PACE: A Unified Condense-and-Extract Paradigm for Fast VLM Inference

Junjie Liu[1] **Shengyuan Ye**[2] **Xu Chen**[1,3]*
[1]Sun Yat-sen University
[2]Power Dispatch Control Center, Guangdong Power Grid Co., Ltd.
Guangzhou, China
[3]Shenzhen Loop Area Institute, Shenzhen, China

Abstract

Vision-Language Models (VLMs) demonstrate exceptional visual reasoning capabilities, yet their inference costs escalate rapidly with the proliferation of visual tokens. Existing visual token pruning methods exhibit two fundamental limitations. First, most approaches operate exclusively post-vision encoder, leaving the substantial latency of the visual encoding phase unoptimized. Second, under strict token budgets, these methods often fail to jointly preserve holistic visual contexts and fine-grained details, leading to performance degradation. To address these bottlenecks, we propose **PACE** (Pixel-Adaptive Condense and Extract), a training-free inference framework that accelerates both the vision encoder and the Large Language Model (LLM) via a unified *Condense-and-Extract* paradigm. During the *Condense* stage, an Adaptive Pixel Compressor (APC) evaluates visual information density prior to encoding, adaptively downsampling redundant inputs, curtailing encoder computation while preserving global context and essential visual cues. In the *Extract* stage, a Dynamic Dual-Attention Extractor (DDAE) selectively retains visual tokens via a fusion of internal visual signals from the encoder and semantic signals from the LLM, safeguarding task-critical details. By integrating PACE into Qwen2.5-VL-7B, the model retains 93.8% of its original performance while utilizing only 10% of the visual tokens, yielding a 3.1× speedup in time to first token (TTFT). Our code is available at https://github.com/jjL357/PACE.

1 Introduction

Vision-Language Models (VLMs) have established a new standard for multimodal understanding by seamlessly integrating visual perception with linguistic reasoning (Alayrac et al., 2022; Liu et al., 2023). Recent architectures have extended these capabilities to highly complex reasoning tasks (Dai et al., 2023; Team et al., 2023; Li et al., 2025). However, this rapid advancement demands an ever-expanding visual token budget. Processing high-resolution images (Achiam et al., 2023; Liu et al., 2024a) or extensive videos (Liu et al., 2025; Wang et al., 2025b) generates massive visual token sequences. Furthermore, modern VLMs equipped with native dynamic resolution capabilities (Bai et al., 2023; Guo et al., 2025; Abouelenin et al., 2025) partition inputs into numerous patches, frequently producing substantial visual redundancy. For instance, encoding a 4K image in Qwen2.5-VL generates over 42,000 patches for the Vision Transformer (ViT) (Dosovitskiy et al., 2020); even after spatial pooling, over 10,500 visual tokens extend the LLM context. Given the quadratic computational complexity of self-attention (Dao et al., 2022), this proliferation imposes a severe inference bottleneck, hindering real-time deployment.

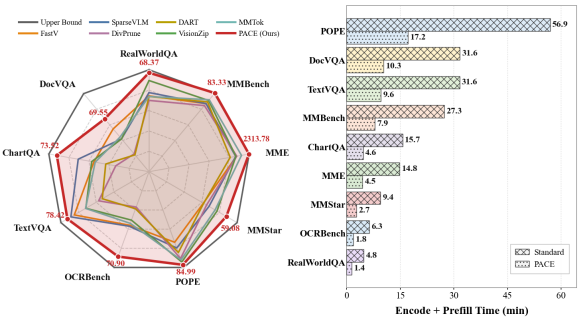

Figure 1: **Accuracy–TTFT trade-off at 10% visual-token retention.** PACE preserves accuracy while reducing pre-generation latency on Qwen2.5-VL-7B; standard denotes Vanilla inference of the original model.

Compressing the visual token sequence is therefore imperative for efficient inference. Visual token pruning naturally addresses this challenge by identifying and retaining critical visual tokens while discarding redundant ones prior to LLM decoding. Current pruning methods discard tokens based on

* Corresponding author: chenxu35@mail.sysu.edu.cn

Findings of the Association for Computational Linguistics: EMNLP 2026, pages 24111–24132
October 24-29, 2026 ©2026 Association for Computational Linguistics

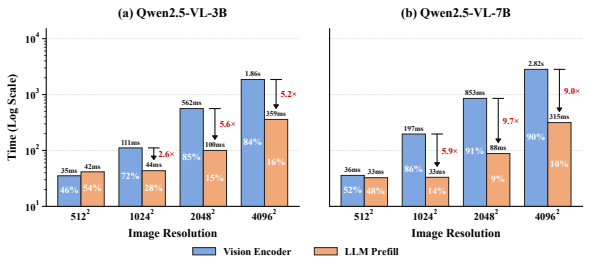

Figure 2: **Latency decomposition for Qwen2.5-VL.** Vision encoding and LLM prefill jointly dominate latency as input resolution increases.

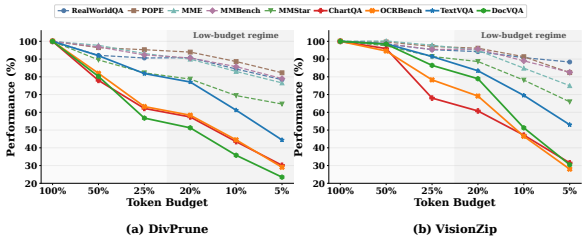

Figure 3: **Performance of existing visual token pruning methods under varying token budgets on Qwen2.5-VL-7B.** Performance on detail-sensitive benchmarks degrades sharply under aggressive token pruning.

attention distributions (Chen et al., 2024a; Takezoe et al., 2026), token similarity (Wen et al., 2025; Zou et al., 2026), diversity (Zhang et al., 2026; Fang et al., 2026), or maximum coverage (Dong et al., 2025; Deng et al., 2026). While these strategies effectively alleviate LLM-side computational demands, they share two fundamental limitations.

The Dual Bottleneck of High-Resolution Inference. Prevailing visual token pruning methods operate exclusively *after* the vision encoder. They truncate the LLM context sequence but neglect the substantial computational overhead of the vision encoder itself. Empirical profiling reveals that at high resolutions, both the ViT encoding and LLM prefill stages impose severe latency bottlenecks, as illustrated in Figure 2. Because most visual token pruning methods intervene strictly during modality alignment or within the LLM layers, the immense computational cost of encoding high-resolution pixels remains unaddressed. Consequently, truncating only the LLM-side sequence resolves merely half of the inference bottleneck.

Information and Detail Loss under Low-Budget Compression. Furthermore, under strict token budgets, existing methods struggle to retain holistic visual contexts and fine-grained details simultaneously. Aggressive pruning inherently fragments visual layouts and discards indispensable visual cues, such as text strokes and alignment anchors. This structural loss results in severe performance degradation, particularly on detail-sensitive tasks. While competitive methods, including DivPrune and VisionZip, maintain robust accuracy on general benchmarks at a 5% token budget, they experience severe performance drops on detail-sensitive tasks such as ChartQA and DocVQA (Figure 3). This vulnerability indicates that simple post-encoder token pruning fails to preserve the critical cues essential for high-density visual reasoning.

These limitations underscore the necessity for a paradigm shift: visual token compression must transcend naive post-encoder sequence reduction by simultaneously preserving holistic layouts and salient visual details *prior* to intensive computation. An optimal framework must fulfill two complementary functions. First, it should *condense* visual information before the expensive encoding phase. Recent studies (Ye et al., 2025; Cai et al., 2025) suggest that condensed visual representations inherently carry concentrated semantic meaning. Pre-encoder condensation ensures the ViT operates on a compact pixel budget without sacrificing layout integrity. Second, it should *extract* the most informative visual tokens post-encoding, guaranteeing that critical, task-relevant details are prioritized within the LLM context.

To this end, we propose **PACE**, a training-free, plug-and-play inference framework organized around a unified **Condense-and-Extract** paradigm. In the **Condense** stage, the Adaptive Pixel Compressor (APC) uses a lightweight feature preview to estimate information density. It dynamically condenses the input image prior to the vision encoder, globally preserving visual cues under strict pixel budgets. In the **Extract** stage, the Dynamic Dual-Attention Extractor (DDAE) fuses ViT self-attention with LLM cross-modal attention. This dynamic dual-attention mechanism enables PACE to extract fine-grained details reliably, markedly outperforming single-source attention pruning. Figure 1 summarizes the resulting accuracy–TTFT trade-off.

Our contributions are summarized as follows:

- **Identification of the Dual Bottleneck.** We demonstrate that conventional visual token pruning overlooks the substantial vision-encoder overhead and incurs severe degrada-

tion on detail-sensitive tasks due to its inability to jointly preserve holistic contexts and fine-grained details.

- **Unified Condense-and-Extract Framework.** We introduce PACE, seamlessly integrating pre-encoder adaptive pixel condensation (APC) with post-encoder dynamic dual-attention extraction (DDAE). APC constructs a compact visual representation, while DDAE safeguards salient visual tokens via confidence-weighted attention fusion.

- **Superior Performance–Efficiency Trade-off.** Extensive evaluations demonstrate that PACE consistently outperforms existing pruning baselines. When deployed on Qwen2.5-VL-7B, PACE preserves over 93% of the original performance while discarding 90% of the visual tokens, unlocking a $3.1\times$ TTFT acceleration.

2 Related Work

2.1 Vision-Language Models

Early VLMs typically rely on fixed-resolution vision encoders, requiring images to be resized or padded before visual encoding (Liu et al., 2023). This design simplifies model processing but may lose fine-grained details. To improve high-resolution perception, models such as InternVL (Chen et al., 2024c) adopt dynamic high-resolution tiling, where images are split into multiple local crops according to their aspect ratio and resolution. More recent models, including Qwen2.5-VL (Bai et al., 2025b) and Qwen3-VL (Bai et al., 2025a), support native dynamic-resolution processing, preserving image aspect ratios and producing variable-length visual token sequences. However, as the visual token length still grows with the input pixel budget, high-resolution inference remains computationally expensive (Dao, 2024).

2.2 Visual Token Pruning

Visual token pruning reduces inference cost by selecting or merging a subset of visual tokens before they enter the LLM. Attention-based methods, such as FastV (Chen et al., 2024a) and SparseVLM (Zhang et al., 2024), estimate token importance from attention or vision–language relevance. Hybrid methods, including LLaVA-PruMerge (Shang et al., 2025) and VisionZip (Yang et al., 2025), combine importance-based selection with similarity-based merging. Other methods exploit redundancy, diversity, or coverage, such as DART (Wen et al., 2025), DivPrune (Alvar et al., 2025), and MMTok (Dong et al., 2025). Although these approaches effectively reduce LLM-side prefill cost, they are mostly applied after visual encoding. Thus, the ViT encoding cost remains unchanged.

2.3 Adaptive Resolution VLMs

Adaptive-resolution VLMs adjust image scale, compression rate, or token allocation according to visual complexity. Existing methods often require additional modules, learned routing, or reinforcement-learning procedures. For example, ViCO (Cui et al., 2025) and HyperVL (Team et al., 2025) introduce learned compression or routing mechanisms, while AdaptVision (Lin et al., 2025) and VisionThink (Yang et al., 2026) formulate adaptive resolution as coarse-to-fine visual acquisition. In contrast, PACE is training-free and requires no architectural modification. It reduces both pre-encoder and post-encoder costs by adapting input resolution before visual encoding and controlling the visual tokens passed to the LLM, while preserving global layout and fine-grained visual evidence.

3 Method

3.1 Overview of PACE

As illustrated in Figure 4, PACE introduces a unified *Condense-and-Extract* pipeline designed to overcome the dual computational bottlenecks of the ViT and LLM, as well as the severe detail loss typically induced by low-budget token pruning. In the **Condense** stage, PACE evaluates visual information density prior to the ViT forward pass. Using a shallow feature preview to quantify detail complexity, PACE adaptively condenses the input image. This mechanism strictly curtails the pixel volume processed by the vision encoder while maintaining essential layout cues. In the **Extract** stage, PACE eliminates residual visual redundancy post-encoding. It dynamically integrates cross-attention from the LLM with self-attention from the ViT, ensuring that token selection prioritizes both semantic relevance and fine-grained detail integrity.

