Patch4Patch: Restoring Structural Connectivity in Patch-based Vision Encoders

Yaqi Zhang[1,*] **Shuntian Yao**[2,*] **Niantai Qu**[2] **Runguo Chen**[1]
Hanyu Lai[2] **Ningxin Su**[3] **Li Sun**[1,†] **Sen Su**[1,†]
[1]Beijing University of Posts and Telecommunications
[2]Tsinghua University
[3]The Hong Kong University of Science and Technology (Guangzhou)
yaqizhang2003@gmail.com

Abstract

As Multimodal Large Language Models (MLLMs) continue to advance and find increasingly broad applications, a key challenge persists: the patch-based vision paradigms widely adopted by MLLMs inevitably introduce structural fragmentation when processing information-dense images such as tables and charts. Specifically, when a table row or chart axis spans multiple visual partitions, the lack of cross-partition communication causes the model to lose track of structural continuity, leading to performance fluctuations when layout changes — a problem we term layout sensitivity. To address this fundamental limitation, we propose Patch-for-Patch (P4P), a lightweight module that can be plugged into existing patch-based vision encoders. P4P dynamically selects a sparse set of tokens as bridging anchors and uses them to establish cross-partition communication through bidirectional attention. Combined with a gated fusion strategy, P4P restores global structural coherence without the quadratic cost of global attention. Extensive experiments across Qwen2.5-VL, MiniCPM-V, and InternVL3.5 demonstrate that P4P consistently improves structured content visual understanding, notably boosting TableEval accuracy by up to 9.19% and significantly reducing layout sensitivity.

1 Introduction

Recently, MLLMs have demonstrated remarkable potential in natural language understanding and complex visual perception tasks [1, 2, 3, 4, 5, 6]. As the field continues to advance, the research frontier of MLLMs has transitioned from basic image-text alignment to parsing ultra-long contexts[7], information-dense images[8], and complex structured documents [9]. One of the primary drivers of this progress is the evolution of the Vision Transformer (ViT) architecture [10], which maps images into serialized token spaces, enabling sequential modeling of visual patterns.

However, as visual encoders scale to higher resolutions for finer-grained perception, a fundamental challenge arises: the quadratic growth of self-attention cost. To balance efficiency and precision, mainstream visual encoders have widely adopted spatial decomposition paradigms. While their implementation details vary, the underlying principle remains consistent: decomposing information-dense images into discrete sub-regions (which we collectively refer to as *partitions*) to reduce the complexity of self-attention. Specifically, Qwen2.5-VL [11] employs window attention to restrict interactions within local windows; MiniCPM-V [12] utilizes a slicing mechanism to decompose large images into sub-images [13]; and InternVL3.5 [14] leverages a dynamic tiling strategy to handle multi-scale inputs.

[*]Equal contribution. [†]Corresponding authors.

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

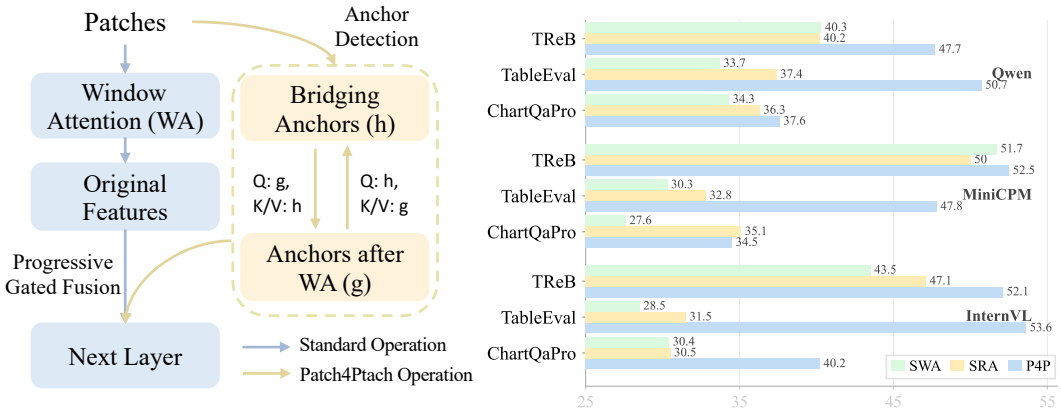


Figure 1: **Overview of P4P.** (a) The P4P framework operates in three steps: Anchor Detection identifies structurally important tokens as bridging anchors $\mathbf{h}$; Bidirectional Cross-Attention enables two-way communication between anchors $\mathbf{h}$ and anchors after WA $\mathbf{g}$; Progressive Gated Fusion controls how much cross-partition information is injected back into the original feature map. (b) Accuracy (%) comparison across three architectures and benchmarks against cross-partition baselines Shifted Window Attention (SWA) and SliceRestore Adapter (SRA).

