TOKENGRAFT: DIAGNOSING AND IMPROVING VISUAL TOKEN COMPRESSION VIA INDEPENDENT RECOVERY

Anonymous authors
Paper under double-blind review

ABSTRACT

Visual token compression reduces the visual sequence processed by vision–language models, but performance–budget curves leave one question unresolved: *does a compressor use the available token slots effectively?* At the same final token count, we compare two allocations of the available budget. One uses only compressor-produced tokens (native expansion), while the other combines fewer compressed tokens with selected pre-compression visual tokens (original-token recovery). Across multiple compressors, even simple recovery rules reveal a *recoverable gap*: this hybrid representation can outperform native expansion. We propose TokenGraft, a training-free, text-query-independent method that restores visually important original tokens from locations under-covered by the compressed representation. On Qwen2.5-VL, TokenGraft improves all 27 main grounding comparisons across three compressors, three retention rates, and RefCOCO, RefCOCO+, and RefCOCOg, with gains up to 33.57 percentage points. Fixed-total-budget sweeps further show how reallocating slots from compression to recovery improves accuracy. These gains extend to InternVL2.5, LLaVA-OneVision, and CV-Bench, with magnitudes that depend on the backbone, compressor, and task. Under our paired runtime protocol, recovery adds less than 1% full-request time across all three compressors. These results establish token allocation, alongside budget size, as a practical design choice for visual token compression.

1 INTRODUCTION

Vision–language models (VLMs) process hundreds or thousands of visual tokens before producing a response. *Visual token compression* includes pruning, merging, and aggregation operations that shorten this sequence before later language-model layers (Chen et al., 2024a; Yang et al., 2025b; Zhang et al., 2025; Shang et al., 2025; Xing et al., 2025). Fewer tokens reduce downstream computation, but aggressive compression can remove task-relevant information. Referring-expression grounding makes this loss easy to observe: the model needs both object semantics and spatial detail to localize the described region.

Compression methods are usually evaluated along a performance–budget curve, which measures accuracy as the compressor retains more tokens. Recent work also makes budgets multi-granular or adaptive across samples and layers (Cai et al., 2025; Ye et al., 2025b;a). Such a curve mixes two questions: how many token slots are available, and what representation occupies them. When the budget grows from K to $K+A$, the same pruning, merging, or aggregation mechanism constructs the larger representation. Increasing its size does not separately test whether different visual information could make better use of those slots. We ask whether allocating all available slots to that mechanism is the most effective choice.

We examine this question with a matched-budget diagnostic. *Native expansion* constructs all $K + A$ tokens through the compressor. *Original-token recovery* keeps its K-token representation and fills the remaining A slots with selected pre-compression visual tokens. Both constructions use the same final token budget, but allocate it differently between compressed and original information. Across several compressors, even simple recovery rules can match or beat native expansion. The

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

comparison therefore exposes information that remains useful to the downstream model beyond the compressor's native output. We call the accuracy advantage of the hybrid construction a *recoverable gap*.

This diagnosis motivates TokenGraft. A useful recovered token should carry visually important content and cover locations underrepresented by the compressed sequence. TokenGraft combines an attention-derived visual-importance signal with spatial complementarity in two-dimensional image coordinates. It greedily restores original tokens, preserves their features and positional information, and interleaves them with the compressed sequence. The method is training-free and independent of the text query.

We first evaluate a fixed recovery allocation across different token budgets, then hold each total budget fixed and vary the allocation between compressed and recovered tokens. Extending these sweeps to two additional VLM backbones and CV-Bench tests the same allocation question beyond the primary Qwen grounding setting. Our contributions are:

1. We identify a *recoverable gap*: at the same visual-token budget, allocating some slots to recovered original tokens can outperform native expansion.
2. We introduce TokenGraft, an image-only recovery method that combines visual importance with spatial complementarity relative to compressor-specific references.
3. We demonstrate the allocation effect across compressors and budget levels, extend fixed-budget sweeps across backbones and tasks, and measure less than 1% incremental full-request time under a paired runtime protocol.

2 Related Work

Visual token reduction for efficient VLM inference. Visual-token reduction shortens the image-derived sequence processed by a VLM, reducing subsequent multimodal and language-model computation. Existing methods differ both in what is removed and where reduction occurs. FastV and SparseVLM prune tokens using attention-derived relevance (Chen et al., 2024a; Zhang et al., 2025); VisionZip and LLaVA-PruMerge combine representative-token selection with aggregation or merging (Yang et al., 2025b; Shang et al., 2025). ViCo progressively reduces visual tokens as redundancy increases through the language-model depth (Xing et al., 2025), while Dynamic-LLaVA sparsifies vision–language context across prefill and decoding (Huang et al., 2025). DUET-VLM further coordinates vision-stage compression with layer-wise text-guided token dropping (Singh et al., 2026). These methods are usually compared across retained-token budgets. We instead ask how to allocate those slots between compression and recovery.

Adaptive budgets and multi-granularity representations. A related line of work makes the token budget itself more flexible. Matryoshka Multimodal Models learns nested coarse-to-fine visual-token sets and exposes controllable test-time granularity (Cai et al., 2025). ATP-LLaVA predicts layer- and instance-specific pruning ratios (Ye et al., 2025b), while FitPrune derives a pruning recipe for a predefined compute budget from attention statistics (Ye et al., 2025a). Task-related token compression uses instruction-conditioned importance to remove task-irrelevant visual tokens at the LLM input (Lei et al., 2026). These methods optimize the amount, location, or granularity of reduction. Our matched-budget setting holds the final token count fixed and compares which visual information occupies the available slots.

Spatial structure, coverage, and representative selection. Spatial structure matters for localization because the model must preserve where visual content occurs. FEATHER studies spatially imbalanced pruning and combines uniform sampling with multistage selection (Endo et al., 2025); PPE preserves positional components during merging (Huang et al., 2026a); and Nuwa explicitly protects spatial structure before a later text-guided pruning stage (Huang et al., 2026b). Representative selection also considers diversity and coverage. DivPrune maximizes diversity among selected tokens (Alvar et al., 2025), and TopV includes relative spatial distance in an optimization-based pruning objective (Yang et al., 2025a). SCOPE combines saliency and coverage (Deng et al., 2025), SCoRe formulates salience–coverage selection as a unified representativeness objective (Xu et al., 2026), and CoIn jointly models informativeness and global feature-space coverage (Du et al.,

2026). After compression, TokenGraft starts from the compressor's reference locations in image coordinates and recovers visually important original tokens that are spatially complementary to the compressed representation.

Retaining information outside the compressed sequence. Several methods preserve information that would otherwise disappear during reduction. SparseVLM recycles pruned information into compact tokens (Zhang et al., 2025); LLaVA-PruMerge merges contextual information around selected representatives (Shang et al., 2025); VFlowOpt aggregates pruned tokens into recycled tokens (Yang et al., 2025c); and Hi-Lo Prune transfers information from pruning candidates before removing them (Sun et al., 2026). Recoverable Compression restores visual tokens with a training-free, text-guided mechanism (Chen et al., 2025). TokenGraft connects this recovery perspective to budget allocation: it compares original-token recovery with native expansion at the same final token count, then applies one image-only recovery rule across heterogeneous compressor interfaces.

3 Diagnosing Budget Allocation in Visual Token Compression

3.1 Matched-budget diagnosis

Let X be the original visual sequence with N tokens, and let $\mathcal{C}_K$ produce $K = \lceil rN \rceil$ compressed tokens. *Native expansion* runs the compressor at budget $K + A$. *Original-token recovery* instead keeps the K-token compressed representation and uses A slots for selected pre-compression visual tokens. Figure 1 illustrates these constructions at the same final token count.

Formally,

$$Z_{\text{exp}} = \mathcal{C}_{K+A}(X), \qquad Z_R = \text{Insert}(\mathcal{C}_K(X), X_{G_A}), \quad |G_A| = A, \tag{1}$$

where G_A contains the recovered original-token indices. Both paths end with exactly $K + A$ visual tokens and use the same model, samples, generation, and scoring procedure. We measure

$$\Delta_R = \text{Acc}(Z_R) - \text{Acc}(Z_{\text{exp}}). \tag{2}$$

Here, $\text{Acc}(Z)$ denotes the downstream model's accuracy using visual representation Z. A positive Δ_R identifies a recoverable gap: at the same final token count, the hybrid construction Z_R is more accurate than native expansion Z_{exp}.

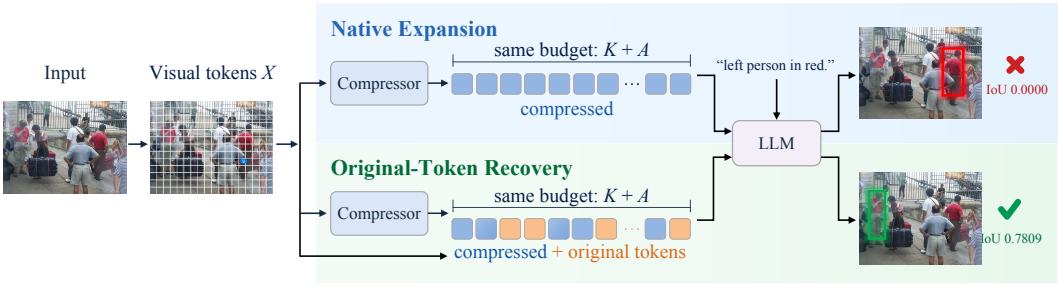


Figure 1: Matched-budget diagnosis. Native expansion uses $K + A$ compressed tokens, whereas recovery uses K compressed and A original tokens. The right-hand panels show a qualitative grounding example from the evaluation set under matched token budgets.

