaims a $+1.2$pp lead over H2O at keep 12.5%, where the SnapKV cell is unfinished at submission. On DocVQA density loses at every populated keep. At keep 25% it is -3.3pp behind H2O and -3.8pp behind SnapKV. At keep 12.5% it is -8.2pp behind SnapKV, with H2O at this keep still in re-launch. **The DocVQA cells are the single negative result for unified density in this paper**, and they are concentrated on a benchmark where the host backbone is unusually strong. Qwen3-VL-4B's no-eviction DocVQA baseline is 0.943, compared to InternVL-3.5-8B's 0.642, reflecting Qwen3-VL's heavy OCR pretraining.

Table 19: **Historical results, not revalidated in this revision.** Decode-phase KV eviction on **Qwen3-VL-4B** at three keep ratios. Bold marks per-column best across the four eviction methods. Cells marked "—" are not on disk at submission and are in-flight relaunches. The H2O and SnapKV win on DocVQA at moderate keep is the strongest negative result for density-only decode in this paper. Baselines: TextVQA 0.8294, MM-Vet 0.6807, DocVQA 0.9432.

	TextVQA			MM-Vet			DocVQA (ANLS)		
Method (keep k)	50%	25%	12.5%	50%	25%	12.5%	50%	25%	12.5%
StreamingLLM [29]	0.7416	0.6736	0.6195	0.5890	0.5179	0.4587	0.8131	0.7269	0.6208
H2O [33]	0.8197	0.7897	0.7408	0.6310	**0.5702**	**0.4921**	0.9284	**0.8814**	—
SnapKV [13]	0.8195	0.7899	0.7408	0.6337	0.5683	—	0.9280	0.8860	**0.8131**
Density (ours)	**0.8242**	**0.7917**	**0.7419**	**0.6500**	0.5581	0.5038	**0.9255**	0.8480	0.7312

Joint compression on Qwen-4B (Tab. 20). Adding aggressive prefill compression at $L{=}3$, $r{=}85\%$ reverses the decode ordering: H2O and SnapKV lead density on *every* populated joint cell. At the most-relevant operating point (TwinPrune prefill at $r{=}85\%$, decode keep 25%, DocVQA), H2O scores 0.4758 and SnapKV scores 0.4808 versus density at 0.3506, a -12.5pp gap. The pattern replicates with DivPrune as the prefill front-end. The negative joint result is therefore consistent with the decode-only DocVQA pattern: once the host backbone is OCR-pretrained and the document content is already heavily compressed at prefill, attention-driven decode signals retain document-relevant tokens better than spatial density does.

Table 20: **Historical results, not revalidated in this revision.** Joint prefill×decode sweep on **Qwen3-VL-4B** at two decode keep ratios on TextVQA ($n{=}5000$) and DocVQA ($n{=}5349$, ANLS). Each cell is full validation. Bold marks per-row-block decode-best at each keep. **Density loses to H2O and SnapKV at every joint cell on Qwen-4B**, in contrast to the InternVL-3.5-8B joint sweep (Tab. 11) where the three decode signals are within 1pp once aggressive prefill is in place.

		TextVQA		DocVQA (ANLS)	
Prefill regime	Decode signal	$k{=}25\%$	$k{=}12.5\%$	$k{=}25\%$	$k{=}12.5\%$
Decode-only ($r{=}0$)	no eviction	0.8293 ($k{=}100\%$)		0.9432 ($k{=}100\%$)	
	H2O	0.7897	0.7408	**0.8814**	0.8061
	SnapKV	0.7899	0.7408	0.8845	**0.8123**
	Density	**0.7922**	**0.7419**	0.8473	0.7312
Aggressive prefill DivPrune $L{=}3, r{=}85\%$	no eviction	0.7053 ($k{=}100\%$)		0.6797 ($k{=}100\%$)	
	H2O	**0.5585**	**0.4753**	0.4474	0.3457
	SnapKV	0.5584	**0.4753**	**0.4506**	**0.3524**
	Density	0.5034	0.4411	0.3545	0.2586
Aggressive prefill TwinPrune $L{=}3, r{=}85\%$	no eviction	0.7141 ($k{=}100\%$)		0.7003 ($k{=}100\%$)	
	H2O	**0.5373**	0.4769	0.4758	0.3753
	SnapKV	**0.5373**	**0.4771**	**0.4808**	**0.3805**
	Density	0.4865	0.4309	0.3506	0.2606