Despite the success of these partition-based schemes in general visual tasks, they share a common limitation: tokens in different partitions cannot attend to each other. For most natural images, this is acceptable because local partitions context is often sufficient. However, for structured visual elements like tables, this becomes problematic. A single table row may be split across two or more partitions, meaning the partition-based vision encoder has no mechanism to associate cells that logically belong together. We refer to this problem as *structural fragmentation*.

To quantify this effect, we conduct a controlled experiment (Section 2) where we render the same table content with different spatial layouts — varying only the cell padding and font size while keeping the logical content identical. We observe that all three evaluated models exhibit performance fluctuations, despite processing semantically identical inputs. We call this phenomenon *layout sensitivity*: the model's performance is sensitive to the layout of visual elements, rather than the underlying content. This finding motivates the design of a mechanism that can restore cross-partition communication.

To address this limitation, we propose the Patch-for-Patch (P4P) mechanism, designed to restore structural coherence within fragmented visual partitions. P4P works by selecting a sparse set of structurally important tokens (bridging anchors), and then enabling these anchors to exchange information across partitions via lightweight bidirectional attention. A gated fusion mechanism controls how much cross-partition information is injected back into the original feature map, preserving pretrained knowledge while gradually incorporating global structural cues. Crucially, since only a small subset of tokens participates in cross-partition attention, P4P avoids the quadratic cost of full global attention.

Notably, P4P exhibits robust architectural adaptability. We integrate P4P into three mainstream architectures: the window-attention-based Qwen, the slicing-based MiniCPM, and the dynamic tiling-based InternVL. Empirical results demonstrate that P4P achieves consistent improvements in complex table parsing, effectively reduces layout sensitivity, and also generalizes to chart understanding tasks with complex visual dependencies.

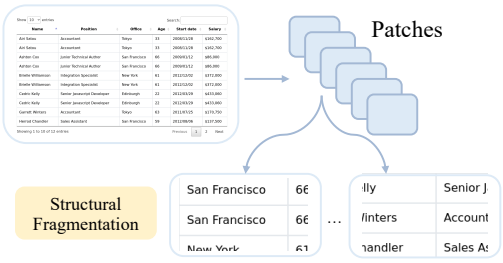
Our primary contributions are summarized as follows:

- **Analysis of Paradigm Limitations:** We identify and characterize structural fragmentation in current patch-based MLLMs, showing that layout variations alone can cause performance fluctuations on table understanding tasks.
- **The P4P Mechanism:** We propose P4P, a lightweight and modular mechanism that selects bridging anchor tokens and uses bidirectional attention with gated fusion to restore cross-partition communication, without requiring full global attention.

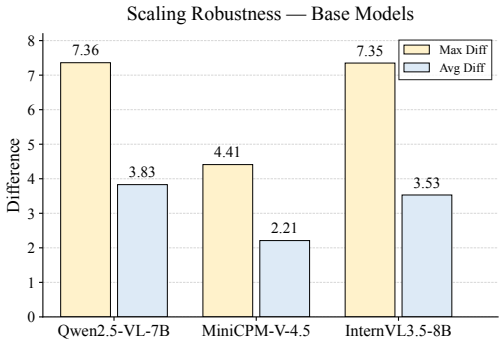
- **Cross-Architecture Validation:** We successfully integrate P4P into three dominant visual paradigms (Qwen, MiniCPM, and InternVL), demonstrating the generalizability and robustness of our method across different partition strategies.

2 Preliminary observations

To quantify the impact of current visual partitioning paradigms on the stability of structured content understanding, we conduct a controlled preliminary experiment. We evaluate three representative MLLMs on tables with identical logical content but different spatial densities. Specifically, we render each table at five density levels by varying cell padding and font size, ranging from Compact (minimal padding, smaller font) through Normal (default rendering) to Relaxed (generous padding, larger font). This changes only the physical layout while preserving the table's logical structure. Detailed rendering parameters are provided in Appendix B.



(a) Illustration of structural fragmentation. A table is decomposed into patches for processing. When a row spans multiple partitions, cells that logically belong together are split into separate, non-communicating partitions, severing structural continuity.



(b) Results of the preliminary experiment. Performance fluctuates substantially across layout densities despite identical logical content.

Figure 2: Structural fragmentation and layout sensitivity.

As shown in Figure 2b, we use two key metrics to measure the model's sensitivity to layout changes: max performance gap and average performance gap. Let p_i denote the model's performance on the same evaluation set under layout i, and let $L = 5$ be the number of layouts. We compute the maximum and average absolute differences over all unordered layout pairs:

$$\text{MaxDiff} = \max_{1 \le i < j \le L} |p_i - p_j|, \qquad \text{AvgDiff} = \frac{2}{L(L-1)} \sum_{i<j} |p_i - p_j|. \tag{1}$$

Lower values indicate stronger layout invariance.