3.2 Condense: Adaptive Pixel Compressor

While heuristically dropping uninformative patches (such as uniform backgrounds or regions lacking salient information) (Wang et al., 2026b; Choi et al., 2026) intuitively reduces input length, it irrevoca-

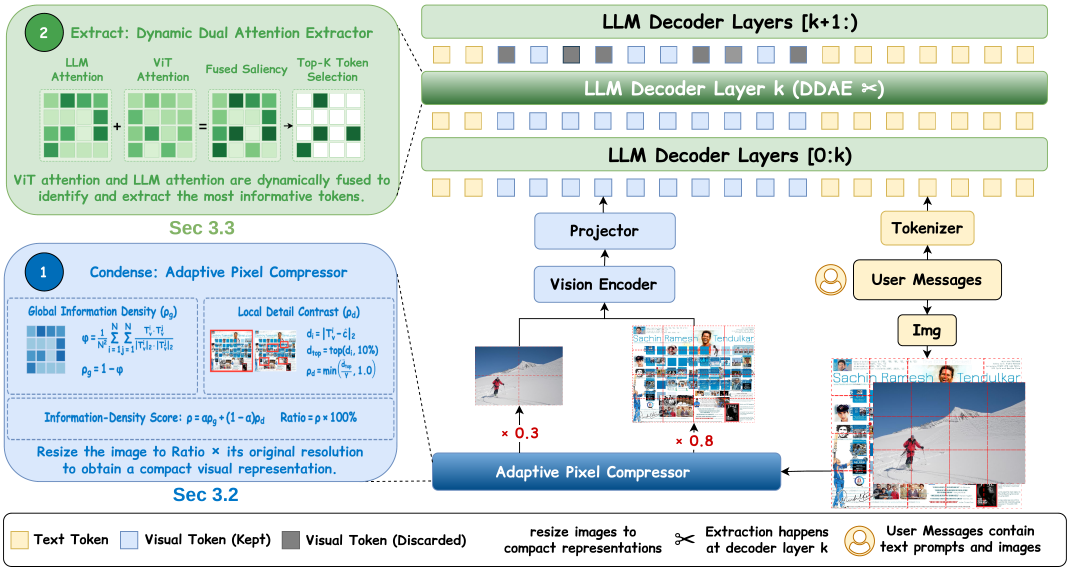

Figure 4: **Overview of PACE's unified Condense-and-Extract pipeline.** In the Condense stage, APC uses a shallow visual preview to estimate global redundancy and local detail, then adaptively resizes the input before the vision encoder. In the Extract stage, DDAE fuses semantic attention from the LLM with self-attention from the ViT and retains the top-K visual tokens at decoder layer k. Together, APC lowers vision-encoder cost and DDAE shortens LLM prefill, enabling efficient low-budget inference while preserving holistic layouts and fine-grained task evidence.

bly disrupts the continuous 2D topological layout required by modern dynamic-resolution ViTs, risking the deletion of latent contextual anchors. Consequently, although heuristic dropping may suffice for narrow, domain-specific applications, it generalizes poorly to diverse, open-world visual scenarios. Conversely, static uniform downsampling blurs microscopic elements, such as text strokes, while continuing to allocate wasteful computation to homogeneous backgrounds.

To address this dilemma, we introduce the Adaptive Pixel Compressor (APC) in the **Condense** stage. Instead of pruning raw patches, APC dynamically modulates the global input resolution, allocating higher pixel budgets to information-dense inputs and lower budgets to redundant ones. This approach preserves continuous visual layouts while dynamically adapting to image complexity.

Shallow Feature Preview. Instead of relying on basic pixel-level statistics, APC uses the initial ViT block ($K = 1$) to compute a lightweight feature preview, similar to AdaPatch (Liu et al., 2026). This minimizes preprocessing overhead while securing essential semantic priors. The preview yields token embeddings, which are subsequently ℓ_2-normalized and denoted as $\mathbf{T}_v = \{\mathbf{T}_v^i\}_{i=1}^N \in \mathbb{R}^{N\times D}$. A controlled ablation in Appendix B.2 confirms that this semantic preview is more reliable than RGB, entropy, edge-density, and Laplacian statistics.

Global Information Density (ρ_g). We quantify global redundancy by calculating the average pairwise cosine similarity φ across all visual tokens:

$$\varphi = \frac{1}{N^2}\sum_{i=1}^{N}\sum_{j=1}^{N}\frac{\mathbf{T}_v^i \cdot \mathbf{T}_v^j}{\|\mathbf{T}_v^i\|_2 \cdot \|\mathbf{T}_v^j\|_2}. \tag{1}$$

A higher φ indicates pronounced redundancy (e.g., large uniform areas) and lower global information density. The global information density score ρ_g is defined as the non-redundant fraction:

$$\rho_g = 1.0 - \varphi \tag{2}$$

Local Detail Contrast (ρ_d). Images dominated by uniform backgrounds may yield high global redundancy metrics while harboring sparse yet indispensable details, such as microscopic text on a large white document. To prevent the erasure of such cues, APC computes a global background baseline $\mathbf{c}$ by averaging all normalized tokens:

$$\mathbf{c} = \frac{1}{N}\sum_{i=1}^{N}\mathbf{T}_v^i. \tag{3}$$

This baseline is normalized as $\hat{\mathbf{c}} = \mathbf{c}/\|\mathbf{c}\|_2$. We then compute the Euclidean distance d_i between each token and the reference baseline:

$$d_i = \left\|\mathbf{T}_v^i - \hat{\mathbf{c}}\right\|_2. \tag{4}$$

Tokens diverging significantly from the baseline correlate strongly with sharp local details. APC isolates the top 10% of tokens with the largest d_i and averages their distances to compute $\bar{d}_{top}$. This tail mean avoids diluting sparse details through a full-image average while being less noise-sensitive than a single maximum; 5%–20% tail choices behave similarly (Appendix B.1). This metric is linearly scaled into a local retention score:

$$\rho_d = \min\left(\frac{\bar{d}_{top}}{\gamma}, 1.0\right), \tag{5}$$

where γ acts as a regulating scaling factor. A high ρ_d signals sharp local contrast, mandating a higher target resolution to safeguard fine-grained visual features.

Adaptive Condensation. Finally, APC integrates the global and local scores into a unified target retention ratio ρ:

$$\rho = \alpha\rho_g + (1-\alpha)\rho_d, \tag{6}$$

where α is a weighting hyperparameter. The target retention ratio is defined as $r = \rho$, and the raw input image is resized accordingly. Importantly, to ensure strict adherence to system memory constraints, if the allocated token budget specifies a maximum scaling ratio lower than ρ, the input is directly resized to satisfy the hard budget. Operating as an independent pre-encoder module, APC seamlessly integrates into other visual token pruning frameworks, as detailed in Section 5.1.

3.3 Extract: Dynamic Dual-Attention Extractor

Although APC reduces the pre-encoder sequence length, encoded features may still harbor latent visual redundancy. The **Extract** stage removes this residual redundancy prior to the LLM prefill phase, isolating tokens that are both semantically relevant and informatively rich.

Conventional post-encoder extraction mechanisms depend heavily on LLM cross-attention. While this mapping accurately isolates prompt-related semantics, it can focus too narrowly on explicitly referenced regions: explicit textual references receive high attention, whereas critical visual anchors, such as chart grid lines, receive negligible weights and are often erroneously pruned. Conversely, ViT self-attention reliably demarcates visual boundaries but operates unconditioned on the query, frequently retaining task-irrelevant background clutter. Relying on either signal in isolation fails to capture both salient targets and fine-grained supporting details.

To resolve this single-modality bias, we introduce the **Dynamic Dual-Attention Extractor (DDAE)**, which adaptively fuses linguistic semantic signals and visual signals via confidence-weighted attention integration.

Initially, DDAE extracts semantic attention scores from a designated LLM layer (denoted as extraction depth L_{ext}) to formulate a semantic relevance map S_{llm}, min-max normalized to $[0, 1]$. Concurrently, internal self-attention maps are fetched from the terminal layers of the vision encoder to construct a visual density map S_{vis}, similarly normalized to $[0, 1]$.

DDAE employs the standard deviations, σ_{llm} and σ_{vis}, of these normalized distributions as unsupervised confidence proxies. A higher standard deviation indicates a sharper distribution, signifying elevated confidence toward a concise set of salient regions. A softmax function dynamically assigns the fusion weights α_{weight} and β_{weight}:

$$\alpha_{weight}, \beta_{weight} = \text{Softmax}\left(\frac{[\sigma_{llm}, \sigma_{vis}]}{\tau}\right) \tag{7}$$

where τ serves as a tunable temperature parameter. The final token saliency score is aggregated as $S_{final} = \alpha_{weight}S_{llm} + \beta_{weight}S_{vis}$. DDAE ranks the visual tokens based on S_{final} and preserves the top-K tokens to formulate the final LLM context sequence. Detailed theoretical analysis regarding computational complexity reduction is provided in Appendix C.

4 Experiments

4.1 Evaluation Setup

Models and Framework. We evaluate PACE on the Qwen2.5-VL architectures (3B and 7B variants) (Bai et al., 2025b). To ensure standardized and reproducible evaluations, we conduct all benchmark experiments using the open-source `lmms-eval` framework (Zhang et al., 2025). To

Method	RealWorldQA Acc. ↑	POPE F1 ↑	MME P+C ↑	MMBench Acc. ↑	MMStar Acc. ↑	ChartQA Acc. ↑	OCRBench Acc. ↑	TextVQA Acc. ↑	DocVQA ANLS ↑	Avg. ↑
Fixed-resolution setting (MinPix = 2048 × 28 × 28, MaxPix = 2048 × 28 × 28)										
Vanilla (100%)	69.54	86.36	2317	82.99	63.91	78.20	77.30	82.37	94.74	100.0%
*Retain 20% $\bar{T}$ (↓ **80% Tokens**)*										
FastV (ECCV'24)	64.58	80.99	2256	80.76	54.79	64.12	66.60	79.16	76.58	90.2%
SparseVLM (ICML'25)	66.41	83.23	2258	81.36	55.65	70.00	56.54	80.33	73.56	90.3%
DivPrune (CVPR'25)	61.83	84.07	2248	80.07	54.66	51.56	51.50	69.99	49.40	81.7%
DART (EMNLP'25)	63.53	82.81	2278	79.64	55.83	58.72	54.00	69.48	50.58	83.5%
VisionZip (CVPR'25)	**67.06**	85.50	2317	81.01	58.93	69.68	64.60	77.52	75.77	92.5%
MMTok (ICLR'26)	63.79	84.81	2278	82.22	58.17	68.40	65.80	76.78	74.25	91.4%
PACE (w/o APC)	66.93	85.29	2310	81.36	59.29	72.84	69.10	80.72	83.00	94.9%
PACE (Ours)	66.93	**86.23**	**2322**	**83.08**	**62.47**	**78.56**	**79.00**	**81.66**	**86.84**	**98.6%**
*Retain 10% $\bar{T}$ (↓ **90% Tokens**)*										
FastV (ECCV'24)	59.08	73.28	2154	77.23	48.78	51.80	52.40	74.23	59.56	79.9%
SparseVLM (ICML'25)	60.39	76.25	2145	77.23	51.30	61.00	53.00	76.30	49.61	81.4%
DivPrune (CVPR'25)	57.25	81.73	2158	76.12	50.22	39.00	41.70	59.32	34.58	72.5%
DART (EMNLP'25)	58.82	78.60	2074	78.26	48.89	44.88	42.70	57.12	33.65	72.6%
VisionZip (CVPR'25)	65.23	83.55	2147	79.04	54.90	53.08	49.40	67.16	49.55	81.1%
MMTok (ICLR'26)	58.82	82.44	2218	79.47	54.04	51.04	51.30	67.33	51.37	80.4%
PACE (w/o APC)	65.36	82.27	2228	80.07	55.25	65.76	58.00	76.18	65.54	87.7%
PACE (Ours)	**68.37**	**84.99**	**2314**	**83.33**	**59.08**	**73.52**	**70.90**	**78.42**	**69.55**	**93.8%**
*Retain 5% $\bar{T}$ (↓ **95% Tokens**)*										
FastV (ECCV'24)	56.08	62.45	1995	73.54	45.88	36.52	41.40	66.21	44.17	69.6%
SparseVLM (ICML'25)	56.60	65.25	2011	72.08	45.01	46.24	43.10	69.53	30.99	70.3%
DivPrune (CVPR'25)	53.46	77.66	1926	72.77	45.20	28.96	29.80	41.28	22.64	62.0%
DART (EMNLP'25)	52.68	71.33	1946	73.97	42.80	30.76	32.90	45.01	23.37	62.2%
VisionZip (CVPR'25)	61.57	78.65	2029	75.17	48.63	40.96	38.30	55.47	29.40	70.5%
MMTok (ICLR'26)	54.12	78.70	2133	75.17	47.00	32.76	36.00	50.95	28.85	67.3%
PACE (w/o APC)	60.39	76.59	2117	75.95	49.26	53.00	46.10	67.32	45.38	76.9%
PACE (Ours)	**63.79**	**80.70**	**2245**	**80.07**	**56.02**	**61.84**	**57.80**	**71.56**	**48.94**	**84.3%**

Table 1: **Performance comparison on Qwen2.5-VL-7B under the fixed-resolution setting.** Benchmark scores cover nine tasks at 20%, 10%, and 5% visual-token retention. PACE delivers the strongest overall result at every evaluated budget, with particularly robust performance on detail-sensitive benchmarks such as OCRBench and DocVQA.

assess cross-model generalization, we further evaluate PACE on InternVL3.5-4B (Wang et al., 2025a), whose multi-tile pipeline differs from Qwen's native dynamic-resolution design; the results are reported in Appendix B.3.