Nuwa has two compression stages. Recovery inserts original tokens after stage 1 and protects them through stage 2. We match both its intermediate $S + A$ and final $K + A$ counts to native expansion. Appendix A gives the interface details.

3.2 Recovery with simple selection rules

We first probe the recovery path with three simple selectors on Qwen2.5-VL / RefCOCO at $r = 25\%$ and $A = 16$: random recovery, visual-importance-only recovery, and spatial-complementarity-only recovery. The recovery arms share the base representation, candidates, insertion rule, and downstream model, so only selection changes.

Table 1: Recovery controls on Qwen2.5-VL / RefCOCO ($r = 25\%$, $A = 16$; Acc@0.5, %). Random recovery reports the three-seed mean $\pm$ population standard deviation. Bold/underline mark the highest/second-highest scores per column.

Configuration	VisionZip	PPE	Nuwa
Expanded	74.63	33.40	80.94
Random recovery	74.57 ± 0.14	55.40 ± 0.18	81.54 ± 0.19
Importance-only	74.61	**68.35**	81.43
Spatial-only	<u>75.51</u>	60.70	**82.02**
TokenGraft (§4)	**75.82**	<u>66.97</u>	<u>81.73</u>

For PPE, random recovery reaches 55.40% accuracy versus 33.40% for native expansion (Table 1). Random recovery also slightly improves Nuwa. For VisionZip it is approximately neutral, whereas spatial-only recovery reaches 75.51%. The strongest selector differs by compressor, but these simple rules already show that recovered original tokens can use the available slots more effectively than native expansion.

3.3 From diagnosis to a selection principle

The controls identify two useful signals for recovery: visual importance and spatial complementarity. Visual importance favors tokens that carry useful content. Distance from the compressor's current spatial references favors locations that are not already well covered. TokenGraft combines these signals in one fixed rule, allowing the same recovery mechanism to operate across different compressors without compressor-specific tuning.

4 TokenGraft

4.1 Overview

TokenGraft starts from a compressed visual sequence and the original tokens at the compression interface, where visual tokens enter the compressor. As shown in Figure 2, TokenGraft computes visual importance and spatial complementarity, iteratively selects A original tokens while updating the reference locations, and interleaves the selected tokens with the compressed sequence. The recovered tokens keep their original features and positional information. No model parameter is trained or changed.

For each original token i, we retain its feature x_i, native positional information p_i, original sequence index a_i, image coordinate $u_i \in [0, 1]^2$, and attention-derived visual-importance score s_i. A compressor provides three pieces of information to the recovery module: its base representation Z, the set of original-token candidates C_0, and spatial references R_0 associated with the locations of its retained representatives or aggregates.

VisionZip uses the original positions of its dominant and contextual representatives. PPE uses cluster centroids. Nuwa uses the positions retained after its first stage and recovers from original tokens omitted at that stage. These references locate the currently represented information and provide the starting points for measuring spatial complementarity.

4.2 Selecting original tokens

At iteration t, let R_t be the current reference locations and C_t the remaining original-token candidates. We quantify spatial complementarity of candidate i by its distance to the nearest reference location in image coordinates:

$$d_i^{(t)} = \min_{u \in R_t} \|u_i - u\|_2, \qquad i \in C_t. \tag{3}$$

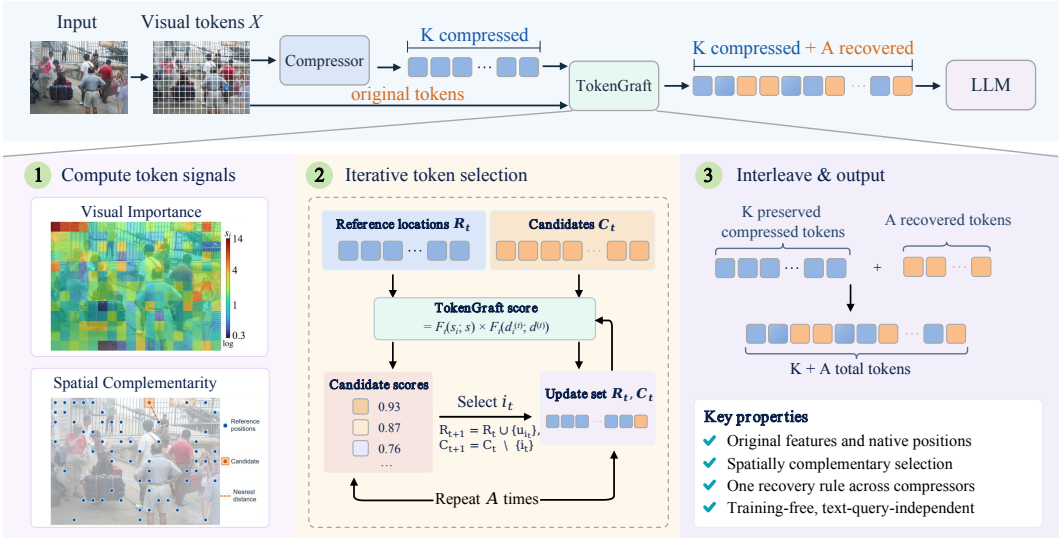


Figure 2: Overview of TokenGraft. Visual importance and spatial complementarity, measured by nearest-reference distances in image coordinates (Equation 3), guide iterative rank-product selection (Equation 6). The A recovered tokens retain their original features and native positional information and are interleaved with the K compressed tokens by original ordering anchors. The visualizations use a real evaluation example. Candidate scores are illustrative.

A larger distance gives the candidate stronger spatial complementarity. For visual importance, let $A_{qj}^{(h)}$ denote the attention weight returned by head h from query position q to visual token j in the selected vision block. We first compute

$$\tilde{s}_j = \frac{1}{H} \sum_{h=1}^{H} \sum_{q \in \mathcal{Q}} A_{qj}^{(h)}, \tag{4}$$

then align $\tilde{s}_j$ to the compression interface. For Qwen2.5-VL, $\mathcal{Q}$ contains the visual query positions from the final full-attention vision block, and the four scores within each 2×2 PatchMerger group are averaged to obtain s_i. Appendix A.2 details visual-importance extraction and token alignment for both transfer backbones. Both s_i and $d_i^{(t)}$ are image-side signals and are independent of the text query and target annotation.

Because visual-importance scores and spatial distances have different numerical scales, we combine their ranks rather than their raw values. For a candidate value v_i, define its empirical cumulative rank among the remaining candidates as

$$F_t(v_i; v) = \frac{1}{|C_t|} \sum_{j \in C_t} \mathbf{1}[v_j \le v_i]. \tag{5}$$

We then choose the candidate with the largest product of visual-importance rank and spatial-complementarity rank:

$$i_t = \arg\max_{i \in C_t} F_t(s_i; s)\, F_t\left(d_i^{(t)}; d^{(t)}\right). \tag{6}$$

After selecting i_t, we add its image location u_{i_t} to R_t and remove the token from C_t. The updated reference set spreads subsequent selections across the image. Ties favor a larger raw visual-importance score and then a smaller original sequence index a_i. The rank product uses no learned weights or compressor-specific tuning. Appendix A gives pseudocode and complexity bounds.

4.3 PUTTING RECOVERED TOKENS BACK INTO THE SEQUENCE

TokenGraft inserts each recovered token using its original ordering anchor and native positional information. VisionZip and Nuwa representatives use their original sequence indices as anchors. PPE clusters use the mean original index of their members, while a recovered token uses its own index. Stable sorting interleaves compressed and recovered tokens by these anchors while jointly permuting their positional fields. The feature vectors themselves are preserved.

Qwen keeps the recovered token's original M-RoPE values; InternVL2.5 and LLaVA-OneVision keep their native one-dimensional positional interfaces. For tiled or multi-view inputs, spatial-reference coordinates are mapped back to the original image. Nuwa inserts recovered original tokens after stage 1 and protects them during stage 2, while its native text-guided pruning still operates on the remaining tokens. The image-only recovered set can be reused for multiple text queries about an unchanged image.

5 EXPERIMENTS

5.1 EXPERIMENTAL SETUP

Our main testbed is Qwen2.5-VL-7B-Instruct (Bai et al., 2025) with VisionZip, PPE, and Nuwa. We evaluate referring-expression grounding on RefCOCO and RefCOCO+ UNC testA+testB (Yu et al., 2016), and RefCOCOg UMD val+test (Mao et al., 2016; Nagaraja et al., 2016). The fixed, deduplicated evaluation lists contain 10,143, 9,937, and 14,466 records, respectively. We report expression-weighted Acc@0.5 with strict, unrounded IoU > 0.5, counting invalid outputs as errors. The main base retention rates are $12.5\%, 25\%, 37.5\%$, with $A = 16$ recovered tokens. For Nuwa, we match both final and intermediate token counts between native expansion and recovery.

Transfer uses InternVL2.5-8B (Chen et al., 2024b), LLaVA-OneVision-Qwen2-7B (OV1) (Li et al., 2025), and Qwen on CV-Bench (Tong et al., 2024). We report pointwise 95% intervals from 10,000 paired bootstrap replicates, clustered by original image and stratified by official split or CV-Bench source. Appendix A details the protocols.

5.2 DOES RECOVERY BEAT NATIVE EXPANSION AT THE SAME BUDGET?

For each compressor, dataset, and base retention rate, we compare K compressed tokens plus $A = 16$ recovered tokens against native expansion to $K + 16$. This tests one recovery allocation across different total budgets. The gain rows in Table 2 are positive in all 27 comparisons, ranging from 0.09 to 33.57 percentage points. PPE gains 17.72–33.57 points. VisionZip improves by 0.98–3.24 points at 12.5% retention and 1.31–1.71 points at 25%, while Nuwa also improves at these budgets. At 37.5%, VisionZip and Nuwa gains narrow to 0.09–0.36 points and their intervals cross zero (Table 5). For both compressors, the gap averaged across the three datasets is largest under aggressive compression and narrows as the native budget grows.