All evaluated models exhibit notable performance fluctuations despite processing logically identical content. Qwen2.5-VL and InternVL3.5 are the most sensitive, with maximum performance gaps of 7.36 and 7.35 points, respectively. MiniCPM-V shows comparatively lower sensitivity, but still shows significant performance fluctuations. These results indicate that current visual encoders are sensitive to layout variations.

3 P4P framework

To address the structural fragmentation and layout sensitivity observed in Section 2, we propose the P4P framework. Rather than replacing existing efficient partitioning paradigms, P4P establishes dynamic bridging anchors that restore cross-partition communication with minimal computational overhead.

3.1 Problem formulation

Before formalizing the problem, we clarify the terminology used throughout this paper. We use the term *partition* as a general abstraction for the discrete sub-regions produced by different visual

encoders. In Qwen2.5-VL, a partition corresponds to an attention window; in MiniCPM-V, it corresponds to an image slice; and in InternVL, it corresponds to a dynamic tile. Despite differences in implementation, all three approaches share the same core property: attention is computed independently within each partition, with no cross-partition token interaction.

In modern MLLMs, to circumvent the $\mathcal{O}(L^2)$ complexity of global self-attention over L total visual tokens [15], the image feature map $\mathbf{X} \in \mathbb{R}^{H \times W \times C}$ (height H, width W, channel dimension C) is typically mapped by a partitioning operator $\mathcal{P}$ into N discrete visual partitions $\{\mathbf{P}_1, \mathbf{P}_2, \ldots, \mathbf{P}_N\}$:

$$\{\mathbf{P}_i\}_{i=1}^{N} = \mathcal{P}(\mathbf{X}), \quad \text{where } \mathbf{P}_i \in \mathbb{R}^{M \times C} \tag{2}$$

Here each partition contains M tokens. The operator $\mathcal{P}$ manifests as non-overlapping windows in Qwen2.5-VL [16, 11], image slices in MiniCPM-V [12], or dynamic tiles in InternVL [14].

Attention is then computed independently within each partition, producing tokens $\mathbf{Y}_i = \text{Attn}(\mathbf{Q}_i, \mathbf{K}_i, \mathbf{V}_i)$. This imposes a hard constraint on the receptive field: cross-partition structural dependencies are severed. For structures that span multiple partitions, the correlation between tokens in $\mathbf{P}_i$ and $\mathbf{P}_j$ is physically truncated. Consequently, the model cannot track structures that span multiple partitions, leading to the layout sensitivity observed in Section 2.

3.2 Bridging anchor detection

To restore global connectivity without introducing full-attention overhead, P4P first employs a lightweight detector $\mathcal{D}$ to identify bridging anchors within each partition. These anchors are tokens that are most likely to carry information relevant to cross-partition structures.

Given a token feature $\mathbf{h} \in \mathbb{R}^D$, we compute its structural saliency score S:

$$S = \text{Sigmoid}(\mathcal{F}_{det}(\mathbf{h})) \tag{3}$$

Depending on the model capacity and feature depth, the mapping function $\mathcal{F}_{det}$ is instantiated as either a lightweight two-layer MLP (for Qwen and MiniCPM) or a computationally efficient linear projection (for InternVL).

The detector is trained jointly with the model using only the standard autoregressive language-modeling loss, without anchor annotations or detector-specific auxiliary losses. Although hard Top-K indices are non-differentiable, the selected detector scores are retained as continuous fusion weights, allowing the final objective to update $\mathcal{D}$.

To ensure cross-architecture compatibility, we adopt a dual-selection strategy:

- **Spatial Prior Selection** (for window-based architectures): In models with explicit physical boundaries like Qwen [11], we combine a boundary mask $\mathbf{M}_{mask}$ to select anchors specifically at window junctions.
- **Saliency Top-K Selection** (for slice/tile architectures): In models lacking explicit local boundaries like MiniCPM [12] or InternVL [14], we directly select the top-K tokens with the highest saliency scores globally to serve as bridges across partitions.

3.3 Bidirectional attention

The shared P4P mechanism selects anchors, exchanges information across partitions, and progressively fuses the resulting features. The implementations below adapt this mechanism to the native partition organization of each backbone. We use two communication patterns depending on the architecture:

Bidirectional Cross-attention (Qwen & MiniCPM). We maintain two sets of features for the anchors: the *local* features $\mathbf{h}_{local}$ (from the input to the attention layer) and a *global summary* $\mathbf{g}$ (from the attention output). Information flows in two steps:

The global summary queries the local anchors to gather structural context:

$$\mathbf{g}' = \text{CrossAttn}(\mathbf{Q}{=}\mathbf{g},\ \mathbf{K}{=}\mathbf{h}_{local},\ \mathbf{V}{=}\mathbf{h}_{local}) + \mathbf{g} \tag{4}$$

The enriched summary is then broadcast back to enhance the anchor tokens:

$$\mathbf{h}_{enh} = \text{CrossAttn}(\mathbf{Q}{=}\mathbf{h}_{local},\ \mathbf{K}{=}\mathbf{g}',\ \mathbf{V}{=}\mathbf{g}') + \mathbf{h}_{local} \tag{5}$$