Baselines. We benchmark PACE against prominent visual token pruning methods: **FastV** (Chen et al., 2024a), **SparseVLM** (Zhang et al., 2024), **DivPrune** (Alvar et al., 2025), **DART** (Wen et al., 2025), **VisionZip** (Yang et al., 2025), and **MMTok** (Dong et al., 2025).

Implementation Details. For the APC module, we set the preview depth $K = 1$, fusion weight $\alpha = 0.6$, and local contrast scaling $\gamma = 1.5$. For DDAE, semantic maps are extracted from the second LLM layer ($L_{ext} = 2$) with a fusion temperature $\tau = 0.5$. Evaluations encompass two configurations: the **fixed-resolution setting** and the **dynamic-resolution setting**, with the former serving as the default unless otherwise specified. Detailed configurations are provided in Appendix A.

Benchmarks. Our evaluation encompasses nine distinct datasets: **MME** (Fu et al., 2026), **POPE** (Li et al., 2023), **MMBench** (Liu et al., 2024b), **MMStar** (Chen et al., 2024b), **RealWorldQA** (X.AI, 2024), **TextVQA** (Singh et al., 2019), **DocVQA** (Mathew et al., 2021), **ChartQA** (Masry et al., 2022), and **OCRBench** (Liu et al., 2024c).

4.2 Main Results

As summarized in Table 1, we report Qwen2.5-VL-7B performance under the strict fixed-resolution setting. Additional results for both the 3B and 7B variants are provided in Appendix B.

PACE consistently surpasses baselines across all configurations. Conventional post-encoder pruning methods suffer marked degradation on detail-sensitive tasks, notably DocVQA and OCRBench, particularly under aggressive 5% or 10% constraints. This vulnerability stems from the irreversible loss of high-frequency visual anchors—a direct consequence of relying predominantly on late-stage LLM relevance signals.

Conversely, the APC module condenses the visual representation while preserving text-dense and

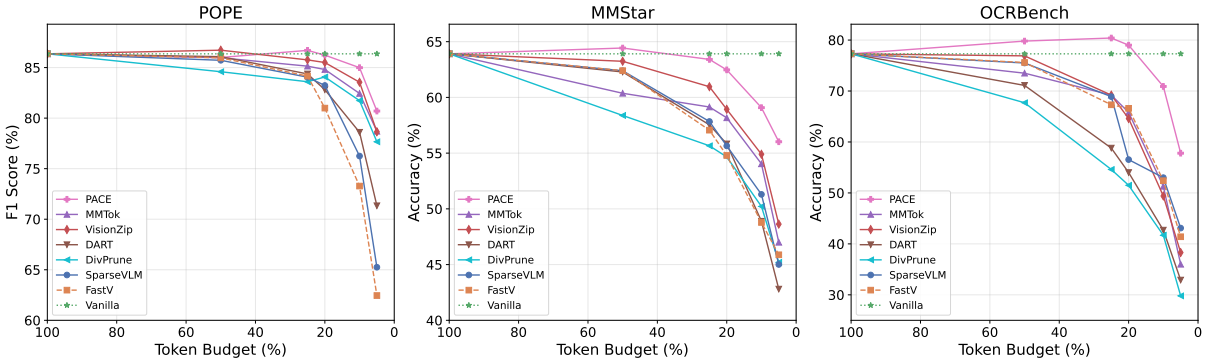

Figure 5: **Performance scaling under continuous token budgets.** PACE degrades more gracefully than pruning baselines as retention decreases, with especially large advantages on the detail-sensitive MMStar and OCRBench benchmarks under strict budgets.

detail-sensitive regions prior to full encoding, enabling DDAE to precisely extract semantically relevant details. Under a highly restrictive 10% token budget, PACE maintains 93.8% of the full model's performance, yielding average performance gains of 12.7 and 13.9 percentage points over VisionZip and FastV, respectively. DocVQA still trails Vanilla because exact answers can depend on microscopic characters and layout relations that become ambiguous after condensation; nevertheless, PACE reaches 69.55, versus 59.56 for FastV and 49.55 for VisionZip. Small gains over Vanilla on a few tasks may result from removing distractors and are not our central claim.

Cross-backbone experiments in Appendix B.3 further evaluate InternVL3.5-4B's multi-tile pipeline. Across 25%, 20%, and 10% retention, PACE attains normalized averages of 85.2%, 80.9%, and 69.5%, exceeding the strongest corresponding baseline by 5.1, 4.7, and 4.0 points.

Robustness Validation. As illustrated in Figure 5, we evaluate the algorithmic robustness of PACE on Qwen2.5-VL-7B across continuous token budgets ranging from 100% down to 5%. We track performance on POPE, MMStar, and OCRBench, which assess object-level perception, visually grounded multimodal reasoning, and fine-grained OCR-centric recognition, respectively.

PACE exhibits robust performance scaling as the retention budget decreases. At the 10% token budget, PACE achieves 84.99% on POPE, 59.08% on MMStar, and 70.90% on OCRBench, outperforming the strongest baseline by 1.44, 4.18, and 17.90 absolute points. This advantage remains pronounced under the more aggressive 5% budget, where PACE exceeds the best baseline by 2.00, 7.39, and 14.70 points across the three tasks. These results confirm that PACE successfully preserves both object-level perception and fine-grained visual evidence under severe compression.

4.3 Efficiency Profiling

We profile the inference efficiency of PACE on Qwen2.5-VL-7B under the fixed-resolution setting with a 10% visual-token budget on a single RTX 4090. Since PACE does not accelerate autoregressive decoding, we report **time to first token (TTFT)** rather than end-to-end generation time. Vanilla TTFT comprises vision encoding and LLM prefill; PACE TTFT additionally includes the Shallow Feature Preview and adaptive resizing. The stage-wise encoder and prefill entries exclude these APC overheads.

PACE reduces the average vision encoder latency from 148.84 ms to 49.47 ms (3.01× speedup) and the LLM prefill latency from 217.05 ms to 32.69 ms (6.64× speedup). Although APC introduces an average 34.62 ms preview-and-resizing overhead, TTFT still decreases from 365.89 ms to 116.79 ms, a 3.13× speedup. Across the full budget sweep in Appendix B.5, average TTFT speedup rises from 1.10× at 80% retention to 1.67×, 2.64×, and 3.13× at 50%, 20%, and 10%, respectively.

5 Analysis and Discussion

In this section, we analyze the core mechanisms of PACE. Additional evaluations—including hyperparameter robustness and ablations on attention-token sources—are detailed in Appendix B.

5.1 Orthogonality of APC

As a pre-encoder module, APC integrates orthogonally with existing post-encoder pruning algo-

Dataset	Encoder Time (ms)		Prefill Time (ms)		TTFT incl. APC (ms)	
	Vanilla / PACE	Spd.	Vanilla / PACE	Spd.	Vanilla / PACE	Spd.
DocVQA	143.41 / 50.74	2.83×	210.41 / 32.70	6.43×	353.82 / 117.54	3.01×
TextVQA	154.28 / 48.20	3.20×	223.68 / 32.69	6.84×	377.96 / 116.03	3.26×
Avg.	148.84 / 49.47	**3.01×**	217.05 / 32.69	**6.64×**	365.89 / 116.79	**3.13×**

Table 2: **Latency of PACE on Qwen2.5-VL-7B at 10% retention.** Encoder and prefill columns report their isolated stages; TTFT includes APC preview and resizing overhead. Measurements use the fixed-resolution setting on one RTX 4090.

rithms. We validate this by augmenting VisionZip and MMTok with APC under 10% and 5% token-retention budgets.

As shown in Table 3, APC improves both baselines, particularly on detail-sensitive benchmarks. Under a 10% budget, APC boosts both VisionZip and MMTok by more than 15 points on ChartQA and OCRBench, alongside a gain of roughly 10 points on DocVQA. These improvements support APC's compatibility with different downstream extractors and its ability to preserve fine-grained visual evidence before encoding.

Method	Budget	RealWorldQA	MME	ChartQA	OCRBench	TextVQA	DocVQA
VisionZip	10%	65.23	2147	53.08	49.40	67.16	49.55
+ APC	10%	**67.32** (+2.09)	**2323** (+176)	**68.96** (+15.88)	**67.50** (+18.10)	**72.39** (+5.23)	**60.45** (+10.90)
MMTok	10%	58.82	2218	51.04	51.30	67.33	51.37
+ APC	10%	**63.27** (+4.45)	**2289** (+71)	**66.84** (+15.80)	**66.80** (+15.50)	**72.56** (+5.23)	**62.05** (+10.68)
VisionZip	5%	61.57	2029	40.96	38.30	55.47	29.40
+ APC	5%	**64.18** (+2.61)	**2206** (+177)	**48.88** (+7.92)	**50.10** (+11.80)	**58.79** (+3.32)	**37.44** (+8.04)
MMTok	5%	54.12	2133	32.76	36.00	50.95	28.85
+ APC	5%	**59.48** (+5.36)	**2221** (+88)	**48.56** (+15.80)	**50.70** (+14.70)	**60.40** (+9.45)	**40.64** (+11.79)

Table 3: **Orthogonal integration of APC.** APC augments VisionZip and MMTok at 10% and 5% token budgets.

5.2 Adaptive Resolution vs. Static Resolution

We evaluate the efficacy of APC's adaptive pixel allocation against static-resolution variants under the dynamic-resolution setting of Qwen2.5-VL-7B. While static-resolution methods apply rigid condensation ratios globally, APC dynamically modulates the input resolution based on global density and local detail metrics.

As illustrated in Figure 6, Adaptive Resolution consistently achieves a favorable performance balance across diverse domains. On holistic tasks such as RealWorldQA, it outperforms the best static-resolution counterpart at both 10% and 5% budgets. Conversely, on detail-sensitive tasks such as ChartQA, Fixed Res.-50 yields a marginal gain over adaptive resolution (0.36 points at 10%) but concurrently induces a significant drop (2.74 points) on RealWorldQA. This trade-off confirms that rigid global scaling inherently overfits specific domains at the expense of general visual robustness, whereas APC successfully balances broad layout contexts with microscopic visual evidence.

5.3 Effect of Attention Fusion in DDAE

We evaluate DDAE's dynamic fusion against single-modality and fixed-weight baselines: **DDAE**, **Only ViT-Attn** (solely vision-side visual attention), **Only LLM-Attn** (solely LLM-side semantic attention), and **Fixed Fusion** (a static combination that assigns a weight of 0.5 to each signal).

As shown in Figure 7, Dynamic DDAE exhibits the most resilient performance profile. Relying exclusively on LLM attention incurs substantial drops on visual tasks, with ChartQA degrading by over 10 points, indicating that text-guided signals neglect unprompted yet vital visual contexts. Conversely, omitting text-guided semantics (Only ViT-Attn) compromises task alignment. These representative gaps affirm that robust feature extraction necessitates dynamic, confidence-weighted multimodal attention.

5.4 Impact of DDAE Extraction Depth

The LLM extraction depth (L_{ext}) governs the trade-off between prefill latency and visual reasoning. A shallower layer initiates earlier pruning, significantly reducing prefill latency, whereas a deeper layer yields more sophisticated cross-modal attention maps at the cost of processing the full visual sequence longer. Table 4 illustrates this trade-off under a 10% token budget.

Deferring DDAE extraction to deeper layers (e.g., depth 24) enhances visual reasoning, markedly boosting ChartQA performance over depth 8. However, under the $2048 \times 28 \times 28$ fixed-resolution setting, the prefill speedup falls from $2.11\times$ at layer 2 to only $1.09\times$ at layer 24. The same trend holds at twice the resolution (Appendix B.7). We therefore use layer 2 ($L_{ext} = 2$) as the efficiency-oriented default.

Depth (L_{ext})	RealWorldQA	POPE	ChartQA	TextVQA	Prefill (ms)	Spd.
2	**68.37**	84.99	73.52	78.42	**35.26**	**2.11×**
8	66.54	84.72	65.68	78.80	42.51	1.75×
16	66.93	85.31	70.60	**79.24**	55.32	1.35×
24	67.71	**86.32**	**78.96**	78.41	68.12	1.09×

Table 4: **Quality–latency trade-off of DDAE extraction depth.** Prefill latency is compared against the 74.52 ms PACE-without-DDAE baseline.