The selector controls further distinguish recovery construction from selection within that construction. On PPE / RefCOCO, random recovery reaches 55.40%, versus 33.40% for native expansion. TokenGraft reaches 66.97%, a further 11.57 points above the random mean under the same insertion procedure. Thus which original tokens are restored also matters.

5.3 HOW SHOULD A FIXED TOKEN BUDGET BE ALLOCATED?

The main experiment fixes $A = 16$ and varies the total budget. We now hold each image's total fixed at $B = \lceil 0.25N \rceil + 16$ and vary $A \in \{0, 9, 16, 25\}$. For $A > 0$, we compare

$$Z_{\mathrm{TG}}(B, A) = \mathrm{Insert}(\mathcal{C}_{B-A}(X), X_{G_A}) \quad \text{with} \quad \mathcal{C}_B(X). \tag{7}$$

At $A = 0$, we directly use $\mathcal{C}_B(X)$ in its native output order. Each nonzero A transfers that many slots to recovery. Nuwa's intermediate total is also fixed. The $A = 16$ point has the same budget allocation as the corresponding main setting.

Every nonzero allocation in Figure 3 improves over $A = 0$, with interval lower bounds above zero. At $A = 25$, VisionZip rises from 74.55% to 77.53%, PPE from 33.31% to 68.69%, and Nuwa from

Table 2: Matched-budget grounding on Qwen2.5-VL ($A = 16$). Expanded denotes native expansion; r is the base retention before recovery. Bold/underline mark the higher/lower paired accuracy; † marks a pointwise 95% paired interval above zero. Complete intervals appear in Table 5.

Compressor	Configuration / metric	RefCOCO	RefCOCO+	RefCOCOg
Base retention $r = 12.5\%$				
VisionZip	Expanded (%)	<u>50.91</u>	<u>45.23</u>	<u>45.85</u>
	TokenGraft (%)	**53.89**	**48.48**	**46.83**
	Gain (pp)	$+2.98^{\dagger}$	$+3.24^{\dagger}$	$+0.98$
PPE	Expanded (%)	<u>14.52</u>	<u>13.18</u>	<u>13.22</u>
	TokenGraft (%)	**37.00**	**33.43**	**30.95**
	Gain (pp)	$+22.48^{\dagger}$	$+20.25^{\dagger}$	$+17.72^{\dagger}$
Nuwa	Expanded (%)	<u>73.42</u>	<u>64.49</u>	<u>69.55</u>
	TokenGraft (%)	**74.58**	**65.91**	**71.00**
	Gain (pp)	$+1.16^{\dagger}$	$+1.42^{\dagger}$	$+1.45^{\dagger}$
Base retention $r = 25\%$				
VisionZip	Expanded (%)	<u>74.85</u>	<u>67.75</u>	<u>70.75</u>
	TokenGraft (%)	**76.49**	**69.46**	**72.06**
	Gain (pp)	$+1.64^{\dagger}$	$+1.71^{\dagger}$	$+1.31^{\dagger}$
PPE	Expanded (%)	<u>33.40</u>	<u>30.83</u>	<u>32.17</u>
	TokenGraft (%)	**66.97**	**62.18**	**60.90**
	Gain (pp)	$+33.57^{\dagger}$	$+31.35^{\dagger}$	$+28.73^{\dagger}$
Nuwa	Expanded (%)	<u>80.94</u>	<u>73.72</u>	<u>78.27</u>
	TokenGraft (%)	**81.73**	**74.51**	**78.76**
	Gain (pp)	$+0.79^{\dagger}$	$+0.78^{\dagger}$	$+0.48$
Base retention $r = 37.5\%$				
VisionZip	Expanded (%)	<u>83.35</u>	<u>76.47</u>	<u>80.47</u>
	TokenGraft (%)	**83.46**	**76.65**	**80.83**
	Gain (pp)	$+0.11$	$+0.18$	$+0.36$
PPE	Expanded (%)	<u>47.92</u>	<u>44.60</u>	<u>46.18</u>
	TokenGraft (%)	**79.46**	**73.52**	**73.95**
	Gain (pp)	$+31.54^{\dagger}$	$+28.92^{\dagger}$	$+27.77^{\dagger}$
Nuwa	Expanded (%)	<u>85.54</u>	<u>78.28</u>	<u>83.24</u>
	TokenGraft (%)	**85.80**	**78.37**	**83.35**
	Gain (pp)	$+0.27$	$+0.09$	$+0.10$

80.94% to 82.18%. Reallocating slots from native compression to recovery therefore improves the use of a fixed budget. Table 7 gives the complete sweep.

Conditioning recovery on the compressed representation. The compressor's spatial references also affect recovery. Removing these references from selection (Without base references) lowers RefCOCO accuracy by 1.16 points for VisionZip and 5.49 for PPE, with positive paired intervals for restoring the references. Nuwa changes by 0.34 points with an interval crossing zero (Table 10).

5.4 Does fixed-budget reallocation transfer?

We extend the same four-point sweep to two additional backbones on RefCOCO and to Qwen on CV-Bench. Grounding keeps $B = \lceil 0.25N \rceil + 16$; CV-Bench uses $B = \lceil 0.125N \rceil + 16$. Relevant Nuwa intermediate totals remain fixed throughout each sweep.

Across backbones. On InternVL2.5, all nine nonzero allocations improve over their all-native baselines with intervals above zero (Table 3). Gains span 1.47–15.53 points across the three compressors. OV1 also has nine positive point estimates: VisionZip and Nuwa gain less than one point with intervals crossing zero, whereas PPE gains 20.03–20.80 points. Fixed-budget reallocation thus

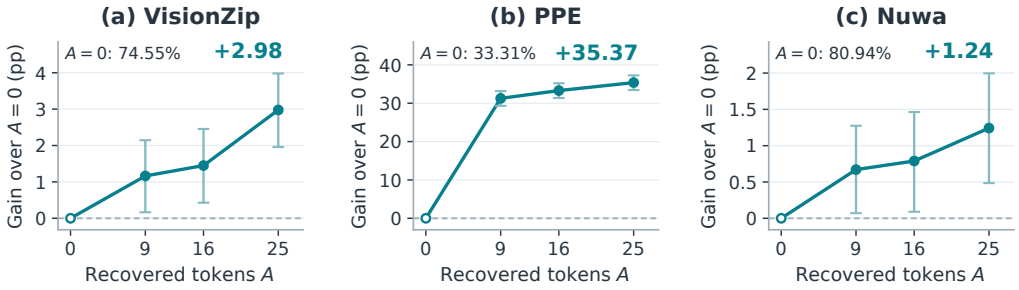


Figure 3: Fixed-budget reallocation on Qwen / RefCOCO. Curves show accuracy gains over $A = 0$ as recovered original tokens replace compressed tokens. Final counts and Nuwa's intermediate total remain fixed. Error bars show pointwise 95% paired intervals. All nine nonzero allocations improve over $A = 0$. Vertical scales differ.

Table 3: Fixed-budget reallocation beyond Qwen grounding. Gains are relative to each row's $A = 0$ baseline. Bold/underline mark the highest/second-highest gains per row; † marks a positive pointwise 95% paired interval. Full intervals appear in Tables 8 and 13.

Compressor	**Baseline (%)**	**Gain over $A = 0$ (pp)**		
	$A = 0$	$A = 9$	$A = 16$	$A = 25$
InternVL2.5 / RefCOCO				
VisionZip	36.66	$+1.65^{\dagger}$	$\underline{+8.74}^{\dagger}$	$\mathbf{+15.53}^{\dagger}$
PPE	68.25	$\mathbf{+8.32}^{\dagger}$	$\underline{+7.62}^{\dagger}$	$+7.28^{\dagger}$
Nuwa	61.45	$+1.47^{\dagger}$	$\underline{+2.16}^{\dagger}$	$\mathbf{+3.04}^{\dagger}$
LLaVA-OneVision / RefCOCO				
VisionZip	50.41	$\underline{+0.57}$	$+0.23$	$\mathbf{+0.65}$
PPE	22.14	$\mathbf{+20.80}^{\dagger}$	$\underline{+20.73}^{\dagger}$	$+20.03^{\dagger}$
Nuwa	58.75	$\mathbf{+0.45}$	$+0.07$	$\underline{+0.09}$
Qwen2.5-VL / CV-Bench				
VisionZip	76.05	$\underline{+0.08}$	$\mathbf{+0.22}$	-0.32
PPE	71.76	$\mathbf{+1.64}^{\dagger}$	$\underline{+1.54}^{\dagger}$	$+1.13$
Nuwa	78.32	$\underline{+0.20}$	$\mathbf{+0.28}$	$+0.12$

extends beyond Qwen, with compressor-dependent responses on each backbone. On InternVL, VisionZip's improvement grows from 1.65 points at $A = 9$ to 15.53 at $A = 25$, whereas PPE has its largest observed improvement at $A = 9$. A fixed recovery rule therefore supports different useful allocations across compressors.

OV1/PPE's large improvement reflects a strong contribution from sequence ordering. At $A = 16$, an original-anchor-sorted compression control captures most of the gain, while TokenGraft further improves RefCOCO, RefCOCO+, and RefCOCOg by 0.47, 1.24, and 0.16 points. The RefCOCO+ interval is above zero (Appendix B.5). The full method combines this sequence organization with original-feature and position recovery. Complete three-dataset transfer results at $A = 16$ appear in Table 6.