Algorithm 1 P4P forward pass (Qwen2.5-VL)

Require: Hidden states $\mathbf{X}$, window indices $\mathbf{I}$, cumulative window lengths $\mathbf{C}$
Ensure: Enhanced features $\mathbf{Y}$

$\mathbf{Y}_{\text{win}} \leftarrow \text{WindowAttention}(\mathbf{X}, \mathbf{C})$
% Step 1: Anchor Selection
$\mathbf{s} \leftarrow \sigma(\text{BoundaryDetector}(\mathbf{X}))$ {saliency scores, $\mathbb{R}^{N\times 1}$}
$\mathbf{B} \leftarrow \text{BoundaryCandidates}(\mathbf{C}, \text{margin} = 2)$ {tokens near window edges}
if $|\mathbf{B}| = 0$ **then**
  return $\mathbf{Y}_{\text{win}}$
end if
$\mathcal{A} \leftarrow \text{TopK}(\mathbf{s}[\mathbf{B}],\ k = \max(1, \lfloor 0.15 \cdot |\mathbf{B}| \rfloor))$ {sparse anchor set}
$\mathbf{h}_{\text{local}} \leftarrow \mathbf{X}[\mathcal{A}], \quad \mathbf{g} \leftarrow \mathbf{Y}_{\text{win}}[\mathcal{A}]$
% Step 2: Bidirectional Cross-Attention
$\mathbf{g}' \leftarrow \text{CrossAttn}\big(\text{Q}{=}\text{Norm}(\mathbf{g}),\ \text{K}{=}\text{Norm}(\mathbf{h}_{\text{local}}),\ \text{V}{=}\mathbf{h}_{\text{local}}\big) + \mathbf{g}$
$\mathbf{h}_{\text{enh}} \leftarrow \text{CrossAttn}\big(\text{Q}{=}\text{Norm}(\mathbf{h}_{\text{local}}),\ \text{K}{=}\text{Norm}(\mathbf{g}'),\ \text{V}{=}\mathbf{g}'\big) + \mathbf{h}_{\text{local}}$
% Step 3: Gated Fusion
$\alpha \leftarrow \sigma\big(\text{MLP}([\mathbf{g};\ \mathbf{h}_{\text{enh}}])\big)$ {per-token gate, initialized near zero}
$\mathbf{w} \leftarrow \mathbf{s}[\mathcal{A}]$ {selected sigmoid scores; gradients retained}
$\mathbf{Y}_{\text{win}}[\mathcal{A}] \leftarrow \mathbf{Y}_{\text{win}}[\mathcal{A}] + \mathbf{w} \odot \alpha \odot (\mathbf{h}_{\text{enh}} - \mathbf{Y}_{\text{win}}[\mathcal{A}])$
return $\mathbf{Y}_{\text{win}}$

In Qwen, $\mathbf{g}$ is the window-attention output at anchor positions; in MiniCPM, $\mathbf{g}$ is the mean-pooled output across all tokens (providing a holistic slice-level view). Both cross-attention modules use pre-norm and zero-initialized output projections for training stability.

Peer-to-peer Self-attention (InternVL). Since InternVL processes each tile independently through the vision encoder without producing a shared global representation, we adopt peer-to-peer self-attention among anchors instead of hub-based cross-attention, and use a shared scalar gate for simplicity. This design avoids introducing an artificial global summary for architectures that lack one natively. All anchor tokens from every tile are concatenated into a single sequence $\mathbf{H}_{struct}$ and processed with one self-attention pass:

$$\mathbf{h}_{enh} = \text{SelfAttn}(\mathbf{H}_{struct}) + \mathbf{h}_{local} \tag{6}$$

This lets every anchor directly attend to every other anchor across all tiles, with the self-attention module using a fused QKV projection and zero-initialized output projection.

3.4 Gated fusion

Finally, the enhanced anchor features are merged back into the original output via a learned gate $\alpha \in [0, 1]$ and the selected detector score S:

$$\mathbf{y}_{fused} = \mathbf{y}_{orig} + S \cdot \alpha \odot (\mathbf{h}_{enh} - \mathbf{y}_{orig}) \tag{7}$$

For Qwen and MiniCPM, α is content-aware — an MLP takes the concatenation of the original and enhanced features: $\alpha = \sigma\big(\text{MLP}([\mathbf{y}_{orig}; \mathbf{h}_{enh}])\big)$ For InternVL, α is a single learnable scalar: $\alpha = \sigma(\beta)$. In all cases, α is initialized near zero so that P4P starts as an identity operation, preserving pre-trained representations. Non-anchor tokens are left untouched. Here S is the sigmoid score of a selected anchor, broadcast across channels, and $\odot$ denotes element-wise multiplication. Retaining S in this residual update provides a differentiable path to the detector even though the selection indices are discrete.