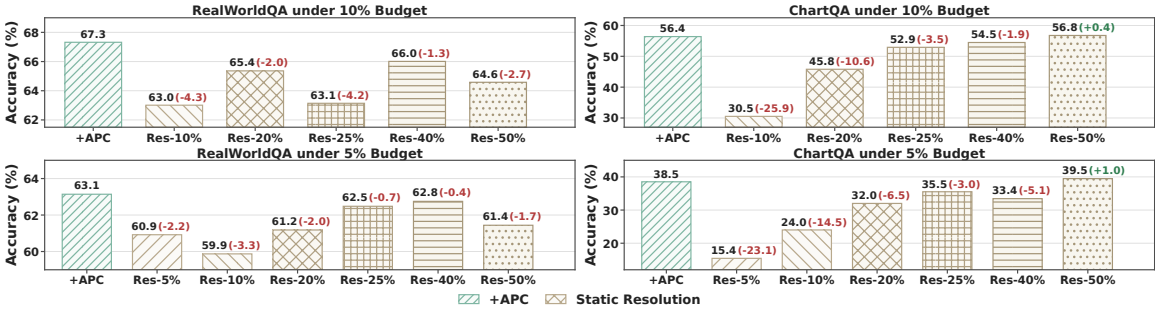

Figure 6: **Adaptive vs. static resolution under the dynamic-resolution setting.** Adaptive pixel allocation is compared with static-resolution baselines at 10% and 5% token budgets.

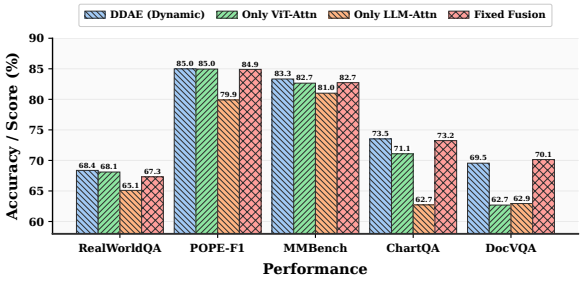

Figure 7: **Effect of attention fusion in DDAE.** Dynamic fusion yields the most balanced performance across benchmarks: language-only attention loses critical visual evidence, while vision-only attention weakens task alignment.

6 Conclusion

This paper introduces **PACE**, a training-free inference framework designed to accelerate high-resolution VLMs via a unified **Condense-and-Extract** paradigm. To overcome the dual computational bottlenecks of the vision encoder and the LLM, the *Condense* phase uses APC to adaptively remove pixel-level redundancy prior to visual encoding, mitigating compute-bound ViT overhead while preserving global layouts. Subsequently, the *Extract* phase employs DDAE to retain salient fine-grained details by dynamically fusing internal visual priors from the ViT with semantic relevance from the LLM. PACE retains 93.8% of Qwen2.5-VL-7B's uncompressed performance at a 90% token reduction and delivers a 3.1× TTFT speedup.

7 Limitations

PACE has two important limitations. First, APC accelerates the encoder only when reducing the pixel or tile budget also reduces the number of encoder tokens. On fixed-grid VLMs, DDAE can still reduce LLM prefill cost, but APC provides no encoder-side gain. Preview overhead also varies with architecture and resolution, so the Qwen2.5-VL speedups may not transfer unchanged to other backbones.

Second, APC performs query-agnostic, one-shot condensation and can miss faint or tiny characters, small chart labels, thin lines, or low-contrast objects. Once resizing makes such evidence ambiguous, DDAE cannot reconstruct it. High-stakes applications should therefore use a higher retention floor, a less aggressive target budget, or a lower α to emphasize local detail. Uncertainty-triggered or query-conditioned recovery of high-resolution crops is a promising extension.

Acknowledgments

This work was supported in part by the Guangdong S&T Programme (Grant No. 2024B0101040007); the National Natural Science Foundation of China (Grant No. U25B2002); the Guangdong Basic and Applied Basic Research Foundation (Grant No. 2023B1515120058); and the Program for Guangdong Introducing Innovative and Entrepreneurial Teams (Grant No. 2017ZT07X355).

References

Abdelrahman Abouelenin, Atabak Ashfaq, Adam Atkinson, Hany Awadalla, Nguyen Bach, Jianmin Bao, Alon Benhaim, Martin Cai, Vishrav Chaudhary, Congcong Chen, and 1 others. 2025. Phi-4-mini technical report: Compact yet powerful multimodal language models via mixture-of-loras. *arXiv preprint arXiv:2503.01743*.

Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, and 1 others. 2023. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*.

Jean-Baptiste Alayrac, Jeff Donahue, Pauline Luc, Antoine Miech, Iain Barr, Yana Hasson, Karel Lenc, Arthur Mensch, Katherine Millican, Malcolm Reynolds, and 1 others. 2022. Flamingo: a visual language model for few-shot learning. *Advances in neural information processing systems*, 35:23716–23736.

Saeed Ranjbar Alvar, Gursimran Singh, Mohammad Akbari, and Yong Zhang. 2025. Divprune: Diversity-based visual token pruning for large multimodal models. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pages 9392–9401.

Jinze Bai, Shuai Bai, Shusheng Yang, Shijie Wang, Sinan Tan, Peng Wang, Junyang Lin, Chang Zhou, and Jingren Zhou. 2023. Qwen-vl: A frontier large vision-language model with versatile abilities. *arXiv preprint arXiv:2308.12966*, 1(2):3.

Shuai Bai, Yuxuan Cai, Ruizhe Chen, Keqin Chen, Xionghui Chen, Zesen Cheng, Lianghao Deng, Wei Ding, Chang Gao, Chunjiang Ge, and 1 others. 2025a. Qwen3-vl technical report. *arXiv preprint arXiv:2511.21631*.

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, and 8 others. 2025b. Qwen2.5-vl technical report. *Preprint*, arXiv:2502.13923.

Mu Cai, Jianwei Yang, Jianfeng Gao, and Yong Jae Lee. 2025. Matryoshka multimodal models. In *International Conference on Learning Representations*, volume 2025, pages 46254–46272.

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

Lin Chen, Jinsong Li, Xiaoyi Dong, Pan Zhang, Yuhang Zang, Zehui Chen, Haodong Duan, Jiaqi Wang, Yu Qiao, Dahua Lin, and 1 others. 2024b. Are we on the right way for evaluating large vision-language models? *Advances in Neural Information Processing Systems*, 37:27056–27087.

Zhe Chen, Jiannan Wu, Wenhai Wang, Weijie Su, Guo Chen, Sen Xing, Muyan Zhong, Qinglong Zhang, Xizhou Zhu, Lewei Lu, and 1 others. 2024c. Internvl: Scaling up vision foundation models and aligning for generic visual-linguistic tasks. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 24185–24198.

Joonmyung Choi, Sanghyeok Lee, Jongha Kim, Sehyung Kim, Dohwan Ko, Jihyung Kil, and Hyunwoo J Kim. 2026. Docprune: Efficient document question answering via background, question, and comprehension-aware token pruning. *arXiv preprint arXiv:2604.22281*.

Rohan Choudhury, JungEun Kim, Jinhyung Park, Eunho Yang, László A Jeni, and Kris M Kitani. 2025. Accelerating vision transformers with adaptive patch sizes. *arXiv preprint arXiv:2510.18091*.

Long Cui, Weiyun Wang, Jie Shao, Zichen Wen, Gen Luo, Linfeng Zhang, Yanting Zhang, Yu Qiao, and Wenhai Wang. 2025. Vico: A training strategy towards semantic aware dynamic high-resolution. *arXiv preprint arXiv:2510.12793*.

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

Tri Dao. 2024. Flashattention-2: Faster attention with better parallelism and work partitioning. In *International Conference on Learning Representations*, volume 2024, pages 35549–35562.

Tri Dao, Dan Fu, Stefano Ermon, Atri Rudra, and Christopher Ré. 2022. Flashattention: Fast and memory-efficient exact attention with io-awareness. *Advances in neural information processing systems*, 35:16344–16359.

Jinhong Deng, Wen Li, Joey Tianyi Zhou, and Yang He. 2026. Scope: Saliency-coverage oriented token pruning for efficient multimodel llms. *Advances in Neural Information Processing Systems*, 38:161527–161552.

Sixun Dong, Juhua Hu, Mian Zhang, Ming Yin, Yanjie Fu, and Qi Qian. 2025. Mmtok: Multimodal coverage maximization for efficient inference of vlms. *arXiv preprint arXiv:2508.18264*.

Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, and 1 others. 2020. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*.

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

Chaoyou Fu, Peixian Chen, Yunhang Shen, Yulei Qin, Mengdan Zhang, Xu Lin, Jinrui Yang, Xiawu Zheng, Ke Li, Xing Sun, and 1 others. 2026. Mme: A comprehensive evaluation benchmark for multimodal large language models. *Advances in Neural Information Processing Systems*, 38.

Dong Guo, Faming Wu, Feida Zhu, Fuxing Leng, Guang Shi, Haobin Chen, Haoqi Fan, Jian Wang, Jianyu Jiang, Jiawei Wang, and 1 others. 2025. Seed1. 5-vl technical report. *arXiv preprint arXiv:2505.07062*.

Yanwei Li, Yuechen Zhang, Chengyao Wang, Zhisheng Zhong, Yixin Chen, Ruihang Chu, Shaoteng Liu, and Jiaya Jia. 2025. Mini-gemini: Mining the potential of multi-modality vision language models. *IEEE Transactions on Pattern Analysis and Machine Intelligence*.

Yifan Li, Yifan Du, Kun Zhou, Jinpeng Wang, Xin Zhao, and Ji-Rong Wen. 2023. Evaluating object hallucination in large vision-language models. In *Proceedings of the 2023 conference on empirical methods in natural language processing*, pages 292–305.

Chenfei Liao, Wensong Wang, Zichen Wen, Xu Zheng, Yiyu Wang, Haocong He, Yuanhuiyi Lyu, Lutao Jiang, Xin Zou, Yuqian Fu, and 1 others. 2025. Are we using the right benchmark: An evaluation framework for visual token compression methods. *arXiv preprint arXiv:2510.07143*.

Ji Lin, Jiaming Tang, Haotian Tang, Shang Yang, Wei-Ming Chen, Wei-Chen Wang, Guangxuan Xiao, Xingyu Dang, Chuang Gan, and Song Han. 2024. Awq: Activation-aware weight quantization for on-device llm compression and acceleration. In *Proceedings of Machine Learning and Systems*, volume 6, pages 87–100.

Zichuan Lin, Yicheng Liu, Yang Yang, Lvfang Tao, and Deheng Ye. 2025. Adaptvision: Efficient vision-language models via adaptive visual acquisition. *arXiv preprint arXiv:2512.03794*.

Haotian Liu, Chunyuan Li, Yuheng Li, Bo Li, Yuanhan Zhang, Sheng Shen, and Yong Jae Lee. 2024a. Llava-next: Improved reasoning, ocr, and world knowledge.

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

Wenzhuo Liu, Weijie Yin, Fei Zhu, Shijie Ma, Haiyang Guo, Xiao-Hui Li, Cheng-Lin Liu, and 1 others. 2026. One patch doesn't fit all: Adaptive patching for native-resolution multimodal large language models. In *The Fourteenth International Conference on Learning Representations*.

Xuyang Liu, Yiyu Wang, Junpeng Ma, and Linfeng Zhang. 2025. Video compression commander: Plug-and-play inference acceleration for video large language models. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pages 1910–1924.

Yuan Liu, Haodong Duan, Yuanhan Zhang, Bo Li, Songyang Zhang, Wangbo Zhao, Yike Yuan, Jiaqi Wang, Conghui He, Ziwei Liu, and 1 others. 2024b. Mmbench: Is your multi-modal model an all-around player? In *European conference on computer vision*, pages 216–233. Springer.

Yuliang Liu, Zhang Li, Mingxin Huang, Biao Yang, Wenwen Yu, Chunyuan Li, Xu-Cheng Yin, Cheng-Lin Liu, Lianwen Jin, and Xiang Bai. 2024c. Ocrbench: on the hidden mystery of ocr in large multimodal models. *Science China Information Sciences*, 67(12):220102.

Ahmed Masry, Xuan Long Do, Jia Qing Tan, Shafiq Joty, and Enamul Hoque. 2022. Chartqa: A benchmark for question answering about charts with visual and logical reasoning. In *Findings of the association for computational linguistics: ACL 2022*, pages 2263–2279.

Minesh Mathew, Dimosthenis Karatzas, and CV Jawahar. 2021. Docvqa: A dataset for vqa on document images. In *Proceedings of the IEEE/CVF winter conference on applications of computer vision*, pages 2200–2209.