Across task formats. CV-Bench tests counting and spatial relations through multiple-choice questions. Eight of nine nonzero allocations have positive overall point estimates. PPE improves by 1.64 and 1.54 points at $A = 9, 16$, with intervals above zero. VisionZip and Nuwa change less, and VisionZip at $A = 25$ has a -0.32-point estimate. These results extend fixed-budget recovery to CV-Bench, with compressor-dependent responses to the recovered share. Appendix C gives the full sweep and the complementary $A = 16$ study across retention rates.

Table 4: Incremental TokenGraft–Expanded cost on Qwen / RTX 3090 at $r = 25\%$, $A = 16$. Each compressor uses 100 images and two balanced-order rounds with exactly 48 output tokens. Paired paths match visual-token trajectories. Overhead is the relative increase in full-request time.

Compressor	ΔTTFT (ms)	ΔRequest (ms)	Overhead (%)	ΔPeak (MiB)
VisionZip	+18.25	+17.98	+0.82	+0.00
PPE	+21.99	+21.00	+0.80	+301.05
Nuwa	+9.34	+7.65	+0.33	+0.20

5.5 What does recovery cost?

We measure TokenGraft against native expansion on Qwen / RTX 3090 at $r = 25\%$, $A = 16$, matching final and intermediate token counts. Each compressor uses one session with 100 images, two balanced-order rounds, and exactly 48 generated tokens per request. Table 4 reports incremental full-request time below 1% for all three compressors: 0.82% for VisionZip, 0.80% for PPE, and 0.33% for Nuwa. TTFT increases by 18.25, 21.99, and 9.34 ms, respectively. PPE requires about 301 MiB more peak allocated memory. A normal-generation check also yields less than 1% incremental request time. Appendix D reports paired intervals, component measurements, and same-session Dense / Native / Expanded / TokenGraft references.

6 Conclusion

Visual token compression depends on both budget size and the representation assigned to that budget. Our matched-budget diagnostic reveals a recoverable gap, and TokenGraft addresses it by restoring visually important, spatially complementary original tokens. Multi-budget evaluations and fixed-budget sweeps show that this hybrid construction can outperform native expansion across compressors, backbones, and tasks. At $r = 25\%$ and $A = 16$, paired runtime measurements show less than 1% incremental full-request time. Together, these results make allocation between compressed and original information an explicit design choice for visual token compression.

AI Use Statement

We used ChatGPT and OpenAI Codex to assist with research ideation, experimental design and refinement, literature search and identification of related work, implementation-related tasks, and interpretation of experimental results. These tools also assisted with drafting and revising manuscript sections, improving readability, translation for author review, and LaTeX preparation. AI-generated reviewer-style feedback was used to identify issues and guide manuscript revision.

Reproducibility Statement

Section 3 defines the comparison and Section 4 specifies the selector and interfaces. Appendix A gives model, input, generation, and implementation conventions; Appendix B provides the paired grounding results and controls supporting the claims. Appendices C and D specify cross-task evaluation and cost measurements. We plan to release the implementation and evaluation protocol with the configurations required to reproduce the comparisons.

References

Saeed Ranjbar Alvar, Gursimran Singh, Mohammad Akbari, and Yong Zhang. DivPrune: Diversity-based visual token pruning for large multimodal models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 9392–9401, 2025. URL https://openaccess.thecvf.com/content/CVPR2025/html/Alvar_DivPrune_Diversity-based_Visual_Token_Pruning_for_Large_Multimodal_Models_CVPR_2025_paper.html.

Shuai Bai, Keqin Chen, Xuejing Liu, Jialin Wang, Wenbin Ge, Sibo Song, Kai Dang, Peng Wang, Shijie Wang, Jun Tang, Humen Zhong, Yuanzhi Zhu, Mingkun Yang, Zhaohai Li, Jianqiang Wan,

Pengfei Wang, Wei Ding, Zheren Fu, Yiheng Xu, Jiabo Ye, Xi Zhang, Tianbao Xie, Zesen Cheng, Hang Zhang, Zhibo Yang, Haiyang Xu, and Junyang Lin. Qwen2.5-VL technical report. *arXiv preprint arXiv:2502.13923*, 2025. URL https://arxiv.org/abs/2502.13923.

Mu Cai, Jianwei Yang, Jianfeng Gao, and Yong Jae Lee. Matryoshka multimodal models. In *International Conference on Learning Representations (ICLR)*, 2025. URL https://proceedings.iclr.cc/paper_files/paper/2025/hash/722fcbc1a6667f2075d75ea79a1b3552-Abstract-Conference.html.

Liang Chen, Haozhe Zhao, Tianyu Liu, Shuai Bai, Junyang Lin, Chang Zhou, and Baobao Chang. An image is worth 1/2 tokens after layer 2: Plug-and-play inference acceleration for large vision-language models. In *European Conference on Computer Vision (ECCV)*, pp. 19–35, 2024a. doi: 10.1007/978-3-031-73004-7_2.

Yi Chen, Jian Xu, Xu-Yao Zhang, Wen-Zhuo Liu, Yang-Yang Liu, and Cheng-Lin Liu. Recoverable compression: A multimodal vision token recovery mechanism guided by text information. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 39, pp. 2293–2301, 2025. doi: 10.1609/aaai.v39i2.32229.

Zhe Chen, Weiyun Wang, Yue Cao, Yangzhou Liu, Zhangwei Gao, Erfei Cui, Jinguo Zhu, Shenglong Ye, Hao Tian, Zhaoyang Liu, Lixin Gu, Xuehui Wang, Qingyun Li, Yiming Ren, Zixuan Chen, Jiapeng Luo, Jiahao Wang, Tan Jiang, Bo Wang, Conghui He, Botian Shi, Xingcheng Zhang, Han Lv, Yi Wang, Wenqi Shao, Pei Chu, Zhongying Tu, Tong He, Zhiyong Wu, Huipeng Deng, Jiaye Ge, Kai Chen, Kaipeng Zhang, Limin Wang, Min Dou, Lewei Lu, Xizhou Zhu, Tong Lu, Dahua Lin, Yu Qiao, Jifeng Dai, and Wenhai Wang. Expanding performance boundaries of open-source multimodal models with model, data, and test-time scaling. *arXiv preprint arXiv:2412.05271*, 2024b. URL https://arxiv.org/abs/2412.05271.

Jinhong Deng, Wen Li, Joey Tianyi Zhou, and Yang He. SCOPE: Saliency-coverage oriented token pruning for efficient multimodel LLMs. In *Advances in Neural Information Processing Systems*, volume 38, 2025. URL https://proceedings.neurips.cc/paper_files/paper/2025/hash/ec6b4456c2bdfd04002d7984043c4936-Abstract-Conference.html.

Chenxi Du, Yongheng Deng, Jiani Liu, Yujia Zhang, Xi Chen, and Ju Ren. CoIn: Coverage and informativeness-guided token reduction for efficient large multimodal models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 10492–10501, 2026. URL https://openaccess.thecvf.com/content/CVPR2026/html/Du_CoIn_Coverage_and_Informativeness-Guided_Token_Reduction_for_Efficient_Large_Multimodal_CVPR_2026_paper.html.

Mark Endo, Xiaohan Wang, and Serena Yeung-Levy. Feather the throttle: Revisiting visual token pruning for vision-language model acceleration. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 22826–22835, 2025. URL https://openaccess.thecvf.com/content/ICCV2025/html/Endo_Feather_the_Throttle_Revisiting_Visual_Token_Pruning_for_Vision-Language_Model_ICCV_2025_paper.html.

Mouxiao Huang, Borui Jiang, Dehua Zheng, Hailin Hu, Kai Han, and Xinghao Chen. PPE: Positional preservation embedding for token compression in multimodal large language models. In *International Conference on Learning Representations (ICLR)*, 2026a. URL https://proceedings.iclr.cc/paper_files/paper/2026/hash/6e73c39cc428c7d264d9820319f31e79-Abstract-Conference.html.

Wenxuan Huang, Zijie Zhai, Yunhang Shen, Shaosheng Cao, Fei Zhao, Xiangfeng Xu, Zheyu Ye, and Shaohui Lin. Dynamic-LLaVA: Efficient multimodal large language models via dynamic vision-language context sparsification. In *International Conference on Learning Representations (ICLR)*, 2025. URL https://proceedings.iclr.cc/paper_files/paper/2025/hash/aeafb73dfed3007ec5299be1604d6f99-Abstract-Conference.html.

Yihong Huang, Fei Ma, Yihua Shao, Jingcai Guo, Zitong Yu, Laizhong Cui, and Qi Tian. Nüwa: Mending the spatial integrity torn by VLM token pruning. In *International Conference on Learning Representations (ICLR)*, 2026b. URL https://proceedings.iclr.cc/paper_files/paper/2026/hash/a46156bd3579c3b268108ea6aca71d13-Abstract-Conference.html.

Lei Lei, Jie Gu, Xiaokang Ma, Chu Tang, Jingmin Chen, and Tong Xu. Task-related token compression in multimodal large language models from an explainability perspective. In *International Conference on Learning Representations (ICLR)*, 2026. URL https://proceedings.iclr.cc/paper_files/paper/2026/hash/b74c141c9f7bbac5f250f73b0c84a478-Abstract-Conference.html.

Bo Li, Yuanhan Zhang, Dong Guo, Renrui Zhang, Feng Li, Hao Zhang, Kaichen Zhang, Peiyuan Zhang, Yanwei Li, Ziwei Liu, and Chunyuan Li. LLaVA-OneVision: Easy visual task transfer. *Transactions on Machine Learning Research*, 2025. URL https://openreview.net/forum?id=zKv8qULV6n.