For clarity, Algorithm 1 details the implementation specifically for the Qwen architecture as a representative instance.

4 Experiments

In this section, we evaluate the effectiveness of the P4P mechanism across various mainstream MLLM architectures. We focus on table understanding and chart reasoning tasks to demonstrate the effectiveness of our approach in mitigating structural fragmentation.

4.1 Experimental setup

We implement P4P as a modular architectural extension for Qwen2.5-VL-7B [11], MiniCPM-V-4.5 [12], and InternVL3.5-8B [14], with full-parameter fine-tuning. We evaluate on TReB [17], ChartQAPro [18], and TableEval [19]. All three benchmarks were released after June 2025, making them as recent as possible relative to the publicly available training-data cutoffs of the evaluated base models, thereby minimizing the risk of data leakage. Benchmark descriptions and training details are provided in Appendix E and D.

4.2 Main results

The quantitative results of our experiments are summarized in Table 1. Across all three evaluated architectures, the P4P mechanism consistently outperforms both the Base models and standard Direct SFT baselines. Crucially, we also compare against two alternative cross-partition communication strategies: Shifted Window Attention (SWA) [20] and the SliceRestore Adapter (SRA) [21]. Details for SWA and SRA are provided in Appendix F.

Comparison with Direct SFT. As shown in Table 1, P4P consistently outperforms Direct SFT across all models and benchmarks. On TReB, Qwen2.5-VL improves from 46.00% to 47.70% and MiniCPM-V from 50.74% to 52.48%; on TableEval, InternVL3.5-8B improves from 50.48% to 53.61%. Since all P4P and SFT models are trained on identical data, these gains confirm that the bottleneck lies not in domain knowledge but in tracking structure across partitions. The consistent improvement on TableEval's diverse Chinese and English tables further suggests that P4P's effectiveness cross different languages.

Cross-partition baselines cause degradation. A striking finding is that both SWA and SRA consistently degrade TableEval performance below the Base model across all three architectures. For InternVL, SWA drops from 44.49% (Base) to 31.48% (-13.01%), and SRA drops to 28.48% (-16.01%). On TReB, both methods also underperform Direct SFT: for Qwen, SWA scores 40.33% and SRA scores 40.21%, both well below Direct SFT's 46.00%.

Under necessary adaptations, SWA and SRA consistently underperform P4P, suggesting that unconditional cross-partition mixing may disrupt pretrained local representations.

In contrast, P4P takes a soft additive approach. The original attention computation is left entirely unchanged. P4P adds a parallel bridging pathway that operates on a sparse subset of boundary tokens, and the fusion gate controls how much of this information is fused back. This design ensures that P4P can only add cross-partition information where it helps, without degrading existing local representations.

4.3 Generalization to chart understanding

To assess generalization beyond tables, we evaluate on ChartQAPro (Table 1). P4P improves performance across all three MLLM backbones, while SWA and SRA show mixed or degraded results. A granular analysis reveals that the largest gains appear in reasoning-heavy categories such as Hypothetical and Conversational, which require the model to reason about spatial relationships across chart regions. For example, Qwen2.5-VL's Hypothetical score rises from 37.80% (SFT) to 42.88% (P4P), whereas SRA on InternVL drops to 21.43% (vs. SFT's 44.85%), suggesting that hard attention modifications actively disrupt multi-step reasoning pathways. In contrast, recognition-oriented tasks like Fact Checking show more modest gains, as they rely primarily on local OCR precision already well-served by high-resolution partitioning. This pattern confirms that P4P's primary benefit is restoring cross-partition structural coherence for reasoning.

4.4 Layout robustness analysis

Section 2 revealed that current MLLMs are sensitive to layout variations in logically identical tables. We now test whether P4P mitigates this. Following the same protocol, we render 500 TReB samples at densities ranging from Compact to Relaxed and measure performance fluctuation. As shown in Figure 3, Direct SFT does not reduce layout sensitivity — for Qwen and InternVL, max performance fluctuation remains unchanged after fine-tuning, confirming that domain knowledge alone cannot

Table 1: **Comprehensive results.** Table Task and ChartQAPro (ood) report accuracy (%).