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

Amanpreet Singh, Vivek Natarajan, Meet Shah, Yu Jiang, Xinlei Chen, Dhruv Batra, Devi Parikh, and Marcus Rohrbach. 2019. Towards vqa models that can read. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 8317–8326.

Rinyoichi Takezoe, Yaqian Li, Zihao Bo, Anzhou Hou, Mo Guang, and Kaiwen Long. 2026. Learnpruner: Rethinking attention-based token pruning in vision language models. *arXiv preprint arXiv:2604.23950*.

Gemini Team, Rohan Anil, Sebastian Borgeaud, Jean-Baptiste Alayrac, Jiahui Yu, Radu Soricut, Johan Schalkwyk, Andrew M Dai, Anja Hauth, Katie Millican, and 1 others. 2023. Gemini: a family of highly capable multimodal models. *arXiv preprint arXiv:2312.11805*.

HyperAI Team, Yuchen Liu, Kaiyang Han, Zhiqiang Xia, Yuhang Dong, Chen Song, Kangyu Tang, Jiaming Xu, Xiushi Feng, WenXuan Yu, and 1 others. 2025. Hypervl: An efficient and dynamic multimodal large language model for edge devices. *arXiv preprint arXiv:2512.14052*.

Haodong Wang, Junjie Liu, Zicong Hong, Qianli Liu, Jian Lin, Song Guo, and Xu Chen. 2026a. Twinquant: Learnable subspace decomposition for 4-bit llm quantization. *arXiv preprint arXiv:2606.01556*.

Nan Wang, Zhiwei Jin, Chen Chen, and Haonan Lu. 2026b. Pixelprune: Pixel-level adaptive visual token reduction via predictive coding. *arXiv preprint arXiv:2604.00886*.

Weiyun Wang, Zhangwei Gao, Lixin Gu, Hengjun Pu, Long Cui, Xingguang Wei, Zhaoyang Liu, Linglin Jing, Shenglong Ye, Jie Shao, and 1 others. 2025a. Internvl3.5: Advancing open-source multimodal models in versatility, reasoning, and efficiency. *arXiv preprint arXiv:2508.18265*.

Yiyu Wang, Xuyang Liu, Xiyan Gui, Xinying Lin, Boxue Yang, Chenfei Liao, Tailai Chen, and Linfeng Zhang. 2025b. Accelerating streaming video large language models via hierarchical token compression. *arXiv preprint arXiv:2512.00891*.

Zichen Wen, Yifeng Gao, Shaobo Wang, Junyuan Zhang, Qintong Zhang, Weijia Li, Conghui He, and Linfeng Zhang. 2025. 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 9972–9991.

X.AI. 2024. Grok-1.5 vision preview. https://x.ai/blog/grok-1.5v.

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

Senqiao Yang, Junyi Li, Xin Lai, Jinming Wu, Wei Li, Zejun MA, Bei Yu, Hengshuang Zhao, and Jiaya Jia. 2026. Visionthink: Smart and efficient vision language model via reinforcement learning. *Advances in Neural Information Processing Systems*, 38:95187–95227.

Xubing Ye, Yukang Gan, Xiaoke Huang, Yixiao Ge, and Yansong Tang. 2025. Voco-llama: Towards vision compression with large language models. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pages 29836–29846.

Kaichen Zhang, Bo Li, Peiyuan Zhang, Fanyi Pu, Joshua Adrian Cahyono, Kairui Hu, Shuai Liu, Yuanhan Zhang, Jingkang Yang, Chunyuan Li, and 1 others. 2025. Lmms-eval: Reality check on the evaluation of large multimodal models. In *Findings of the Association for Computational Linguistics: NAACL 2025*, pages 881–916.

Qizhe Zhang, Mengzhen Liu, Lichen Li, Ming Lu, Yuan Zhang, Junwen Pan, Qi She, and Shanghang Zhang. 2026. Beyond attention or similarity: Maximizing conditional diversity for token pruning in mllms. *Advances in Neural Information Processing Systems*, 38:25438–25468.

Yuan Zhang, Chun-Kai Fan, Junpeng Ma, Wenzhao Zheng, Tao Huang, Kuan Cheng, Denis Gudovskiy, Tomoyuki Okuno, Yohei Nakata, Kurt Keutzer, and 1 others. 2024. Sparsevlm: Visual token sparsification for efficient vision-language model inference. *arXiv preprint arXiv:2410.04417*.

Xin Zou, Di Lu, Yizhou Wang, Yibo Yan, Yuanhuiyi Lyu, Xu Zheng, Linfeng Zhang, and Xuming Hu. 2026. Don't just chase "highlighted tokens" in mllms: Revisiting visual holistic context retention. *Advances in Neural Information Processing Systems*, 38:39800–39832.

A Detailed Experimental Setup

A.1 Models, Baselines, and Evaluation Framework

To ensure fair and reproducible comparisons, we detail the backbone models, token-reduction baselines, and evaluation framework used in our experiments.

Models. We integrate PACE into **Qwen2.5-VL-3B**[1] and **Qwen2.5-VL-7B**[2] (Bai et al., 2025b). Qwen2.5-VL uses native dynamic-resolution processing to convert images of different sizes into variable-length visual token sequences. Its visual backbone combines a dynamic-resolution Vision Transformer with Window Attention to reduce self-attention cost. Because visual-token length still grows with input resolution, high-resolution inputs remain expensive to encode. This design makes Qwen2.5-VL suitable for evaluating PACE's adaptive pixel allocation before full visual encoding.

Baselines. We compare PACE with six visual-token reduction methods:

- **FastV** (Chen et al., 2024a): Discards visual tokens exhibiting minimal attention scores within early LLM layers, thereby truncating the context sequence for subsequent layers.
- **SparseVLM** (Zhang et al., 2024): Deploys text-guided, training-free sparsification utilizing decoder self-attention to assess token importance, paired with a token recycling mechanism.
- **DivPrune** (Alvar et al., 2025): Formulates token pruning mathematically as a Max-Min Diversity Problem, ensuring the selected subset remains visually and semantically heterogeneous.
- **DART** (Wen et al., 2025): Evaluates contextual duplication to purge visual tokens highly redundant with selected pivot tokens while preserving distinct visual representations.
- **VisionZip** (Yang et al., 2025): Attenuates sequences through sequential token selection and fusion, merging redundant local patches into compressed contextual representations.
- **MMTok** (Dong et al., 2025): Models visual-token selection explicitly as a submodular maximum coverage problem, employing greedy heuristics to optimize conceptual graph overlap.

All six baselines operate after visual representation extraction. They reduce LLM prefill cost but do not change the cost of encoding the original high-resolution input with the visual backbone.

Evaluation Framework. We standardize all experiments using the open-source **`lmms-eval`** framework (Zhang et al., 2025). This pipeline provides a unified evaluation suite encompassing standardized task definitions, exact prompt templates, deterministic generation configurations, and uniform metric computation. Unless explicitly stated otherwise, we adhere strictly to the official `lmms-eval` configuration to ensure direct comparability across all methods.

A.2 Detailed Benchmark Descriptions

We comprehensively evaluate PACE across nine diverse benchmarks covering disparate visual processing demands, with their statistical properties cataloged in Table 5.

- **RealWorldQA** (X.AI, 2024): Evaluates general visual understanding in real-world scenes, including object localization and scene-level awareness.
- **POPE** (Li et al., 2023): Measures object hallucination under different visual conditions.
- **MME** (Fu et al., 2026): Covers perception, OCR, logical reasoning, and commonsense tasks.
- **MMBench** (Liu et al., 2024b): Uses multiple-choice questions to evaluate a range of multimodal capabilities.
- **MMStar** (Chen et al., 2024b): Contains 1,500 curated questions designed to reduce language-only shortcuts and require visual evidence.
- **OCRBench** (Liu et al., 2024c): Evaluates OCR-related capabilities across scene text, documents, and mathematical content.
- **TextVQA** (Singh et al., 2019): Evaluates reading and reasoning over text embedded in natural images.

[1] https://huggingface.co/Qwen/Qwen2.5-VL-3B-Instruct

[2] https://huggingface.co/Qwen/Qwen2.5-VL-7B-Instruct

- **ChartQA** (Masry et al., 2022): Evaluates numerical and structural reasoning over charts.
- **DocVQA** (Mathew et al., 2021): Evaluates document reading and layout understanding on scanned documents.

Dataset	Total	Avg. Resolution	Avg. Pixels	Duplicates	Dup. Ratio
RealWorldQA	765	1316×1030	1,341,566	3	0.39%
OCRBench	1,000	615×732	1,017,574	70	7.00%
MMStar	1,500	511×391	245,161	70	4.67%
MME	2,374	1086×945	1,881,888	1,197	50.42%
ChartQA	2,500	768×583	454,992	991	39.64%
$\text{MMBench}_{\text{en_dev}}$	4,329	440×338	153,326	3,208	74.10%
$\text{TextVQA}_{\text{val}}$	5,000	952×819	770,513	1,834	36.68%
$\text{DocVQA}_{\text{val}}$	5,349	1783×2099	3,935,008	4,064	75.98%
POPE	9,000	585×479	277,243	8,500	94.44%

Table 5: **Statistics of the evaluation datasets.** The table reports dataset size, native resolution, pixel count, and duplication statistics. "Duplicates" counts additional question instances that reuse an image already paired with another question.

A.3 Detailed Implementation Specifics

For the APC module, we fix the feature preview depth to the initial ViT block ($K = 1$) to limit preprocessing overhead. We set the global-local balancing weight α to 0.6 and the local contrast regularization term γ to 1.5. The target retention ratio $r = \rho$ determines the final pixel count relative to the original image area. Consequently, the image width and height are both scaled by a factor of $\sqrt{r}$ using bicubic interpolation (`Image.Resampling.BICUBIC`).

In the DDAE module, the visual self-attention score (S_{vis}) comes directly from the final ViT block. The semantic relevance map (S_{llm}) is aggregated from the cross-modal attention of the second LLM block ($L_{ext} = 2$), with weights averaged uniformly across all valid tokens. We set the temperature scalar τ for confidence-weighted softmax fusion to 0.5. All experiments use greedy decoding (temperature $= 0$). We conduct all efficiency profiling and latency measurements, including TTFT and TPOT calculations, on a single NVIDIA RTX 4090 GPU.

B Additional Experiments

B.1 Hyperparameter Robustness

PACE uses a small set of interpretable hyperparameters. The scaling variable γ limits over-retention caused by noisy, high-frequency patterns in the local contrast score. The temperature τ controls the sharpness of DDAE's softmax confidence weights.

Sensitivity of Global-Local Balancing Weight (α). The coefficient α controls the balance between global redundancy estimation and local detail preservation in APC. Specifically, a larger α places greater emphasis on the global information density score ρ_g, whereas a smaller α increases the contribution of the local detail contrast score ρ_d. To examine this trade-off, we evaluate different α values under the restrictive 5% token budget in the dynamic-resolution setting of Qwen2.5-VL-7B.

Table 6 reports the fine-grained MMStar breakdown. Although larger values such as $\alpha = 1.0$ and $\alpha = 0.8$ improve specific categories, including coarse perception and math reasoning, they consistently degrade several layout- and relation-sensitive dimensions. For example, compared with the default $\alpha = 0.6$, setting $\alpha = 1.0$ decreases instance reasoning by 3.61 points, logical reasoning by 4.35 points, and science & technology by 3.03 points.

Conversely, smaller values such as $\alpha = 0.4$ and $\alpha = 0.2$ over-emphasize local contrast cues. This makes the adaptive resolution policy more sensitive to isolated high-frequency patterns and weakens its ability to preserve coherent global layouts. As a result, these settings reduce the MMStar average by 3.32 and 2.90 points, respectively. Overall, $\alpha = 0.6$ provides the most balanced behavior, securing a stable compromise between suppressing globally redundant regions and retaining task-critical local cues under severe token constraints.

Sensitivity of the Top-Detail Percentile. We compare 5%, 10%, and 20% tail fractions while holding all other settings fixed. Their performance is very similar, so we use 10% as a fixed midpoint across models, datasets, and token budgets.

B.2 Shallow Feature Preview Ablation

We isolate the preview signal while keeping APC's adaptive resizing and DDAE unchanged under the native dynamic-resolution setting at 10% visual-token retention. A literal removal of all preview signals would make image-dependent allocation impossible and reduce APC to static resolution; we therefore replace the shallow ViT feature with four training-free pixel statistics.