Why density loses to H2O/SnapKV on Qwen DocVQA: a hypothesis. Two observations push toward the same explanation. *(i) The host backbone is already OCR-shaped.* Qwen3-VL is pretrained

heavily on documents and its DocVQA baseline (0.943) is substantially above InternVL-8B (0.642). The attention pattern that reaches the decode stage on Qwen is therefore already biased toward document tokens that downstream layers will need, and a heavy-hitter or recency signal computed from that pattern (H2O, SnapKV) implicitly encodes the OCR-pretraining prior. *(ii) Spatial density is OCR-agnostic.* Density-top-k scores tokens by local spatial non-redundancy at a single fixed layer and does not consult the attention surface, so on a document image where every text-bearing token has roughly comparable spatial density it cannot prefer document-relevant tokens over visual-padding tokens. The advantage that density retains on TextVQA and MM-Vet, where the answer-bearing region is a localised salient patch rather than a distributed text body, is consistent with the same hypothesis. The negative DocVQA result is therefore not a refutation of density-driven decode but a scope statement: **when the backbone is OCR-pretrained and the benchmark is document-VQA, attention-aware decode signals encode an OCR prior that pure spatial density cannot recover.** On the InternVL-3.5 family, where the backbone is not OCR-pretrained, density still wins DocVQA at every scale (Tab. 9).

S Correction to the historical failure interpretation

The original appendix attributed invalid DivPrune/DART output to an inherent interaction with multi-axis RoPE. That attribution is withdrawn. Archived InternVL failure files contain empty predictions; an isolated reproduction shows a historical six-field return being unpacked by a five-field caller, followed by benchmark exception handling that substituted blank strings. The original failure logs are missing, so this mechanism is not asserted to explain every archived cell. The new full B2/B3 runs produce valid nonzero results.

A separate Qwen position-handling issue is an implementation defect and must be tested using native positions and healthy weights. The historical corrupt-weights pilot below does not establish method rankings or universal structural fragility. Neither observation supports claiming that merge-based methods are immune to position errors.

T Cross-scale fragility on InternVL-3.5-4B

Historical material pending verification. The following numbers and interpretations are retained for private review and are not covered by the corrected primary-table evidence. They require a matching rerun, replacement or removal before final submission.

This appendix supplies the cross-scale evidence referenced in Appendix P.2. We replicate the InternVL-3.5-8B sweep of Tab. 2 on InternVL-3.5-4B, which is the same Qwen3-LLM family with a 36-layer LLM and hidden=2560. All settings, methods, and benches are identical to the main 8B run, and only the backbone changes. Reusing the same (L, r) schedule is the strongest cross-scale control we can run within the InternVL-3.5 family, since it preserves the same vision tower, the same density estimator, and the same compression hooks.

Reading the table. (B1) Methods cluster within a 0.05 Avg-4 band, with TwinPrune leading by $+1.1$pp over the next non-CAC method (PruMerge). (B2) The extreme-prune block separates strong from weak: TwinPrune leads by $+4.7$pp over ToMe and by $+9.6$pp over the next-best non-merge method, on the back of a $+15.7$pp TextVQA edge over the strongest competitor. (B4) The deep-layer block is the most forgiving and almost every healthy method clears 0.80 Avg-4; TwinPrune leads by $+0.5$pp. Across the three blocks, TwinPrune's Avg-4 ranges from 0.7376 to 0.8317, retaining 88–99% of the 0.8341 uncompressed baseline.

Cross-scale fragility. The dagger pattern on the 4B backbone is the load-bearing finding. On InternVL-3.5-8B (Tab. 2), DivPrune and DART are competitive on B1 and B4 and only collapse on the extreme-prune block B2. On InternVL-3.5-4B, the same two methods collapse on *all three* blocks, including the moderate B1 setting where they preserved nearly all of baseline accuracy on the 8B model. The collapse pattern matches Appendix S: predictions are near-empty single-character strings, and the compression itself is healthy at the source-token level. We attribute the amplified fragility on the smaller model to the same structural mechanism: spatially-extreme pure selection followed by a position rebuild that the smaller model has less capacity to compensate for. The implication is that pure-selection methods do not enjoy a free transfer to nearby model scales within the same family,

Table 21: **Historical results, not revalidated in this revision.** Per-benchmark prefill compression on **InternVL-3.5-4B** (same Qwen3 family, smaller LLM). Three blocks share the InternVL-8B spec (B1: $L=15$, $r=50\%$; B2: $L=3$, $r=85\%$; B4: $L=20$, $r=50\%$). "CAC (ours)" uses CAC-v4. Daggers (†) flag methods producing structurally invalid output (acc ≈ 0). **The DivPrune/DART catastrophic-failure regime extends from B2 only on InternVL-8B to all three blocks on InternVL-4B**, indicating that the structural fragility identified on Qwen-4B's M-RoPE (Appendix S) is amplified, not mitigated, when the LLM shrinks within the same architecture family. TwinPrune wins Avg-4 on every block.