Model	Method	Table Task		ChartQAPro (ood)				
		TReB	TableEval	Factoid	Conv.	Hypoth.	FC	MCQ
Qwen2.5-VL-7B	Base	34.60	42.07	24.15	31.76	26.51	**46.31**	36.92
	Direct SFT	46.00	41.52	24.98	41.03	37.80	40.16	**38.79**
	SWA	40.33	33.66	**28.71**	40.90	37.87	28.28	35.98
	SRA	40.21	37.42	26.49	41.52	40.15	41.31	32.12
	P4P (Ours)	**47.70**	**50.71**	27.38	**42.48**	**42.88**	40.98	34.11
MiniCPM-V-4.5	Base	49.10	47.36	16.99	26.85	18.73	**45.90**	26.64
	Direct SFT	50.74	44.65	24.78	39.06	32.50	35.66	39.25
	SWA	51.74	30.26	15.35	18.16	30.13	43.44	30.84
	SRA	49.96	32.83	25.76	37.61	33.19	40.98	37.85
	P4P (Ours)	**52.48**	**47.84**	**26.44**	**39.43**	**34.00**	32.38	**40.19**
InternVL3.5-8B	Base	48.28	44.49	20.66	30.74	28.03	20.49	**43.46**
	Direct SFT	52.00	50.48	25.27	37.32	**44.85**	45.49	40.19
	SWA	43.50	31.48	18.23	29.18	25.95	43.44	35.05
	SRA	47.12	28.48	21.86	33.18	21.43	43.85	32.24
	P4P (Ours)	**52.10**	**53.61**	**25.88**	**40.07**	44.81	**47.95**	42.06

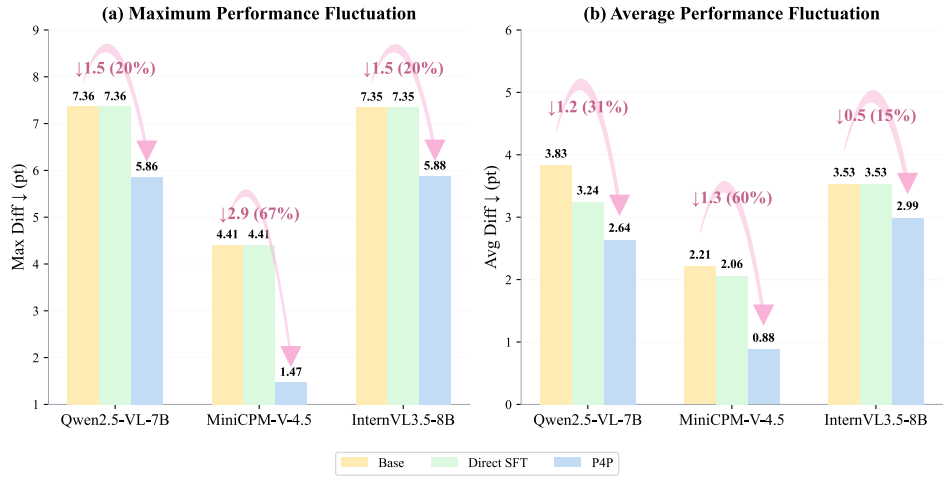


Figure 3: **Layout robustness analysis.** Performance fluctuation (Accuracy %) across layout densities (Compact to Relaxed) on 500 sampled TReB cases. Max Diff and Avg Diff measure the maximum and the average absolute performance difference between all pairs of layouts — lower values indicate stronger layout invariance.

address structural fragmentation. P4P reduces layout fluctuations substantially while also improving absolute performance. The strongest result is on MiniCPM-V: max performance fluctuation drops from 4.41 to 1.47 and average performance fluctuation falls below 0.9, indicating that P4P largely rectifies the fragmentation caused by sub-image slicing. These results confirm that P4P's gains stem not from domain-specific knowledge but from restoring cross-partition structural awareness.

4.5 General multimodal performance and inference cost

To evaluate behavior beyond tables and charts, we test on MMBench v1.1 [22], MMStar [23], and MME [24]. Compared with Direct SFT, P4P largely preserves general multimodal performance across the three backbones, with modest improvements on several metrics. After table-task fine-

Table 2: **Ablation study.** All experiments use Qwen2.5-VL-7B. Table Task and ChartQAPro (ood) report accuracy (%). Speed is measured on 8×RTX 4090 GPUs.

	Table Task		ChartQAPro (ood)					Efficiency
Method	**TReB**	**TableEval**	**Factoid**	**Conv.**	**Hypoth.**	**FC**	**MCQ**	**Speed (it/s)**
w/o Bidir. Attn	43.33	46.03	24.06	42.38	38.00	**41.44**	33.92	0.0389
w/o Gated Fusion	45.50	44.01	25.61	42.02	41.77	39.34	32.12	**0.0486**
P4P (Ours)	**47.70**	**50.71**	**27.38**	**42.48**	**42.88**	40.98	**34.11**	0.0457

tuning, both Direct SFT and P4P show small performance drops on some general multimodal benchmark metrics relative to the Base models. Full benchmark results and efficiency measurements are provided in Appendix I.

5 Ablation study

To verify the necessity of each component in the P4P mechanism, we conducted extensive ablation experiments using Qwen2.5-VL-7B as the backbone. We focus on two core designs: the bidirectional attention and the gated fusion.

- **Ablation A (w/o Bidirectional Attention):** We replace the bidirectional cross-attention with a standard self-attention mechanism for bridging anchors. In this setup, local and global tokens are concatenated and interact without the explicit bidirectional cross attention.
- **Ablation B (w/o Gated Fusion):** We replace the content-aware MLP gate (Section 3.4) with a fixed fusion coefficient, removing the model's ability to dynamically modulate the injection strength of cross-partition information. The bidirectional attention and anchor detection remain unchanged.