Pixel statistics measure color variation or local high-frequency response but cannot reliably separate semantic information density from texture, noise, decorative patterns, or irrelevant edges. Edge density is comparatively effective on ChartQA because plotted lines convey useful structure, whereas

α	Coarse Perc.	Fine-grained Perc.	Instance Reason.	Logical Reason.	Math	Science & Tech.	Avg.
0.6 (Default)	59.48 (0.00)	**39.07** (0.00)	**57.63** (0.00)	**48.47** (0.00)	43.45 (0.00)	**33.06** (0.00)	**46.86** (0.00)
1.0	**62.00** (+2.52)	37.25 (-1.82)	54.02 (-3.61)	44.12 (-4.35)	46.25 (+2.80)	30.03 (-3.03)	45.16 (-1.70)
0.8	60.35 (+0.87)	33.65 (-5.42)	53.64 (-3.99)	45.63 (-2.84)	**50.02** (+6.57)	27.52 (-5.54)	45.14 (-1.72)
0.4	58.32 (-1.16)	36.42 (-2.65)	53.83 (-3.80)	41.77 (-6.70)	44.24 (+0.79)	26.68 (-6.38)	43.54 (-3.32)
0.2	58.52 (-0.96)	33.41 (-5.66)	52.57 (-5.06)	44.56 (-3.91)	43.84 (+0.39)	30.84 (-2.22)	43.96 (-2.90)

Table 6: **Sensitivity of APC's global-local balancing weight α on MMStar under the dynamic-resolution setting.** Results are reported on Qwen2.5-VL-7B at the 5% token budget.

APC Feature	RealWorldQA	ChartQA
Vanilla (100%)	69.54	83.92
RGB statistics	63.40	47.64
Color entropy	65.10	51.40
Edge density	64.18	55.96
Laplacian	64.31	55.04
Shallow Feature Preview	**67.32**	**56.40**

Table 7: **Ablation of APC's information-density feature.** Compressed variants retain 10% of visual tokens on Qwen2.5-VL-7B; Vanilla is shown only as the uncompressed reference. The shallow ViT preview is the strongest adaptive feature on both a natural-image and a chart benchmark.

the Laplacian is more sensitive to fine-scale noise. The shallow ViT preview provides a lightweight semantic prior and transfers more consistently across the two domains.

B.3 Cross-Model Generalization to InternVL3.5-4B

We evaluate PACE on InternVL3.5-4B, whose dynamic multi-tile pipeline differs substantially from Qwen-style native-resolution patching. For applicable baselines, "-T" prunes within each tile before concatenation, whereas "-G" concatenates all tile tokens before one global selection under the same total budget. We report both reasonable treatments of tile boundaries.

PACE achieves the strongest overall performance, surpassing the best baseline average by 5.1, 4.7, and 4.0 points at 25%, 20%, and 10%, respectively. It is not uniformly best on every task, but it provides the most balanced result and preserves substantially more OCRBench and DocVQA performance. These results support transfer to a distinct tiling-based architecture; latency and preview amortization remain backbone- and resolution-dependent.

B.4 Comprehensive Experimental Results

We report complete results for **Qwen2.5-VL-7B** (Table 9) and **Qwen2.5-VL-3B** (Table 10) under two input settings:

- **Fixed-resolution setting (MinPix = MaxPix = $2048 \times 28 \times 28$):** Pads or resizes every input to the same pixel budget, which standardizes the input size and can introduce additional redundant patches.
- **Dynamic-resolution setting (MinPix = $256 \times 28 \times 28$, MaxPix = $2048 \times 28 \times 28$):** Uses the model's native resolution range while preserving each image's aspect ratio.

Across both settings, post-encoder baselines degrade substantially under the **fixed-resolution setting** at 10% and 5% retention, particularly on detail-sensitive benchmarks. PACE degrades more gradually across both model sizes, consistent with the benefit of adapting the input resolution before full visual encoding.

B.5 TTFT Profiling Across Retention Budgets

Table 11 extends the main 10% measurement to milder retention budgets. TTFT is the complete pre-generation latency: Vanilla includes vision encoding and LLM prefill, while PACE additionally includes the Shallow Feature Preview and adaptive resizing. The isolated encoder and prefill columns exclude these APC overheads. Autoregressive decoding is not included because it is not accelerated by PACE.

The average TTFT speedup across DocVQA and TextVQA is $1.10\times$, $1.67\times$, $2.64\times$, and $3.13\times$ at 80%, 50%, 20%, and 10% retention, respectively. The modest 80% gain reflects that preview overhead is nearly fixed while relatively little encoder and prefill computation is removed; the overhead is progressively amortized at stricter budgets.

Dataset-wise 10% profiling. We further report the number of evaluated samples and average per-

Method	RWQA	POPE	MME	MMB	MMStar	ChartQA	OCRB	TextVQA	DocVQA	Avg.
Vanilla (100%)	65.62	89.47	2280.88	81.36	66.53	85.72	80.80	75.82	91.08	100.0%
Retain 25% visual tokens										
FastV	57.91	87.24	2108.89	76.46	54.09	**59.68**	38.10	66.39	57.65	80.1%
DivPrune-T	58.04	87.97	2011.20	74.74	52.34	48.16	30.30	56.58	42.05	73.3%
DivPrune-G	56.86	87.65	2026.90	77.58	54.81	50.36	34.00	57.05	46.54	75.4%
VisionZip-T	**58.43**	88.25	1922.14	73.71	50.69	31.72	16.10	45.22	24.05	64.6%
MMTok-T	57.65	88.17	2023.92	74.66	51.08	38.72	12.10	34.89	27.26	64.4%
MMTok-G	58.04	**88.42**	2049.51	76.46	51.82	40.16	12.70	31.89	29.59	65.1%
PACE (Ours)	58.17	88.03	**2238.93**	**80.58**	**58.47**	55.64	**52.30**	**66.46**	**70.36**	**85.2%**
Retain 20% visual tokens										
FastV	56.21	85.95	2032.87	74.66	52.17	**51.88**	33.70	**63.91**	52.76	76.2%
DivPrune-T	55.29	87.45	2002.21	73.80	50.08	42.48	23.30	53.15	36.80	69.4%
DivPrune-G	54.12	86.79	1966.18	76.12	52.39	44.68	29.70	53.15	40.31	71.2%
VisionZip-T	55.82	**88.25**	1884.68	73.37	49.56	26.80	11.70	39.89	19.31	61.2%
MMTok-T	55.82	87.57	1967.05	72.42	48.57	32.88	8.50	31.44	23.33	60.8%
MMTok-G	**56.60**	87.78	1968.71	74.48	48.43	32.40	9.80	28.94	24.49	61.1%
PACE (Ours)	55.42	87.55	**2183.78**	**78.44**	**57.30**	48.24	**45.80**	63.43	**64.62**	**80.9%**
Retain 10% visual tokens										
FastV	51.11	83.33	1804.05	71.13	43.06	**36.80**	21.80	**56.39**	39.00	65.5%
DivPrune-T	51.50	86.19	1853.69	70.79	45.53	30.00	14.00	43.09	25.23	60.9%
DivPrune-G	51.63	85.16	1851.55	71.48	46.99	29.36	17.50	43.77	28.13	62.0%
VisionZip-T	50.46	86.33	1819.59	67.70	43.35	20.20	7.60	26.03	13.99	53.8%
MMTok-T	53.20	86.08	1848.30	69.07	43.23	23.00	6.10	25.41	16.06	54.9%
MMTok-G	**54.38**	**86.42**	1924.16	70.53	44.06	21.92	6.00	22.39	17.76	55.4%
PACE (Ours)	52.16	84.55	**2083.74**	**75.34**	**51.35**	30.04	**29.20**	53.86	**43.82**	**69.5%**

Table 8: **Cross-model performance on InternVL3.5-4B.** Benchmark scores are reported for a variable-token, multi-tile architecture at 25%, 20%, and 10% retention. Avg. first normalizes each score by the corresponding Vanilla score and then averages the nine ratios; bold marks the best compressed result within each budget.

Method	RealWorldQA Acc. ↑	POPE F1 ↑	MME P+C ↑	MMBench Acc. ↑	MMStar Acc. ↑	ChartQA Acc. ↑	OCRBench Acc. ↑	TextVQA Acc. ↑	DocVQA ANLS ↑	Avg. ↑
Fixed-resolution setting (MinPix = $2048 \times 28 \times 28$**, MaxPix =** $2048 \times 28 \times 28$**)**										
Vanilla (100% Tokens)	69.54	86.36	2317	82.99	63.91	78.20	77.30	82.37	94.74	100.0%
Retain 20% $\bar{T}$										
FastV (ECCV'24)	64.58	80.99	2256	80.76	54.79	64.12	66.60	79.16	76.58	90.2%
SparseVLM (ICML'25)	66.41	83.23	2258	81.36	55.65	70.00	56.54	80.33	73.56	90.3%
DivPrune (CVPR'25)	61.83	84.07	2248	80.07	54.66	51.56	51.50	69.99	49.40	81.7%
DART (EMNLP'25)	63.53	82.81	2278	79.64	55.83	58.72	54.00	69.48	50.58	83.5%
VisionZip (CVPR'25)	**67.06**	85.50	2317	81.01	58.93	69.68	64.60	77.52	75.77	92.5%
MMTok (ICLR'26)	63.79	84.81	2278	82.22	58.17	68.40	65.80	76.78	74.25	91.4%
PACE (w/o APC)	66.93	85.29	2310	81.36	59.29	72.84	69.10	80.72	83.00	94.9%
PACE (Ours)	66.93	**86.23**	**2322**	**83.08**	**62.47**	**78.56**	**79.00**	**81.66**	**86.84**	**98.6%**
Retain 10% $\bar{T}$										
FastV (ECCV'24)	59.08	73.28	2154	77.23	48.78	51.80	52.40	74.23	59.56	79.9%
SparseVLM (ICML'25)	60.39	76.25	2145	77.23	51.30	61.00	53.00	76.30	49.61	81.4%
DivPrune (CVPR'25)	57.25	81.73	2158	76.12	50.22	39.00	41.70	59.32	34.58	72.5%
DART (EMNLP'25)	58.82	78.60	2074	78.26	48.89	44.88	42.70	57.12	33.65	72.6%
VisionZip (CVPR'25)	65.23	83.55	2147	79.04	54.90	53.08	49.40	67.16	49.55	81.1%
MMTok (ICLR'26)	58.82	82.44	2218	79.47	54.04	51.04	51.30	67.33	51.37	80.4%
PACE (w/o APC)	65.36	82.27	2228	80.07	55.25	65.76	58.00	76.18	65.54	87.7%
PACE (Ours)	**68.37**	**84.99**	**2314**	**83.33**	**59.08**	**73.52**	**70.90**	**78.42**	**69.55**	**93.8%**
Retain 5% $\bar{T}$										
FastV (ECCV'24)	56.08	62.45	1995	73.54	45.88	36.52	41.40	66.21	44.17	69.6%
SparseVLM (ICML'25)	56.60	65.25	2011	72.08	45.01	46.24	43.10	69.53	30.99	70.3%
DivPrune (CVPR'25)	53.46	77.66	1926	72.77	45.20	28.96	29.80	41.28	22.64	62.0%
DART (EMNLP'25)	52.68	71.33	1946	73.97	42.80	30.76	32.90	45.01	23.37	62.2%
VisionZip (CVPR'25)	61.57	78.65	2029	75.17	48.63	40.96	38.30	55.47	29.40	70.5%
MMTok (ICLR'26)	54.12	78.70	2133	75.17	47.00	32.76	36.00	50.95	28.85	67.3%
PACE (w/o APC)	60.39	76.59	2117	75.95	49.26	53.00	46.10	67.32	45.38	76.9%
PACE (Ours)	**63.79**	**80.70**	**2245**	**80.07**	**56.02**	**61.84**	**57.80**	**71.56**	**48.94**	**84.3%**
Dynamic-resolution setting (MinPix = $256 \times 28 \times 28$**, MaxPix =** $2048 \times 28 \times 28$**)**										
Vanilla (100% Tokens)	69.54	86.46	2308	84.28	62.75	83.92	84.30	82.94	94.73	100.0%
Retain 20% $\bar{T}$										
FastV (ECCV'24)	64.44	74.76	2157	78.18	49.69	62.60	59.90	77.49	75.93	84.9%
SparseVLM (ICML'25)	64.44	79.18	2195	77.06	50.06	66.00	56.10	**80.14**	72.92	85.5%
DivPrune (CVPR'25)	63.14	81.21	2076	76.37	49.39	48.16	49.20	63.93	48.56	76.5%
DART (EMNLP'25)	63.01	80.72	2200	77.75	48.43	44.04	52.70	65.64	50.07	77.3%
VisionZip (CVPR'25)	65.49	83.11	2190	80.58	55.61	51.00	58.30	69.18	74.76	84.6%
MMTok (ICLR'26)	63.01	83.05	2209	79.12	52.36	59.60	60.20	72.00	73.81	85.2%
PACE (w/o APC)	65.49	82.08	2271	79.21	54.11	**70.48**	64.20	78.61	82.37	90.0%
PACE (Ours)	**68.10**	**83.52**	**2276**	**80.93**	**56.24**	68.84	**72.20**	78.11	**86.35**	**92.4%**
Retain 10% $\bar{T}$										
FastV (ECCV'24)	58.56	65.45	1946	73.28	44.99	45.52	45.00	70.69	58.66	73.1%
SparseVLM (ICML'25)	59.22	69.03	1930	63.65	41.45	50.00	36.20	**75.19**	48.40	70.5%
DivPrune (CVPR'25)	58.69	76.61	1916	72.25	43.57	36.48	37.50	50.77	33.91	66.2%
DART (EMNLP'25)	56.99	73.49	2017	73.11	44.25	31.20	42.00	51.76	33.23	66.2%
VisionZip (CVPR'25)	63.14	**78.97**	1959	75.00	49.03	39.56	39.20	57.67	48.55	72.1%
MMTok (ICLR'26)	58.69	77.63	2020	75.08	47.31	38.96	44.80	59.07	50.55	72.3%
PACE (w/o APC)	63.79	74.89	2056	74.66	48.76	51.60	46.30	70.81	64.59	78.2%
PACE (Ours)	**67.32**	76.86	**2112**	**77.32**	**51.12**	**56.40**	**53.00**	72.81	**68.49**	**82.3%**
Retain 5% $\bar{T}$										
FastV (ECCV'24)	51.63	47.57	1646	65.21	39.26	31.24	30.50	58.27	43.15	58.9%
SparseVLM (ICML'25)	54.90	38.89	1542	47.25	34.97	29.00	19.70	63.04	29.68	52.0%
DivPrune (CVPR'25)	54.51	71.14	1767	66.58	40.58	25.32	24.50	36.85	22.23	56.5%
DART (EMNLP'25)	52.94	61.60	**1846**	67.44	38.48	22.24	32.70	39.55	23.03	56.2%
VisionZip (CVPR'25)	61.44	**71.22**	1731	69.59	41.36	26.44	23.60	43.96	28.87	59.7%
MMTok (ICLR'26)	51.37	68.85	1808	66.84	40.26	23.76	31.40	43.67	28.22	58.1%
PACE (w/o APC)	58.69	61.85	1736	69.07	43.55	35.84	28.50	57.51	44.28	63.9%
PACE (Ours)	**63.14**	64.75	1783	**72.51**	**45.36**	**38.52**	**32.80**	**63.36**	**47.64**	**68.1%**