Junhua Mao, Jonathan Huang, Alexander Toshev, Oana Camburu, Alan L. Yuille, and Kevin Murphy. Generation and comprehension of unambiguous object descriptions. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 11–20, 2016. URL https://openaccess.thecvf.com/content_cvpr_2016/html/Mao_Generation_and_Comprehension_CVPR_2016_paper.html.

Varun K. Nagaraja, Vlad I. Morariu, and Larry S. Davis. Modeling context between objects for referring expression understanding. In *European Conference on Computer Vision (ECCV)*, pp. 792–807, 2016. doi: 10.1007/978-3-319-46493-0_48.

Yuzhang Shang, Mu Cai, Bingxin Xu, Yong Jae Lee, and Yan Yan. LLaVA-PruMerge: Adaptive token reduction for efficient large multimodal models. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 22857–22867, 2025. URL https://openaccess.thecvf.com/content/ICCV2025/html/Shang_LLaVA-PruMerge_Adaptive_Token_Reduction_for_Efficient_Large_Multimodal_Models_ICCV_2025_paper.html.

Aditya Kumar Singh, Hitesh Kandala, Pratik Prabhanjan Brahma, Zicheng Liu, and Emad Barsoum. DUET-VLM: Dual stage unified efficient token reduction for VLM training and inference. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 17651–17660, 2026. URL https://openaccess.thecvf.com/content/CVPR2026/html/Singh_DUET-VLM_Dual_stage_Unified_Efficient_Token_reduction_for_VLM_Training_CVPR_2026_paper.html.

Zixun Sun, Yubo Dong, Hehe Fan, and Yi Yang. Hi-lo prune: Look at what you'll lose before pruning with hierarchical token selection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 31941–31951, 2026. URL https://openaccess.thecvf.com/content/CVPR2026/html/Sun_Hi-Lo_Prune_Look_at_What_Youll_Lose_before_Pruning_with_CVPR_2026_paper.html.

Shengbang Tong, Ellis Brown, Penghao Wu, Sanghyun Woo, Manoj Middepogu, Sai Charitha Akula, Jihan Yang, Shusheng Yang, Adithya Iyer, Xichen Pan, Austin Wang, Rob Fergus, Yann LeCun, and Saining Xie. Cambrian-1: A fully open, vision-centric exploration of multimodal LLMs. In *Advances in Neural Information Processing Systems*, volume 37, 2024. URL https://proceedings.neurips.cc/paper_files/paper/2024/hash/9ee3a664ccfeabc0da16ac6f1f1cfe59-Abstract-Conference.html.

Long Xing, Qidong Huang, Xiaoyi Dong, Jiajie Lu, Pan Zhang, Yuhang Zang, Yuhang Cao, Conghui He, Jiaqi Wang, Feng Wu, and Dahua Lin. Conical visual concentration for efficient large vision-language models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 14593–14603, 2025. URL https://openaccess.thecvf.com/content/CVPR2025/html/Xing_Conical_Visual_Concentration_for_Efficient_Large_Vision-Language_Models_CVPR_2025_paper.html.

Tong Xu, Hailong Shi, and Xingyu Gao. SCoRe: Salience-coverage reduction for vision token pruning in vision-language models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 24686–24695, 2026. URL https://openaccess.thecvf.com/content/CVPR2026/html/Xu_SCoRe_Salience-Coverage_Reduction_for_Vision_Token_Pruning_in_Vision-Language_Models_CVPR_2026_paper.html.

Cheng Yang, Yang Sui, Jinqi Xiao, Lingyi Huang, Yu Gong, Chendi Li, Jinghua Yan, Yu Bai, Ponnuswamy Sadayappan, Xia Hu, and Bo Yuan. TopV: Compatible token pruning with inference time optimization for fast and low-memory multimodal vision language model. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 19803–19813, 2025a. URL https://openaccess.thecvf.com/content/CVPR2025/html/Yang_TopV_Compatible_Token_Pruning_with_Inference_Time_Optimization_for_Fast_CVPR_2025_paper.html.

Senqiao Yang, Yukang Chen, Zhuotao Tian, Chengyao Wang, Jingyao Li, Bei Yu, and Jiaya Jia. VisionZip: Longer is better but not necessary in vision language models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 19792–19802, 2025b. URL https://openaccess.thecvf.com/content/CVPR2025/html/Yang_VisionZip_Longer_is_Better_but_Not_Necessary_in_Vision_Language_CVPR_2025_paper.html.

Sihan Yang, Runsen Xu, Chenhang Cui, Tai Wang, Dahua Lin, and Jiangmiao Pang. VFlowOpt: A token pruning framework for LMMs with visual information flow-guided optimization. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 23924–23934, 2025c. URL https://openaccess.thecvf.com/content/ICCV2025/html/Yang_VFlowOpt_A_Token_Pruning_Framework_for_LMMs_with_Visual_Information_ICCV_2025_paper.html.

Weihao Ye, Qiong Wu, Wenhao Lin, and Yiyi Zhou. Fit and prune: Fast and training-free visual token pruning for multi-modal large language models. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 39, pp. 22128–22136, 2025a. doi: 10.1609/aaai.v39i21.34366.

Xubing Ye, Yukang Gan, Yixiao Ge, Xiao-Ping Zhang, and Yansong Tang. ATP-LLaVA: Adaptive token pruning for large vision language models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 24972–24982, 2025b. URL https://openaccess.thecvf.com/content/CVPR2025/html/Ye_ATP-LLaVA_Adaptive_Token_Pruning_for_Large_Vision_Language_Models_CVPR_2025_paper.html.

Licheng Yu, Patrick Poirson, Shan Yang, Alexander C. Berg, and Tamara L. Berg. Modeling context in referring expressions. In *European Conference on Computer Vision (ECCV)*, pp. 69–85, 2016. doi: 10.1007/978-3-319-46475-6_5.

Yuan Zhang, Chun-Kai Fan, Junpeng Ma, Wenzhao Zheng, Tao Huang, Kuan Cheng, Denis A. Gudovskiy, Tomoyuki Okuno, Yohei Nakata, Kurt Keutzer, and Shanghang Zhang. SparseVLM: Visual token sparsification for efficient vision-language model inference. In *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pp. 74840–74857, 2025. URL https://proceedings.mlr.press/v267/zhang25s.html.

A Implementation and Evaluation Protocols

A.1 Datasets, budgets, and scoring

Grounding uses COCO train2014 images and fixed, deduplicated expression lists from the official splits. We use the `omlab/VLM-R1` JSON annotations used by Nuwa. Within each split, we retain the first occurrence in source order of each exact tuple of image-path string, complete grounding prompt (including the expression), and ground-truth box in original-image pixel coordinates. Strings are matched exactly, and the four box coordinates are compared exactly after conversion to floating-point values. No text normalization or coordinate tolerance is applied. Records with different prompts or target boxes remain distinct. All methods use the same frozen lists and record order.

After deduplication, RefCOCO UNC testA/testB contains 5,349/4,794 expressions (10,143 total) and 1,500 image clusters; RefCOCO+ UNC testA+testB contains 9,937 expressions and 1,500 clusters; RefCOCOg UMD val+test contains 14,466 expressions and 3,900 clusters. We sum correct expressions over the combined splits rather than average split accuracies equally. Multiple expressions for one image remain together in the statistical resampling. Predictions are mapped to original-image coordinates and scored by the frozen Nuwa-aligned discrete IoU implementation with strict unrounded IoU > 0.5. All invalid outputs remain in the denominator.

N is measured at the compression interface: after Qwen's PatchMerger, or after the native feature projection/pixel shuffle of the transfer backbone. $K = \lceil rN \rceil$ is rounded separately for each image. For main base rates $r = 12.5\%, 25\%, 37.5\%$, Nuwa uses first-stage budgets $S = \lceil 0.25N \rceil, \lceil 0.35N \rceil, \lceil 0.475N \rceil$, respectively. Both recovery and Expanded use $S + A \to K + A$. Transfer uses the middle configuration. LLaVA-OneVision's K and A count patch carriers; preserved image-newline tokens are outside those budgets and are retained in both conditions.

A.2 Backbones and generation

Qwen2.5-VL-7B-Instruct uses BF16, greedy cached generation, and a 48-token output limit for grounding. Qwen uses its native single-image processor and chat template; predictions are interpreted in resized-image coordinates and mapped back with the native `smart_resize` convention. PPE uses `ppe_k=8`. Deterministic settings and software/hardware environments are held fixed within each paired comparison.

InternVL2.5-8B uses its native 448×448 dynamic tiling pipeline (up to 12 local tiles plus a thumbnail when applicable), BF16, and a 64-token output limit. **LLaVA-OneVision-Qwen2-7B** uses native SigLIP 384-pixel views with the unpadded `anyres_max_9` layout, FP16, the `qwen_1_5` prompt, and a 64-token limit. Both use greedy cached generation. InternVL predictions follow the model's supported 0–1000/0–1 coordinate conventions; LLaVA-OneVision uses a fixed 0–1 convention. Invalid parses remain errors rather than being removed from evaluation.

Visual importance on transfer backbones. InternVL2.5 uses the last InternViT block's Q/K with native normalization and scaling, excluding CLS queries and keys. FP32 softmax, head averaging, and patch-query summation precede 2×2 spatial averaging to the native v2 pixel-shuffled 16×16 grid. Tiles and the optional thumbnail are processed separately and concatenated in native order.

OV1 uses executed SigLIP block 26 (zero-based index 25), with 27×27 CLS-free patches per 384-pixel view. Native Q/K operations are followed by FP32 softmax, head averaging, and query summation. Scores follow the features' AnyRes layout, unpadding, and required bilinear resizing before concatenating the global view and local grid, excluding newline tokens.