Bidirectional attention matters for structural reasoning. As shown in Table 2, removing bidirectional attention causes the largest drop on TReB, confirming that explicit bidirectional routing is critical for reconstructing cross-partition table structures. The ChartQAPro breakdown (Table 2) further shows that P4P's gains concentrate on Factoid and Hypothetical sub-tasks, which require tracing data across chart regions, while the self-attention variant performs comparably on Fact Check tasks that rely more on local text matching.

Gated fusion enables stable integration. Removing the content-aware gate degrades TReB by 2.20% and TableEval by 6.70%. The fixed α forces equal mixing regardless of content, injecting noise at positions where the structural signal is weak. The fusion gate allows P4P to start as an identity operation and progressively learn where cross-partition information helps.

Computational overhead. Table 2 compares training efficiency on $8\times$ RTX 4090. P4P is 17.5% faster than the self-attention variant (w/o Bidir. Attn), since the bidirectional cross-attention avoids the quadratic cost of concatenating all anchor and global tokens. Compared to the static-gate variant, P4P is marginally slower ($\sim$6%) due to the MLP gate computation, a modest cost for the accuracy gains.

Anchor visualization To complement the quantitative ablations above, we visualize the spatial distribution of selected bridging anchors at three network depths across four representative inputs: a composite chart, a multi-category table, a stacked area chart, and a scientific data table (Figure 4).

Two key observations emerge. **(1) Progressive boundary alignment:** In early layers, anchors are scattered without clear structural preference. As depth increases, they converge onto structurally salient positions—row/column dividers in tables, curve boundaries in charts, and sub-figure borders in composite images—forming increasingly precise patterns by the deepest layer. **(2) Content-adaptive anchor allocation:** The distribution pattern varies qualitatively by input type. For tabular

inputs, anchors progressively form grid-like patterns aligned with row and column separators, with highest density at category headers and information-dense cells (e.g., scientific notation). For continuous charts, anchors instead concentrate along curve boundaries and inflection points rather than forming regular grids. This divergence confirms that the anchor detector is genuinely content-aware rather than applying a fixed spatial heuristic.



Figure 4: **Anchor visualization across network depths.** Cyan dots mark selected bridging anchors. From left to right: input image, early-layer, mid-layer, and deep-layer anchor distributions. Anchors progressively align with structural boundaries as depth increases.

6 Related work

Spatial decomposition in vision encoders To handle high-resolution visual inputs efficiently, mainstream MLLMs have transitioned from fixed-size encoding [1] to spatial decomposition paradigms. These include window-based attention [11], image slicing [12], and dynamic tiling [14]. However, these partitioning strategies introduce structural fragmentation, where visual entities are severed across non-communicating partitions. Unlike resolution or sequence-length challenges that have received substantial attention[25], the structural coherence problem across partitions remains largely overlooked in existing work. While approaches such as global context branches [14] attempt to partially mitigate boundary discontinuities, they provide only coarse or local remedies and are often insufficient to recover fine-grained structural coherence for elements that span multiple partitions.

Cross-partition communication mechanisms Restoring connectivity across discrete visual partitions is a nascent but critical challenge. In general vision Transformers, Shifted Window attention [20] enables cross-window communication via alternating window layouts, but is limited to fixed-stride local shifts and cannot adapt to semantic boundaries. Most recently, HiRes-LLaVA [21] proposes the SliceRestore Adapter (SRA), which merges sliced patches back into their original spatial layout and applies convolution and attention across all tokens to restore cross-partition context.

Under the adaptations evaluated in Section 4.2, SWA and SRA underperform P4P. This is consistent with possible disruption of local representations by unconditional mixing. In the context of MLLMs, other solutions primarily rely on global context branches [14] or partition-level spatial schemas [12] to partially preserve inter-partition context.

7 Conclusion

In this paper, we have presented P4P, a lightweight mechanism designed to mitigate structural fragmentation in modern patch-based MLLMs. Under the evaluated adaptations, Shifted Window Attention and the SliceRestore Adapter underperform P4P, suggesting that unconditional cross-partition mixing may disrupt pretrained local representations. In contrast, P4P's soft additive design, which selects sparse bridging anchors and controls fusion through adaptive gating, restores cross-partition communication without disrupting pre-trained knowledge. Experimental results across multiple state-of-the-art architectures demonstrate that P4P not only improves performance on information-dense structures like tables but also generalizes to chart understanding tasks with complex visual dependencies. Our results suggest that efficient spatial decomposition and cross-partition communication are complementary rather than conflicting goals, and that lightweight mechanisms like P4P can meaningfully improve structural understanding without sacrificing the efficiency benefits of partition-based processing.

Acknowledgements and disclosures

Funding. This work was supported by the National Natural Science Foundation of China under Grant No. 62072052.