Table 9: **Comprehensive performance on Qwen2.5-VL-7B.** Results cover the fixed- and dynamic-resolution settings at 20%, 10%, and 5% token-retention budgets.

Method	RealWorldQA Acc. ↑	POPE F1 ↑	MME P+C ↑	MMBench Acc. ↑	MMStar Acc. ↑	ChartQA Acc. ↑	OCRBench Acc. ↑	TextVQA Acc. ↑	DocVQA ANLS ↑	Avg. ↑
Fixed-resolution setting (MinPix = $2048 \times 28 \times 28$**, MaxPix =** $2048 \times 28 \times 28$**)**										
Vanilla (100% Tokens)	67.52	87.40	2088	77.58	55.93	84.04	72.30	78.69	93.02	100.0%
Retain 20% $\bar{T}$										
FastV (ECCV'24)	52.03	84.44	1998	73.97	50.93	74.32	60.10	74.51	74.25	89.1%
SparseVLM (ICML'25)	51.50	86.06	2044	73.88	51.31	68.24	54.90	74.47	62.74	86.5%
DivPrune (CVPR'25)	48.89	84.73	1920	72.42	50.03	55.76	41.80	59.69	44.31	76.9%
DART (EMNLP'25)	53.20	84.54	2043	73.45	52.12	68.44	47.30	64.01	52.16	82.8%
VisionZip (CVPR'25)	53.99	86.72	2074	75.08	53.09	77.36	57.70	69.17	70.93	89.6%
MMTok (ICLR'26)	53.73	86.62	2017	75.25	54.80	74.16	60.20	72.47	69.51	89.8%
PACE (w/o APC)	53.86	87.08	2102	75.52	53.37	79.04	60.20	72.63	75.14	91.5%
PACE (Ours)	**55.56**	**87.61**	**2138**	**76.98**	**55.61**	**82.60**	**71.40**	**76.59**	**82.19**	**96.3%**
Retain 10% $\bar{T}$										
FastV (ECCV'24)	48.10	78.11	1900	70.87	47.82	60.56	47.40	69.65	54.84	79.3%
SparseVLM (ICML'25)	47.58	80.43	1922	69.58	47.21	43.92	40.70	68.31	40.02	74.1%
DivPrune (CVPR'25)	49.02	83.96	1893	70.10	49.01	45.68	32.70	50.44	30.87	70.5%
DART (EMNLP'25)	51.50	80.43	1863	69.84	46.59	51.48	34.80	50.61	32.97	71.1%
VisionZip (CVPR'25)	52.81	84.50	1932	71.64	50.59	60.84	42.50	57.23	43.32	77.9%
MMTok (ICLR'26)	53.07	85.22	1916	72.68	50.82	61.00	46.50	63.48	50.62	80.5%
PACE (w/o APC)	50.59	84.95	1950	73.11	50.82	67.80	47.70	65.06	53.95	82.0%
PACE (Ours)	**53.73**	**86.77**	**2027**	**74.31**	**53.95**	**76.56**	**60.50**	**70.33**	**61.95**	**88.8%**
Retain 5% $\bar{T}$										
FastV (ECCV'24)	42.48	65.23	1742	66.67	43.01	42.56	38.60	62.16	38.56	67.6%
SparseVLM (ICML'25)	40.78	64.91	1696	60.14	40.79	24.76	31.10	57.52	24.43	59.8%
DivPrune (CVPR'25)	49.02	83.96	1893	66.32	46.00	45.68	32.70	50.44	22.65	68.4%
DART (EMNLP'25)	48.76	73.92	1744	65.38	41.98	33.84	22.50	36.86	21.75	60.1%
VisionZip (CVPR'25)	52.81	84.50	**1932**	68.21	47.04	60.84	42.50	57.23	26.45	74.6%
MMTok (ICLR'26)	**53.07**	**85.22**	1919	67.87	47.43	61.00	**46.50**	**63.48**	31.29	76.8%
PACE (w/o APC)	47.06	79.96	1804	70.36	47.65	52.84	37.10	53.94	34.06	71.4%
PACE (Ours)	50.72	83.66	1896	**73.54**	**51.33**	**63.04**	46.10	61.33	**40.71**	**78.7%**
Dynamic-resolution setting (MinPix = $256 \times 28 \times 28$**, MaxPix =** $2048 \times 28 \times 28$**)**										
Vanilla (100% Tokens)	59.74	86.47	2154	78.01	55.67	83.36	77.90	78.75	92.89	100.0%
Retain 20% $\bar{T}$										
FastV (ECCV'24)	54.12	78.73	2002	71.56	46.91	64.76	54.50	73.29	72.84	85.5%
SparseVLM (ICML'25)	54.51	81.63	2005	70.18	46.07	56.08	46.20	**74.41**	61.74	82.1%
DivPrune (CVPR'25)	53.20	79.90	1877	69.15	45.42	51.40	43.40	56.52	43.77	75.0%
DART (EMNLP'25)	55.29	80.52	1926	70.10	46.39	53.00	45.30	55.77	51.40	77.3%
VisionZip (CVPR'25)	**57.25**	82.80	1905	71.90	49.21	65.48	51.30	64.22	70.17	84.7%
MMTok (ICLR'26)	55.42	82.23	2053	72.59	48.22	66.16	56.30	67.56	69.23	86.1%
PACE (w/o APC)	55.95	83.43	1948	72.51	49.97	70.48	56.90	71.49	74.49	88.0%
PACE (Ours)	56.86	**84.29**	**2065**	**75.69**	**52.83**	**71.20**	**64.80**	71.77	**81.74**	**92.0%**
Retain 10% $\bar{T}$										
FastV (ECCV'24)	49.93	68.80	1834	65.46	41.38	48.32	41.70	67.52	**54.09**	73.5%
SparseVLM (ICML'25)	47.97	72.65	1801	55.84	39.68	28.48	27.50	**69.37**	39.16	65.6%
DivPrune (CVPR'25)	51.11	75.54	1771	64.86	41.52	42.20	32.20	46.53	30.19	66.3%
DART (EMNLP'25)	53.99	72.37	1763	64.08	41.64	37.48	30.50	41.88	32.39	65.0%
VisionZip (CVPR'25)	**55.82**	77.53	1722	67.09	44.50	47.68	32.80	48.28	42.41	70.6%
MMTok (ICLR'26)	53.73	**79.05**	**1921**	68.47	43.58	47.60	40.90	56.51	49.81	74.6%
PACE (w/o APC)	52.81	77.79	1795	68.21	46.30	53.92	38.80	62.02	52.97	75.8%
PACE (Ours)	52.42	78.00	1855	**71.05**	**48.46**	**56.44**	**43.00**	63.52	52.42	**78.0%**
Retain 5% $\bar{T}$										
FastV (ECCV'24)	45.23	54.81	1630	54.04	37.36	31.08	26.80	**58.50**	37.71	59.8%
SparseVLM (ICML'25)	43.79	51.84	1533	38.66	35.52	16.16	13.00	57.54	23.82	50.3%
DivPrune (CVPR'25)	48.63	71.11	1577	57.82	36.92	33.52	21.30	36.47	21.94	57.2%
DART (EMNLP'25)	50.33	62.27	1548	55.93	37.34	25.76	22.10	30.35	21.09	54.2%
VisionZip (CVPR'25)	**51.90**	69.90	1557	59.62	39.89	32.96	18.90	35.46	25.93	58.3%
MMTok (ICLR'26)	48.89	**73.17**	**1709**	59.36	39.21	23.76	**28.50**	42.57	30.47	60.5%
PACE (w/o APC)	50.98	64.11	1541	60.48	**41.26**	**38.80**	22.20	48.90	33.49	61.8%
PACE (Ours)	50.72	65.31	1652	**62.03**	41.10	37.52	26.70	52.05	**39.57**	**64.3%**

Table 10: **Comprehensive performance on Qwen2.5-VL-3B.** Results cover the fixed- and dynamic-resolution settings at 20%, 10%, and 5% token-retention budgets.

Retention	Dataset	Encoder: Vanilla / PACE (Spd.)	Prefill: Vanilla / PACE (Spd.)	TTFT incl. APC: Vanilla / PACE (Spd.)
80%	DocVQA	144.77 / 109.07 (1.33×)	209.35 / 168.63 (1.24×)	354.12 / 321.57 (1.10×)
	TextVQA	156.27 / 117.33 (1.33×)	220.02 / 174.56 (1.26×)	376.29 / 339.98 (1.11×)
50%	DocVQA	144.69 / 66.28 (2.18×)	207.84 / 105.81 (1.96×)	352.54 / 210.30 (1.68×)
	TextVQA	155.24 / 70.17 (2.21×)	220.40 / 114.70 (1.92×)	375.64 / 224.88 (1.67×)
20%	DocVQA	144.73 / 51.97 (2.78×)	207.72 / 50.92 (4.08×)	352.45 / 138.53 (2.54×)
	TextVQA	155.20 / 49.46 (3.14×)	220.26 / 51.32 (4.29×)	375.46 / 137.34 (2.73×)
10%	**DocVQA**	**143.41 / 50.74** (2.83×)	**210.41 / 32.70** (6.43×)	**353.82 / 117.54** (3.01×)
	TextVQA	**154.28 / 48.20** (3.20×)	**223.68 / 32.69** (6.84×)	**377.96 / 116.03** (3.26×)

Table 11: **Latency across visual-token retention budgets on Qwen2.5-VL-7B.** Average per-sample milliseconds are measured under the fixed-resolution setting on one RTX 4090. At 80% retention (only 20% token reduction), PACE already improves TTFT by about 1.10×; its benefit grows as more redundant computation is removed.

sample latency over all nine benchmarks. As shown in Table 12, PACE consistently reduces both encoder-side and LLM-side latency. On average, it reduces encoder latency from 152.41 ms to 46.10 ms and prefill latency from 222.16 ms to 31.63 ms, corresponding to 3.32× and 7.02× speedups. Including APC and resizing, average TTFT decreases from 374.58 ms to 112.08 ms (3.34×).

B.6 Impact of Attention Token Sources

We compare three semantic cross-attention aggregation strategies for DDAE: **Vision + Text** (aggregating over all sequence tokens), **Text** (restricting strictly to linguistic tokens), and **Last** (relying exclusively on the terminal token).