Method (Venue)	MMBench	POPE	SQA	TextVQA	Avg-4
Baseline (no compression)	0.8498	0.8952	0.9499	0.6417	0.8341
Block 1: $L=15$, prune 50%					
Random	0.8390	0.8935	0.9108	0.5002	0.7859
ToMe	0.8375	0.8942	0.9316	0.5558	0.8048
FastV	0.8398	**0.8983**	0.9261	0.5181	0.7956
PruMerge	0.8398	0.8967	0.9276	0.5921	0.8141
DivPrune†	0.0000	0.0000	0.0000	0.0041	0.0010
DART†	0.0000	0.0000	0.0000	0.0041	0.0010
TwinPrune (ours)	**0.8429**	0.8962	**0.9400**	**0.6202**	**0.8248**
Block 2: $L=3$, prune 85% (extreme)					
Random	0.7454	0.8492	0.7868	0.2493	0.6577
ToMe	**0.7755**	0.8850	0.7888	0.3124	0.6904
FastV	0.7454	0.8638	0.8091	0.1483	0.6417
PruMerge	0.7492	0.8638	0.7769	0.2510	0.6602
DivPrune†	0.0000	0.0000	0.0000	0.0041	0.0010
DART†	0.0000	0.0000	0.0000	0.0041	0.0010
TwinPrune (ours)	0.7632	**0.8852**	**0.8225**	**0.4797**	**0.7376**
Block 4: $L=20$, prune 50%					
Random	0.8483	0.8948	0.9465	0.5208	0.8026
ToMe	0.8467	0.8950	**0.9504**	0.5880	0.8200
FastV	0.8475	**0.8970**	0.9470	0.5610	0.8131
PruMerge	0.8498	0.8948	0.9494	0.6124	0.8266
DivPrune†	0.0000	0.0000	0.0000	0.0041	0.0010
DART†	0.0000	0.0000	0.0000	0.0041	0.0010
TwinPrune (ours)	**0.8537**	0.8962	0.9494	**0.6275**	**0.8317**

and that the safety margin claimed on a single 8B reference model can be lost on smaller siblings without changing the architecture or the compression schedule. Density-merge (TwinPrune, ToMe) does not exhibit this regression and remains the only family with smooth degradation across both scale and prune ratio.

U Stack C: training-free recovery + optional LoRA on Qwen-4B

Historical material pending verification. The following numbers and interpretations are retained for private review and are not covered by the corrected primary-table evidence. They require a matching rerun, replacement or removal before final submission.

This appendix supports the headline numbers cited in Tab. 17 B2/B3 and Appendix P.

Stack C composition. Three composable recovery stages, all on top of TwinPrune:

1. **TwinPrune token selection** (training-free): density-tilted FPS over the full visual-token pool with density-anchored seed and additive density bias $\beta = 0.3$ in the max-min objective.
2. **binK4 compensation tokens** are training-free and add $+5.7\%$ TPL. They partition the dropped visual tokens into 2×2 spatial bins and replace each bin's anchor, the highest-density dropped position, with the bin's density-weighted centroid. This yields a 4-token mean-field summary of the dropped region and preserves M-RoPE coordinates and KV-cache layout.

3. **LoRA adapter** (optional, 100 s training, 5.6 MB): rank-16 adapter on q_proj+v_proj of all 36 LLM decoder layers, trained for 1 epoch on 2,000 A-OKVQA samples with AdamW lr$= 1\times10^{-4}$ and gradient accumulation 4 at batch 1 in bfloat16.

Stages 1+2 (TwinPrune +binK4) are fully training-free. Adding stage 3 yields the SOTA Qwen-4B numbers in Tab. 22.