A.3 Interface details

VisionZip's candidate set C_0 excludes the original indices of dominant and contextual representatives; tokens used only in aggregation remain eligible. PPE admits all original tokens, restricted to patches with valid image content for OV1. Nuwa recovers from original indices omitted after stage 1.

For tiled or multi-view transfer inputs, spatial references are measured in original-image coordinates. LLaVA-OneVision limits candidates to valid image content; references use valid-content representatives or content-support-weighted cluster centroids. In the TokenGraft path, PPE keeps its cluster position slots while stable original-anchor ordering jointly permutes features and position fields. OV1/PPE native compression, including the $A = 0$ sweep condition, preserves the official DPC cluster output order. The separate Expanded + sorting control changes that order to the mean original sequence index of each cluster, without recovery. For the transfer models, Nuwa keeps its native two-dimensional neighborhood candidate rule; recovered tokens are protected through stage 2, and the native positional interfaces of each backbone are retained.

A.4 Selector pseudocode and computational bounds

Algorithm 1 Original-token recovery with TokenGraft

Require: $Z, C_0, R_0, \{x_i, p_i, a_i, u_i, s_i\}, A; |C_0| \geq A$
$C \leftarrow C_0, R \leftarrow R_0, G \leftarrow \varnothing$
for $t = 0, \ldots, A-1$ **do**
  if $R = \varnothing$ **then**
    $d_i \leftarrow 1$ for all $i \in C$
  else
    $d_i \leftarrow \min_{u \in R} \|u_i - u\|_2$ for all $i \in C$
  end if
  $q_i \leftarrow F_C(s_i; s)F_C(d_i; d)$ for all $i \in C$
  $i^* \leftarrow \arg\max_{i \in C}(q_i, s_i, -a_i)$ ▷ lexicographic order
  $G \leftarrow G \cup \{i^*\}, R \leftarrow R \cup \{u_{i^*}\}, C \leftarrow C \setminus \{i^*\}$
end for
Copy original features and positions indexed by G
Stably interleave with Z by original anchors; jointly reorder all position fields
return augmented representation and graft mask

The empty-reference branch implements the Without base references control: equal spatial-complementarity ranks make the first selection depend on visual importance, and later steps reference only previously recovered tokens. Random recovery samples distinct candidates with seeds 17, 42, and 73. Spatial-only greedily maximizes the same nearest-reference distance.

Let N be the number of original visual positions, $M = |C_0|$, and $R = |R_0|$, with $R \geq 1$ and $1 \leq A \leq M \leq N$. At step t, $m_t = M - t$ candidates remain. The implementation recomputes both empirical-CDF ranks by pairwise comparisons in $\Theta(m_t^2)$ time and all nearest-reference distances in $\Theta(m_t(R+t))$ time. Summing over A steps gives $O(AM^2 + AMR + A^2M)$ for the selection loop.

The timed selector function also computes nearest-reference distances for all N original positions before and after recovery, and initial-candidate importance percentiles, for diagnostics. Including these diagnostics gives $O(AM^2 + AMR + A^2M + N(R+A))$. These bounds start from available importance scores and coordinates. They exclude visual-importance extraction, visual encoding, compression and reference construction, token insertion and positional reordering, and decoder inference.

B Paired Evidence for the Main Claims

B.1 Statistical definition

For official split s and image i, let n_{si} be its expression count and a_{si} the difference in correct-expression counts between the paired conditions. The expression-weighted difference is

$$\widehat{\Delta} = 100 \frac{\sum_s \sum_i a_{si}}{\sum_s \sum_i n_{si}}. \tag{8}$$

Percentile intervals use 10,000 bootstrap replicates, sampling images with replacement within each official split and retaining all expressions and paired predictions. Seeds are 20260727 for Qwen main/component and fixed-total analyses, and 20260916 for transfer, cross-model sweeps, and OV1/PPE ordering. These pointwise intervals quantify within-comparison image-sampling uncertainty without multiple-comparison correction. Accuracies, differences, and endpoints are independently rounded to two decimals.

Random recovery reports the population standard deviation of three seed accuracies. Paired comparisons use their expression-level mean correctness; seed variation and image-sampling uncertainty are distinct.

Prediction sets. Each evaluation series uses its own paired prediction set, including separate Qwen VisionZip/PPE main and sweep runs. InternVL's sweep partly reuses verified endpoint predictions. All gains and intervals retain their within-series pairing.

B.2 Complete main-study intervals

Table 5 completes all 27 Qwen grounding comparisons in Table 2. Table 6 reports three-dataset transfer accuracies and paired intervals at $A = 16$.

Table 5: All main-study TokenGraft–Expanded gains and pointwise 95% paired intervals. r is the base retention before recovery. † marks intervals above zero; a lower endpoint printed as $+0.00$ is not treated as strictly positive.

Compressor	Dataset	Gain (pp)	95% CI (pp)
Base retention $r = 12.5\%$			
VisionZip	RefCOCO	$+2.98^{\dagger}$	$[+1.65, +4.30]$
	RefCOCO+	$+3.24^{\dagger}$	$[+1.93, +4.54]$
	RefCOCOg	$+0.98$	$[+0.00, +1.95]$
PPE	RefCOCO	$+22.48^{\dagger}$	$[+20.70, +24.25]$
	RefCOCO+	$+20.25^{\dagger}$	$[+18.60, +21.95]$
	RefCOCOg	$+17.72^{\dagger}$	$[+16.56, +18.89]$
Nuwa	RefCOCO	$+1.16^{\dagger}$	$[+0.22, +2.12]$
	RefCOCO+	$+1.42^{\dagger}$	$[+0.45, +2.40]$
	RefCOCOg	$+1.45^{\dagger}$	$[+0.74, +2.17]$
Base retention $r = 25\%$			
VisionZip	RefCOCO	$+1.64^{\dagger}$	$[+0.65, +2.64]$
	RefCOCO+	$+1.71^{\dagger}$	$[+0.69, +2.75]$
	RefCOCOg	$+1.31^{\dagger}$	$[+0.60, +2.01]$
PPE	RefCOCO	$+33.57^{\dagger}$	$[+31.66, +35.48]$
	RefCOCO+	$+31.35^{\dagger}$	$[+29.50, +33.15]$
	RefCOCOg	$+28.73^{\dagger}$	$[+27.46, +30.00]$
Nuwa	RefCOCO	$+0.79^{\dagger}$	$[+0.09, +1.46]$
	RefCOCO+	$+0.78^{\dagger}$	$[+0.04, +1.54]$
	RefCOCOg	$+0.48$	$[-0.07, +1.03]$
Base retention $r = 37.5\%$			
VisionZip	RefCOCO	$+0.11$	$[-0.59, +0.81]$
	RefCOCO+	$+0.18$	$[-0.56, +0.93]$
	RefCOCOg	$+0.36$	$[-0.16, +0.87]$
PPE	RefCOCO	$+31.54^{\dagger}$	$[+29.52, +33.50]$
	RefCOCO+	$+28.92^{\dagger}$	$[+27.09, +30.77]$
	RefCOCOg	$+27.77^{\dagger}$	$[+26.50, +29.04]$
Nuwa	RefCOCO	$+0.27$	$[-0.28, +0.83]$
	RefCOCO+	$+0.09$	$[-0.50, +0.69]$
	RefCOCOg	$+0.10$	$[-0.31, +0.52]$

Table 6: Transfer results ($r = 25\%$, $A = 16$). Gains are TokenGraft (TG) minus Expanded; OV1/PPE Expanded uses native cluster order. Bold/underline mark the higher/lower paired accuracy; † marks positive pointwise 95% paired intervals.

Compressor	Dataset	Accuracy (%)		Paired difference (pp)	
		Expanded	TG	Gain	95% CI
InternVL2.5-8B					
VisionZip	RefCOCO	<u>36.70</u>	**45.15**	$+8.46^{\dagger}$	$[+7.08, +9.83]$
	RefCOCO+	<u>31.91</u>	**40.72**	$+8.81^{\dagger}$	$[+7.47, +10.16]$
	RefCOCOg	<u>34.25</u>	**44.28**	$+10.04^{\dagger}$	$[+8.99, +11.07]$
PPE	RefCOCO	<u>68.29</u>	**75.93**	$+7.64^{\dagger}$	$[+6.61, +8.67]$
	RefCOCO+	<u>61.47</u>	**68.66**	$+7.20^{\dagger}$	$[+6.17, +8.23]$
	RefCOCOg	<u>67.61</u>	**75.63**	$+8.01^{\dagger}$	$[+7.30, +8.75]$
Nuwa	RefCOCO	<u>61.37</u>	**63.76**	$+2.39^{\dagger}$	$[+1.49, +3.27]$
	RefCOCO+	<u>54.00</u>	**56.44**	$+2.44^{\dagger}$	$[+1.55, +3.35]$
	RefCOCOg	<u>59.79</u>	**61.89**	$+2.10^{\dagger}$	$[+1.48, +2.73]$
LLaVA-OneVision-Qwen2-7B					
VisionZip	RefCOCO	<u>50.41</u>	**50.64**	$+0.23$	$[-0.86, +1.30]$
	RefCOCO+	<u>46.81</u>	**47.40**	$+0.58$	$[-0.47, +1.65]$
	RefCOCOg	<u>51.11</u>	**51.71**	$+0.59$	$[-0.19, +1.38]$
PPE	RefCOCO	<u>22.14</u>	**42.88**	$+20.73^{\dagger}$	$[+19.27, +22.21]$
	RefCOCO+	<u>20.12</u>	**40.74**	$+20.62^{\dagger}$	$[+19.22, +22.04]$
	RefCOCOg	<u>21.95</u>	**41.28**	$+19.33^{\dagger}$	$[+18.26, +20.40]$
Nuwa	RefCOCO	<u>58.75</u>	**58.82**	$+0.07$	$[-0.55, +0.68]$
	RefCOCO+	<u>55.03</u>	**55.22**	$+0.19$	$[-0.46, +0.84]$
	RefCOCOg	**58.35**	<u>58.21</u>	-0.14	$[-0.64, +0.35]$

B.3 FIXED-TOTAL-BUDGET SWEEPS

Tables 7 and 8 report the Qwen and transfer-backbone sweeps over $A \in \{0, 9, 16, 25\}$ on 10,143 RefCOCO expressions from 1,500 images. The common totals are $B = \lceil 0.25N \rceil + 16$ and, for Nuwa, $U = \lceil 0.35N \rceil + 16$. The native parts receive $B - A$ and $U - A$ tokens, respectively. OV1 image-newline tokens remain outside the patch budget and identical across allocations. Gains are relative to the corresponding $A = 0$ condition.