Figure 8 shows that the **Vision + Text** strategy performs best on both MME and DocVQA. Restricting aggregation to linguistic or terminal tokens progressively reduces performance, with DocVQA dropping by nearly 10 points. This result suggests that complete-sequence aggregation better preserves the spatial and visual evidence needed for layout-sensitive comprehension.

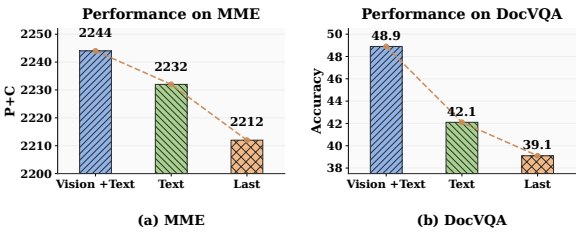

Figure 8: **Attention-token sources for DDAE.** We compare semantic attention aggregated from the complete sequence, text tokens only, or the final token. Complete aggregation best preserves both MME perception and DocVQA layout evidence.

B.7 Full DDAE Extraction-Depth Latency

Table 13 reports the complete latency counterpart to the quality ablation in Table 4. Measurements use the same 400 samples under two fixed-resolution settings. Delaying DDAE monotonically erodes prefill acceleration: moving from layer 2 to layer 24 reduces speedup from 2.11× to 1.09× at 2048 patch units and from 2.32× to 1.10× at 4096. Decode and time per output token remain approximately unchanged (≈ 1.00×), confirming that PACE's measured acceleration should be attributed to TTFT rather than the full generation process.

C Detailed Computational Complexity Analysis

We analyze PACE using **Qwen2.5-VL** (Bai et al., 2025b), distinguishing encoder patches from the post-merger visual tokens passed to the LLM.

C.1 Architecture and Stage-wise Bottlenecks of Qwen2.5-VL

An $H \times W$ image produces $N = HW/P^2$ encoder patches of dimension D_v. Each of the L_v ViT blocks costs $\mathcal{O}(ND_v^2 + N^2 D_v)$. A spatial merger then produces N_{vis} visual tokens in the LLM dimension D_l. With T prompt tokens, Vanilla has prefill length $S_0 = T + N_{vis}$ and per-layer cost $\mathcal{O}(S_0 D_l^2 + S_0^2 D_l)$. Autoregressive decoding determines TPOT and is unchanged by PACE.

C.2 Theoretical Complexity Reduction

Let $p_1 \in (0, 1]$ be APC's retained image-area ratio. The full encoder processes $N_{enc} \approx p_1 N$ patches, and the merger outputs $N_{vis}^c \approx p_1 N_{vis}$ tokens. Because the preview uses $K = 1$ block at the original

Benchmark	# Samples	Encoder Time (ms)			Prefill Time (ms)			TTFT incl. APC (ms)		
		Van.	PACE	Spd.	Van.	PACE	Spd.	Van.	PACE	Spd.
RealWorldQA	765	152.33	44.83	3.40×	222.86	31.59	7.05×	375.19	110.50	3.40×
POPE	9,000	154.90	47.71	3.25×	224.60	31.60	7.11×	379.50	114.37	3.32×
MME	2,374	152.28	46.85	3.25×	221.92	31.58	7.03×	374.20	112.78	3.32×
MMBench	4,329	154.02	43.23	3.56×	224.17	31.63	7.09×	378.19	108.91	3.47×
MMStar	1,500	153.76	44.04	3.49×	223.87	31.66	7.07×	377.63	109.87	3.44×
ChartQA	2,500	152.80	45.03	3.39×	223.33	31.57	7.07×	376.13	110.91	3.39×
OCRBench	1,000	153.13	44.50	3.44×	223.06	31.76	7.02×	376.20	110.38	3.41×
TextVQA	5,000	154.70	48.10	3.22×	224.53	31.68	7.09×	379.23	114.91	3.30×
DocVQA	5,349	143.79	50.58	2.84×	211.16	31.58	6.69×	354.94	116.09	3.06×
Macro Avg.	–	**152.41**	**46.10**	**3.32×**	**222.16**	**31.63**	**7.02×**	**374.58**	**112.08**	**3.34×**

Table 12: **Dataset-wise latency profiling on Qwen2.5-VL-7B at 10% retention.** Average per-sample latency is reported in milliseconds under the fixed-resolution setting. Isolated stage times exclude APC, whereas PACE TTFT includes its preview and resizing overhead.

Fixed-resolution setting	Depth	w/o DDAE (ms)	DDAE (ms)	Spd.
$2048 \times 28 \times 28$	2	74.52	35.26	2.11×
	8	74.52	42.51	1.75×
	16	74.52	55.32	1.35×
	24	74.52	68.12	1.09×
$4096 \times 28 \times 28$	2	134.65	58.01	2.32×
	8	134.65	75.57	1.78×
	16	134.65	99.11	1.36×
	24	134.65	122.77	1.10×

Table 13: **LLM prefill latency across DDAE extraction depths.** "w/o DDAE" processes the full visual sequence throughout prefill. Earlier extraction leaves fewer full-sequence layers and therefore yields greater acceleration.

resolution, the vision-side cost is

$$\begin{aligned} \mathcal{F}_{\text{ViT}}^{\text{PACE}} &= \mathcal{C}_{\text{prev}} + \mathcal{C}_{\text{enc}}, \\ \mathcal{C}_{\text{prev}} &= \mathcal{O}(KND_v^2 + KN^2D_v), \quad K = 1, \\ \mathcal{C}_{\text{enc}} &= \mathcal{O}(L_v p_1 N D_v^2 + L_v p_1^2 N^2 D_v). \end{aligned} \tag{8}$$

Let $p_2 \in (0, 1]$ be DDAE's retention after condensation and $\mathcal{B} = p_1 p_2$ the final ratio relative to Vanilla. Then

$$N_{keep} = p_2 N_{vis}^c \approx \mathcal{B} N_{vis}. \tag{9}$$

Define $S_{pre} = T + N_{vis}^c$ and $S_{post} = T + N_{keep}$. Extracting after layer L_{ext} in an L_l-layer LLM gives

$$\begin{aligned} \mathcal{F}_{\text{pre}}^{\text{PACE}} = \mathcal{O}(L_{ext} S_{pre} D_l^2 + L_{ext} S_{pre}^2 D_l) \\ + \mathcal{O}((L_l - L_{ext}) S_{post} D_l^2 \\ + (L_l - L_{ext}) S_{post}^2 D_l). \end{aligned} \tag{10}$$

APC therefore reduces full-encoder computation, whereas earlier DDAE extraction leaves fewer LLM layers operating on S_{pre}.

D Qualitative Results

To better understand the behavior of APC across heterogeneous visual domains, we visualize the distribution of the information density score across all evaluation benchmarks in Figure 9, alongside representative samples at varying density levels.

As shown in Figure 9, the information density score varies noticeably across benchmarks. General visual-understanding datasets, such as RealWorldQA, POPE, MME, and MMBench, exhibit relatively concentrated distributions, indicating that many samples harbor moderate visual redundancy. In contrast, detail-sensitive datasets, including ChartQA, OCRBench, TextVQA, and DocVQA, display broader or heavier high-density regions, reflecting the presence of fine-grained textual, structural, or layout information. These samples are highly vulnerable to uniform downsampling or aggressive post-encoder pruning, as small visual elements frequently contain task-critical evidence.

This qualitative evidence complements the quantitative results. A single fixed-resolution policy cannot consistently accommodate the diverse visual characteristics across benchmarks: low-density samples can be aggressively condensed, whereas high-density samples mandate higher pixel budgets to preserve local details. By estimating informa-

tion density before full visual encoding, APC dynamically adjusts the input resolution according to the intrinsic complexity of each image, preserving holistic structures while retaining fine-grained evidence under strict token budgets.

E Future Work

The empirical success of PACE establishes pre-encoder pixel allocation as a promising frontier for efficient VLM inference. Future research can advance this paradigm across three dimensions. First, an autoregressive recovery mechanism could dynamically fetch high-resolution crops when fine-grained evidence is insufficient, unlocking more aggressive compression. Second, pixel allocation can be advanced to a query-aware regime—conditioned on textual prompts rather than purely intrinsic visual redundancy—using denoised frameworks like VTC-Bench (Liao et al., 2025) for precise evaluation. Finally, integrating PACE with localized patching paradigms, such as density-based boundary allocation (Choudhury et al., 2025; Liu et al., 2026), and model-quantization techniques (Lin et al., 2024; Wang et al., 2026a) could further accelerate end-to-end inference by reducing both visual-token and model-weight computation.

F LLM Usage Disclosure Statement

During manuscript preparation, we used LLMs to assist with language editing and debugging experimental code. The authors independently developed the ideas, experimental design, implementation, analysis, and core technical content. We reviewed and verified all LLM-assisted content and take full responsibility for the submission's originality, scientific integrity, and technical accuracy.

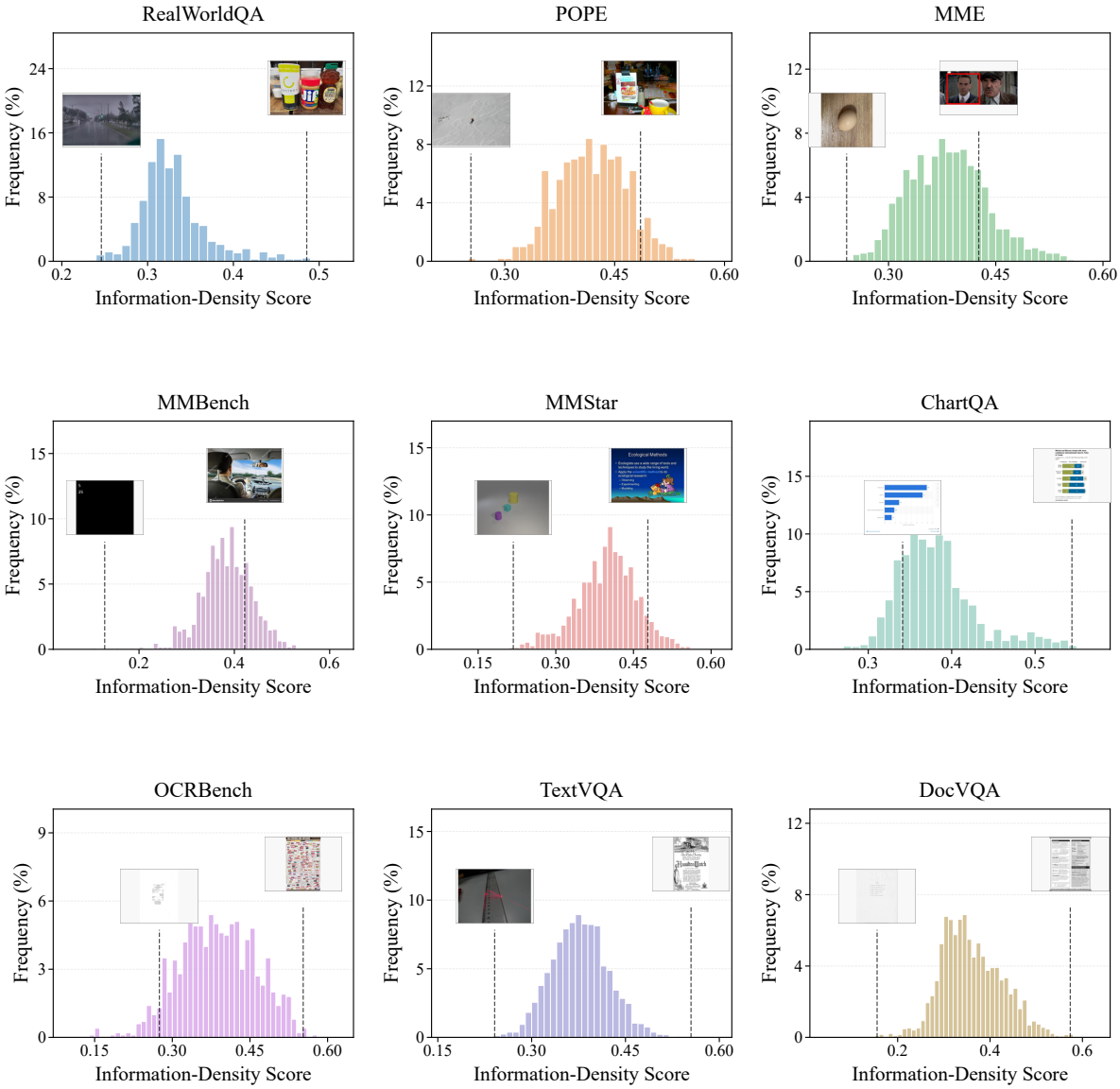

Figure 9: **Information density score distributions across benchmarks.** Histograms and examples contrast APC's information density scores across nine benchmarks.