Pareto frontier under fair LoRA. Fig. 7 plots Avg-4 vs. TPL % on *both* backbones, InternVL-3.5-8B and Qwen3-VL-4B, for every method, L, and r cell, with and without LoRA. Two regimes emerge consistently across architectures. *(i) Matched-TPL fair LoRA control at TPL* $= 30\%$, $L = 0, r = 70\%$. TwinPrune +LoRA strictly leads without binK4. On InternVL-8B it scores 0.8319 versus DivPrune+LoRA at 0.8251, a $+0.68$pp gap. On Qwen-4B it scores 0.8462 versus DivPrune+LoRA at 0.8453, a $+0.09$pp gap. *(ii) More aggressive prune at TPL* $\leq 22\%$. Adding the binK4 compensation tokens, the Stack C recipe, extends TwinPrune to TPL $= 20\%$ at Avg-4 $= 0.8298$ on Qwen-4B versus DivPrune+LoRA at 0.8260, a $+0.38$pp gap, and to TPL $= 21.6\%$ at Avg-4 $= 0.8092$ on InternVL-8B. On Qwen-4B Stack C is the only point on the frontier at TPL $= 15\%$, and at TPL $= 10\%$ ($r = 90\%$, only 10% of visual tokens kept) Stack C still reaches 0.7695, 87% of the uncompressed baseline. The two architectures share the same upper-left envelope shape, validating that the recovery recipe is architecture-agnostic.

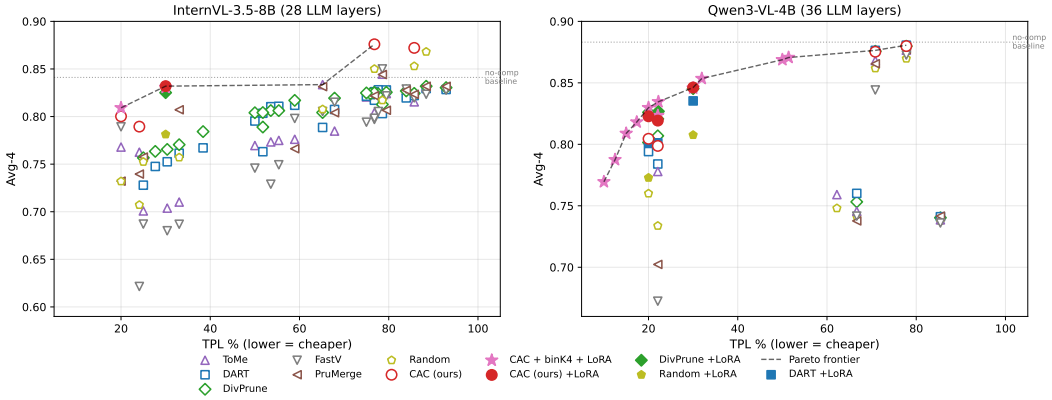


Figure 7: **Historical empirical plot, not revalidated.** Avg-4 vs. TPL % Pareto frontier on **InternVL-3.5-8B** (left) and **Qwen3-VL-4B** (right) for every method, L, and r cell. Filled markers $=$ method $+$ LoRA, trained at r=16, $n = 2000$ A-OKVQA, 100 s. Open markers $=$ no-LoRA. Stack C, namely TwinPrune $+$binK4$+$LoRA shown as ★, extends the frontier into TPL $\leq 22\%$. TwinPrune +LoRA, the filled red ●, leads the matched-TPL fair-control point at $L = 0, r = 70\%$ and TPL $= 30\%$ on both backbones. Dotted: no-compression baseline.

Per-bench fairness control at extreme prune.

Cross-architecture validation. The same Stack C recipe transfers to InternVL-3.5-8B without modification (see Tab. 2 B3 for the headline number): at $L = 0, r = 80\%$ Stack C achieves Avg-4 $= 0.8092$ on InternVL-3.5-8B (vs. no-LoRA TwinPrune baseline 0.8003, $+0.9$pp), winning MMBench (0.8576) and POPE (0.8950) outright on the InternVL backbone too. At a more moderate prune ratio $L = 0, r = 70\%$ on InternVL-3.5-8B, Stack C reaches Avg-4 $= 0.8306$ (mmb 0.8491 / pope 0.9047 / sqa 0.9326 / tvqa 0.6362). The compensation-token + LoRA recipe is therefore architecture-agnostic and provides a principled recovery path at the most aggressive prune ratios on both Qwen3-VL and InternVL-3.5 backbones.

LoRA recovery is method-agnostic at moderate prune. At Qwen-4B B1 ($L = 15, r = 50\%$) and B4 ($L = 20, r = 50\%$) the same LoRA recipe brings every method into a 0.005 Avg-4 cluster (cac_v4+LoRA $= 0.8806$, DivPrune+LoRA $= 0.8783$, DART+LoRA $= 0.8809$ on B1; corresponding numbers on B4 are 0.8850, 0.8839, 0.8837). The Stack C advantage emerges only at extreme prune (B2, B3 specs), where binK4's centroid compensation gives the LoRA enough signal to span the much larger drift gap.