Table 7: Qwen / RefCOCO fixed-total sweep. Gains are relative to $A = 0$; bold/underline rank accuracies within each compressor. † marks positive pointwise 95% paired intervals.

Compressor	A	Accuracy (%)	Gain (pp)	95% CI (pp)
VisionZip	0	74.55	—	—
	9	75.72	$+1.16^{\dagger}$	$[+0.17, +2.15]$
	16	<u>76.00</u>	$+1.45^{\dagger}$	$[+0.43, +2.46]$
	25	**77.53**	$+2.98^{\dagger}$	$[+1.96, +3.98]$
PPE	0	33.31	—	—
	9	64.58	$+31.26^{\dagger}$	$[+29.32, +33.16]$
	16	<u>66.61</u>	$+33.29^{\dagger}$	$[+31.35, +35.20]$
	25	**68.69**	$+35.37^{\dagger}$	$[+33.44, +37.26]$
Nuwa	0	80.94	—	—
	9	81.61	$+0.67^{\dagger}$	$[+0.07, +1.27]$
	16	<u>81.73</u>	$+0.79^{\dagger}$	$[+0.09, +1.46]$
	25	**82.18**	$+1.24^{\dagger}$	$[+0.48, +2.00]$

Table 8: Cross-model RefCOCO fixed-total sweeps. Gains are relative to $A = 0$; bold/underline rank accuracies within each model/compressor. † marks positive pointwise 95% paired intervals.

Compressor	A	Accuracy (%)	Gain (pp)	95% CI (pp)
InternVL2.5-8B				
VisionZip	0	36.66	—	—
	9	38.30	$+1.65^{\dagger}$	$[+0.32, +2.97]$
	16	<u>45.39</u>	$+8.74^{\dagger}$	$[+7.34, +10.11]$
	25	**52.18**	$+15.53^{\dagger}$	$[+14.09, +16.99]$
PPE	0	68.25	—	—
	9	**76.57**	$+8.32^{\dagger}$	$[+7.34, +9.32]$
	16	<u>75.87</u>	$+7.62^{\dagger}$	$[+6.63, +8.63]$
	25	75.53	$+7.28^{\dagger}$	$[+6.29, +8.30]$
Nuwa	0	61.45	—	—
	9	62.92	$+1.47^{\dagger}$	$[+0.68, +2.24]$
	16	<u>63.61</u>	$+2.16^{\dagger}$	$[+1.27, +3.03]$
	25	**64.49**	$+3.04^{\dagger}$	$[+2.10, +3.97]$
LLaVA-OneVision-Qwen2-7B				
VisionZip	0	50.41	—	—
	9	<u>50.98</u>	$+0.57$	$[-0.48, +1.63]$
	16	50.64	$+0.23$	$[-0.83, +1.29]$
	25	**51.06**	$+0.65$	$[-0.43, +1.76]$
PPE	0	22.14	—	—
	9	**42.95**	$+20.80^{\dagger}$	$[+19.31, +22.28]$
	16	<u>42.88</u>	$+20.73^{\dagger}$	$[+19.27, +22.21]$
	25	42.18	$+20.03^{\dagger}$	$[+18.60, +21.47]$
Nuwa	0	58.75	—	—
	9	**59.20**	$+0.45$	$[-0.09, +1.00]$
	16	58.82	$+0.07$	$[-0.55, +0.69]$
	25	<u>58.84</u>	$+0.09$	$[-0.60, +0.77]$

B.4 Selector and spatial-reference controls

Table 9 gives the paired comparisons for the selectors in Table 1. Importance-only is strongest for PPE, spatial-only for Nuwa, and TokenGraft for VisionZip.

Table 9: Paired selector controls for Table 1. Gains are TokenGraft minus the listed comparator. Random recovery uses mean expression-level correctness over three seeds. † marks pointwise 95% paired intervals above zero.

Compressor	Comparator	Gain (pp)	95% CI (pp)
VisionZip	Expanded	$+1.18^{\dagger}$	$[+0.18, +2.18]$
	Random recovery	$+1.25^{\dagger}$	$[+0.47, +2.02]$
	Importance-only	$+1.20^{\dagger}$	$[+0.32, +2.07]$
	Spatial-only	$+0.31$	$[-0.53, +1.12]$
PPE	Expanded	$+33.57^{\dagger}$	$[+31.66, +35.48]$
	Random recovery	$+11.57^{\dagger}$	$[+10.29, +12.84]$
	Importance-only	-1.38	$[-2.59, -0.16]$
	Spatial-only	$+6.27^{\dagger}$	$[+4.99, +7.53]$
Nuwa	Expanded	$+0.79^{\dagger}$	$[+0.09, +1.46]$
	Random recovery	$+0.19$	$[-0.37, +0.76]$
	Importance-only	$+0.31$	$[-0.34, +0.96]$
	Spatial-only	-0.29	$[-0.91, +0.35]$

Table 10: Base-reference control on Qwen / RefCOCO ($r = 25\%$, $A = 16$). Without base references omits R_0 from distance computation; base tokens, candidates, importance scores, budget, and insertion are unchanged. Gains are TG minus this control. Bold/underline rank paired accuracies; † marks positive pointwise 95% paired intervals.

Compressor	Accuracy (%)		Paired difference (pp)	
	Without base references	TG	Gain	95% CI
VisionZip	<u>74.92</u>	**76.08**	$+1.16^{\dagger}$	$[+0.28, +2.02]$
PPE	<u>61.48</u>	**66.97**	$+5.49^{\dagger}$	$[+4.19, +6.82]$
Nuwa	<u>81.46</u>	**81.79**	$+0.34$	$[-0.34, +1.02]$

B.5 OV1/PPE SEQUENCE-ORDER CONTROL

Expanded follows the official DPC cluster output order. Expanded + sorting compresses to the same final budget and orders the output by the clusters' original-image sequence anchors. TokenGraft interleaves $B - 16$ compressed tokens with 16 recovered original tokens using the same anchors. Table 11 combines their accuracies and paired differences across the three grounding datasets.

Sorting accounts for most of the gain over Expanded. TokenGraft further improves the three dataset point estimates by 0.47, 1.24, and 0.16 points, with the RefCOCO+ interval above zero.

Table 11: OV1/PPE sequence-order control at $r = 25\%$, $A = 16$. Expanded and Expanded + sorting each compress to $B = \lceil 0.25N \rceil + 16$; the latter uses original-anchor order. TokenGraft uses $B - 16$ compressed and 16 recovered tokens. Bold/underline mark the two highest accuracies per dataset; † marks a pointwise 95% paired interval above zero.

Configuration / metric	RefCOCO	RefCOCO+	RefCOCOg
Accuracy (%)			
Expanded	22.14	20.12	21.95
Expanded + sorting	<u>42.40</u>	<u>39.50</u>	<u>41.12</u>
TokenGraft	**42.88**	**40.74**	**41.28**
Expanded + sorting − Expanded			
Gain (pp)	$+20.26^{\dagger}$	$+19.38^{\dagger}$	$+19.18^{\dagger}$
95% CI (pp)	$[+18.79, +21.70]$	$[+18.03, +20.76]$	$[+18.16, +20.20]$
TokenGraft − (Expanded + sorting)			
Gain (pp)	$+0.47$	$+1.24^{\dagger}$	$+0.16$
95% CI (pp)	$[-0.51, +1.46]$	$[+0.28, +2.18]$	$[-0.54, +0.86]$

C CV-BENCH EVALUATION

C.1 PROTOCOL AND RESULTS ACROSS RETENTION RATES

We evaluate the complete CV-Bench dataset (Tong et al., 2024): 2,638 questions from 2,112 source images spanning Count, Relation, Depth, and Distance. We keep the official images, questions, options, source-balanced scoring, and native Qwen preprocessing. Generation uses BF16, greedy cached decoding, a 16-token limit, and a fixed answer-letter prompt; invalid parses count as errors. Paired bootstrap resampling is performed over source images within each source with 10,000 replicates.

Official scores use source-balanced weighting,

$$a_{2D} = \frac{a_{\mathrm{ADE20K}} + a_{\mathrm{COCO}}}{2}, \quad a_{3D} = a_{\mathrm{Omni3D}}, \quad a_{\mathrm{overall}} = \frac{a_{2D} + a_{3D}}{2}. \tag{9}$$

We test the same $A = 16$ recovery rule at $r = 12.5\%$ and 25%. Five of six overall point estimates are positive, with PPE intervals above zero at both rates. Table 12 reports the complete 2D/3D/overall contrasts and intervals.

Table 12: Dynamic-input CV-Bench TokenGraft–Expanded gains at both tested retention rates ($A = 16$). † marks pointwise 95% paired intervals above zero.

Compressor	Subset	Gain (pp)	95% CI (pp)
Base retention $r = 12.5\%$			
VisionZip	2D	$+0.96$	$[-0.51, +2.45]$
	3D	-0.50	$[-1.96, +0.95]$
	Overall	$+0.23$	$[-0.81, +1.26]$
PPE	2D	$+0.86$	$[-0.87, +2.57]$
	3D	$+2.50^{\dagger}$	$[+0.45, +4.50]$
	Overall	$+1.68^{\dagger}$	$[+0.34, +3.00]$
Nuwa	2D	$+0.26$	$[-0.94, +1.48]$
	3D	$+0.25$	$[-0.60, +1.10]$
	Overall	$+0.25$	$[-0.48, +0.99]$
Base retention $r = 25\%$			
VisionZip	2D	-0.11	$[-1.22, +0.98]$
	3D	-0.17	$[-0.99, +0.66]$
	Overall	-0.14	$[-0.83, +0.55]$
PPE	2D	$+2.10^{\dagger}$	$[+0.44, +3.82]$
	3D	$+2.42^{\dagger}$	$[+0.33, +4.59]$
	Overall	$+2.26^{\dagger}$	$[+0.89, +3.67]$
Nuwa	2D	$+0.00$	$[-1.00, +1.00]$
	3D	$+0.17$	$[-0.63, +0.99]$
	Overall	$+0.08$	$[-0.56, +0.73]$

C.2 Fixed-total-budget allocation sweep

The four allocations $A \in \{0, 9, 16, 25\}$ share a final total $B = \lceil 0.125N \rceil + 16$; Nuwa also shares the intermediate total $U = \lceil 0.25N \rceil + 16$. Each native part receives $B - A$ slots at the final interface and, for Nuwa, $U - A$ slots after stage 1. All four points use the same images and Qwen preprocessing with `min_pixels=57232` and `max_pixels=12845056`. The resulting N spans 77–4,661 and the smallest B is 26, leaving at least one native token even at $A = 25$. These preprocessing bounds differ from the native-input study in Appendix C.1, so the two studies use different prediction sets.

Table 13 reports the complete sweep under the scoring and resampling protocol in Appendix C.1, with bootstrap seed 20260727.

Table 13: Qwen / CV-Bench fixed-total sweep. Overall gains are relative to $A = 0$. Bold/underline rank scores within each compressor and metric; † marks positive pointwise 95% paired intervals.

Compressor	A	Accuracy (%)			Overall difference (pp)	
		2D	3D	Overall	Gain	95% CI
VisionZip	0	70.35	**81.75**	76.05	—	—
	9	70.93	<u>81.33</u>	<u>76.13</u>	$+0.08$	$[-0.90, +1.05]$
	16	**71.29**	81.25	**76.27**	$+0.22$	$[-0.82, +1.25]$
	25	<u>71.04</u>	80.42	75.73	-0.32	$[-1.55, +0.88]$
PPE	0	68.94	74.58	71.76	—	—
	9	**69.88**	<u>76.92</u>	**73.40**	$+1.64^{\dagger}$	$[+0.33, +2.94]$
	16	<u>69.68</u>	<u>76.92</u>	<u>73.30</u>	$+1.54^{\dagger}$	$[+0.15, +2.90]$
	25	68.78	**77.00**	72.89	$+1.13$	$[-0.37, +2.59]$
Nuwa	0	74.31	82.33	78.32	—	—
	9	<u>74.38</u>	<u>82.67</u>	<u>78.52</u>	$+0.20$	$[-0.42, +0.84]$
	16	<u>74.38</u>	**82.83**	**78.61**	$+0.28$	$[-0.44, +1.00]$
	25	**74.39**	82.50	78.44	$+0.12$	$[-0.68, +0.92]$

D Runtime Protocol and Absolute References

D.1 Workload and paired measurement

We use BF16 Qwen2.5-VL-7B-Instruct at batch size one on RTX 3090 GPUs. The workload contains 100 RefCOCO images, 50 per split, with one fixed expression each. Each compressor runs in its own process on a dedicated GPU within a shared host. Its Dense, Native, Expanded, and TokenGraft conditions share one session. Each condition has five warm-up requests. The main measurement uses exactly 48 output tokens and two rounds, yielding 200 paired Expanded/TG requests per compressor. A separate normal-generation check uses 100 image pairs, greedy EOS stopping, and a 48-token cap.

The four conditions are measured per image using balanced orders, reversed in the second round. Expanded and TokenGraft are adjacent, and each condition appears equally often at each order position. Images are reprocessed for every request and no image-method cache is reused. Dense uses stock Qwen. The three compressed conditions use the same loaded model object and the formal compressor implementations.

Native uses $K = \lceil 0.25N \rceil$ and $A = 0$. Expanded and TokenGraft both end with $\lceil 0.25N \rceil + 16$ patch tokens. Nuwa matches the intermediate $\lceil 0.35N \rceil + 16$ total and uses the exact-budget configuration. Paired input grids, final budgets, and fixed output lengths are checked for equality. Layer-wise token/KV trajectories are verified in the separate 20-image profile; Nuwa's main requests additionally retain per-layer traces.

Full-request time includes image reading, preprocessing, visual encoding, compression/recovery, and generation. TTFT ends when the first output token reaches the CPU. Weight loading, hook installation, output-string decoding, input hashing, and file writing are excluded. Peak memory is PyTorch allocated memory, including model weights, rather than reserved memory or whole-device occupancy.

D.2 Absolute references and paired statistics

Table 14 keeps each compressor's Dense reference in its own session. Relative to those Dense means, the complete TokenGraft pipelines change full-request time by -1.92%, $+17.69\%$, and $+2.28\%$ for VisionZip, PPE, and Nuwa. These are pipeline-level comparisons; the paired Expanded comparator measures recovery's incremental cost.

Table 14: Same-session absolute runtime references with exactly 48 generated tokens. Each entry averages 100 images and two rounds. Memory is the mean per-request peak PyTorch allocation, including model weights. Each compressor retains its own Dense reference.

Configuration	TTFT (ms)	Request (ms)	Peak (MiB)
VisionZip			
Dense	262.26	2250.44	16006.39
Native	194.39	2182.57	16225.79
Expanded	201.57	2189.22	16225.79
TokenGraft	219.81	2207.19	16225.79
PPE			
Dense	266.46	2251.66	16006.46
Native	199.20	2624.29	15924.66
Expanded	203.24	2629.07	15924.74
TokenGraft	225.23	2650.07	16225.79
Nuwa			
Dense	266.59	2248.76	16006.73
Native	285.62	2280.28	16293.94
Expanded	295.11	2292.44	16293.87
TokenGraft	304.45	2300.09	16294.08

For the paired intervals, we first average the two rounds for each image, then resample image clusters 10,000 times with seed 20260925. Overhead is $\mathrm{mean}(T_{\mathrm{TG}} - T_{\mathrm{Expanded}})/\,\mathrm{mean}(T_{\mathrm{Expanded}})$. Medians summarize all request-pair differences. Table 15 reports the request increments. For VisionZip, PPE, and Nuwa, respectively, TTFT increment intervals are $[18.11, 18.39]$, $[21.08, 22.94]$, and $[9.11, 9.57]$ ms; peak-memory increment intervals are $[0.00, 0.00]$, $[283.40, 319.98]$, and $[-0.17, 0.59]$ MiB.

Table 15: Paired full-request increments (TokenGraft minus Expanded) with fixed 48-token generation. Bootstrap intervals cluster by image after averaging its two rounds. Overhead is the relative increase in full-request time.

Compressor	Mean Δ (ms)	Median Δ (ms)	95% CI (ms)	Overhead (%)
VisionZip	17.98	18.14	$[+16.49, +19.31]$	0.82
PPE	21.00	21.82	$[+18.53, +23.30]$	0.80
Nuwa	7.65	8.94	$[+5.73, +9.32]$	0.33

D.3 Normal generation and component costs

Table 16 reports normal-generation request times, output lengths, and paired increments.

Table 16: Normal-generation check with greedy EOS stopping, a 48-token limit, and 100 image pairs per compressor. The paired request increment is TokenGraft minus Expanded. Overhead is the relative request-time increase; intervals are pointwise 95% paired intervals.

Configuration / metric	VisionZip	PPE	Nuwa
Request time (ms)			
Expanded	1939.58	2312.48	2022.64
TokenGraft	1956.95	2332.41	2032.45
Output length (tokens)			
Expanded	42.11	42.06	42.09
TokenGraft	42.10	42.04	42.13
TokenGraft − Expanded			
ΔRequest (ms)	17.37	19.92	9.80
95% CI (ms)	$[+13.08, +21.59]$	$[+6.77, +32.30]$	$[+6.08, +13.78]$
Overhead (%)	0.90	0.86	0.48

An independent 20-image profile keeps probes out of the main measurement. Adapter time includes prefill hooks and PPE CTM (selection, budgets, insertion, positions, and diagnostics), excluding backbone computation and Nuwa's shared upstream steps. Table 17's TG selector includes the loop and Appendix A.4 diagnostics.

PPE additionally reconstructs full visual attention for importance extraction in its TokenGraft path, increasing measured peak allocation by approximately 301 MiB. Its Native/Expanded paths omit this step; VisionZip and Nuwa obtain importance upstream.

Table 17: Independent 20-image adapter profile (ms). Adapter time includes selection, budgets, positions, and prefill diagnostics. The increment is paired TG minus Expanded; TG selector includes diagnostics.

Compressor	Adapter time (ms)		Paired increment (ms)		TG selector
	Expanded	TG	ΔAdapter	95% CI	(ms)
VisionZip	5.57	24.08	18.52	$[18.35, 18.73]$	11.74
PPE	3.14	26.32	23.19	$[21.82, 24.46]$	18.88
Nuwa	14.90	20.77	5.87	$[-0.13, 8.93]$	11.56