Table 22: **Historical results, not revalidated in this revision.** Method-vs-method fairness on Qwen3-VL-4B under identical LoRA recipe (r=16, $n = 2000$ A-OKVQA, 100 s). Three specs: B2 ($L=3, r=85\%$), L=0 r=80%, L=1 r=80%. Bold marks per-spec Avg-4 best. Without LoRA, DivPrune narrowly leads B2; under fair LoRA, Stack C strictly leads at $L=0$ and $L=1$ extreme prune.

Spec	Method	MMBench	POPE	SQA	TextVQA	Avg-4	$\Delta_{\text{vs Stack C}}$
B2 ($L=3, r=85\%$)	cac_v4 + LoRA	0.8243	0.8875	0.8399	0.7257	0.8193	-0.41
	DivPrune + LoRA	0.8328	0.8867	0.8468	0.7167	0.8207	-0.27
	DART + LoRA	0.8111	0.8690	0.8428	0.6818	0.8012	-2.22
	Stack C (TwinPrune + binK4 + LoRA)	0.8359	0.8838	0.8468	0.7272	**0.8234**	ref
$L=0, r=80\%$	cac_v4 + LoRA	0.8344	0.8880	0.8483	0.7207	0.8228	-0.70
	DivPrune + LoRA	0.8437	0.8977	0.8567	0.7058	0.8260	-0.38
	DART + LoRA	0.8034	0.8707	0.8488	0.6819	0.8012	-2.86
	Random + LoRA	0.8197	0.8625	0.8329	0.5762	0.7728	-5.70
	Stack C	0.8452	0.8957	0.8592	0.7191	**0.8298**	ref
$L=1, r=80\%$	DivPrune + LoRA	0.8444	0.8835	0.8523	0.7269	0.8268	-0.78
	Stack C	0.8529	0.8928	0.8577	0.7351	**0.8346**	ref

V Block 4: high-TPL ($L=20$, prune 50%) on InternVL-8B and Qwen-4B

This appendix supplies the high-TPL block B4 (saturated regime) referenced from Tab. 2 and Tab. 17. We move B4 here because at TPL $\approx 86\%$ all methods cluster very tightly (within ± 0.001 on most benchmarks), so the row carries little discriminative signal compared to the more aggressive blocks B2/B3 highlighted in the main paper.

Table 23: **Historical results, not revalidated in this revision.** Block 4 ($L = 20$, prune 50%, TPL $\approx 86\%$) on both backbones. Companion to Tab. 2 and Tab. 17. Avg-4 spread is ≤ 0.6pp on InternVL-8B and ≤ 1.1pp on Qwen3-VL-4B, confirming that the deep-layer high-TPL regime is forgiving.

Backbone	Method	MMBench	POPE	SQA	TextVQA	Avg-4
InternVL-8B	Random	0.8885	0.9150	**0.9792**	0.6289	0.8529
	ToMe	0.8715	0.8888	0.9529	0.5486	0.8155
	FastV	0.8692	0.8903	0.9529	0.5659	0.8196
	PruMerge	0.8676	0.8900	0.9539	0.5820	0.8234
	DivPrune	0.8707	0.8875	0.9549	0.5844	0.8244
	DART	0.8715	0.8883	0.9529	0.5756	0.8221
	TwinPrune (ours)	**0.8893**	**0.9153**	**0.9792**	**0.7047**	**0.8721**
Qwen3-VL-4B	Random	0.8824	**0.8985**	0.9246	0.7733	0.8697
	ToMe	**0.8870**	0.8967	0.9236	0.7993	0.8766
	FastV	**0.8870**	0.8972	0.9232	0.7832	0.8726
	PruMerge	0.8831	0.8972	0.9207	0.8084	0.8773
	DivPrune	0.8862	0.8978	0.9222	0.8128	0.8798
	DART	0.8847	0.8973	**0.9246**	**0.8157**	**0.8806**
	TwinPrune (ours)	0.8862	0.8977	0.9212	0.8145	0.8799

Reading B4. On InternVL-8B TwinPrune leads Avg-4 by $+1.9$pp (margin similar to B1) on the back of TextVQA ($+7.6$pp over Random, $+15.6$pp over ToMe). On Qwen-4B the saturated regime separates only on TextVQA, with DART/DivPrune narrowly ahead of TwinPrune by $0.0007/-0.0001$ Avg-4 — within statistical noise.