Competing interests. The authors declare no competing interests.

References

[1] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language supervision. In *International conference on machine learning*, pages 8748–8763. PmLR, 2021.

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

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

[4] Shukang Yin, Chaoyou Fu, Sirui Zhao, Ke Li, Xing Sun, Tong Xu, and Enhong Chen. A survey on multimodal large language models. *National Science Review*, 11(12):nwae403, 2024.

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

[6] Zichun Guo, Yuling Shi, Wenhao Zeng, Chao Hu, Haotian Lin, Terry Yue Zhuo, Jiawei Chen, Xiaodong Gu, and Wenping Ma. Shredbench: Evaluating the semantic reasoning capabilities of multimodal llms in document reconstruction. *arXiv preprint arXiv:2604.23813*, 2026.

[7] Runze Cui, Fangxin Shang, Yehui Yang, Qing Yang, and Tao Chen. Fcmbench-video: Benchmarking document video intelligence. *arXiv preprint arXiv:2604.25186*, 2026.

[8] Zhihang Yi, Jian Zhao, Jiancheng Lv, and Tao Wang. Multimodal information fusion for chart understanding: A survey of mllms–evolution, limitations, and cognitive enhancement. *arXiv preprint arXiv:2602.10138*, 2026.

[9] Beichen Zhang, Pan Zhang, Xiaoyi Dong, Yuhang Zang, and Jiaqi Wang. Long-clip: Unlocking the long-text capability of clip. In *European conference on computer vision*, pages 310–325. Springer, 2024.

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

[11] Shuai Bai, Keqin Chen, Xuejing Liu, Jialin Wang, Wenbin Ge, Sibo Song, Kai Dang, Peng Wang, Shijie Wang, Jun Tang, et al. Qwen2. 5-vl technical report. *arXiv preprint arXiv:2502.13923*, 2025.

[12] Tianyu Yu, Zefan Wang, Chongyi Wang, Fuwei Huang, Wenshuo Ma, Zhihui He, Tianchi Cai, Weize Chen, Yuxiang Huang, Yuanqian Zhao, et al. Minicpm-v 4.5: Cooking efficient mllms via architecture, data, and training recipe. *arXiv preprint arXiv:2509.18154*, 2025.

[13] Zonghao Guo, Ruyi Xu, Yuan Yao, Junbo Cui, Zanlin Ni, Chunjiang Ge, Tat-Seng Chua, Zhiyuan Liu, and Gao Huang. Llava-uhd: an lmm perceiving any aspect ratio and high-resolution images. In *European Conference on Computer Vision*, pages 390–406. Springer, 2024.

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

[15] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

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

[17] Ce Li, Xiaofan Liu, Zhiyan Song, Ce Chi, Chen Zhao, Jingjing Yang, Zhendong Wang, Kexin Yang, Boshen Shi, Xing Wang, et al. Treb: A comprehensive benchmark for evaluating table reasoning capabilities of large language models. *arXiv preprint arXiv:2506.18421*, 2025.

[18] Ahmed Masry, Mohammed Saidul Islam, Mahir Ahmed, Aayush Bajaj, Firoz Kabir, Aaryaman Kartha, Md Tahmid Rahman Laskar, Mizanur Rahman, Shadikur Rahman, Mehrad Shahmohammadi, et al. Chartqapro: A more diverse and challenging benchmark for chart question answering. *arXiv preprint arXiv:2504.05506*, 2025.

[19] Junnan Zhu, Jingyi Wang, Bohan Yu, Xiaoyu Wu, Junbo Li, Lei Wang, and Nan Xu. Tableeval: A real-world benchmark for complex, multilingual, and multi-structured table question answering. *arXiv preprint arXiv:2506.03949*, 2025.

[20] Ze Liu, Yutong Lin, Yue Cao, Han Hu, Yixuan Wei, Zheng Zhang, Stephen Lin, and Baining Guo. Swin transformer: Hierarchical vision transformer using shifted windows. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 10012–10022, 2021.

[21] Runhui Huang, Xinpeng Ding, Chunwei Wang, Jianhua Han, Yulong Liu, Hengshuang Zhao, Hang Xu, Lu Hou, Wei Zhang, and Xiaodan Liang. Hires-llava: Restoring fragmentation input in high-resolution large vision-language models. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pages 29814–29824, 2025.

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

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

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

[25] Daniel Bolya, Cheng-Yang Fu, Xiaoliang Dai, Peizhao Zhang, Christoph Feichtenhofer, and Judy Hoffman. Token merging: Your ViT but faster. *arXiv preprint arXiv:2210.09461*, 2023.

[26] Boammani Aser Lompo and Marc Haraoui. Visual-tableqa: Open-domain benchmark for reasoning over table images. *arXiv preprint arXiv:2509.07966*, 2025.

[27] Tom Agonnoude and Cyrile Delestre. cmarkea/table-vqa. Hugging Face dataset, 2024. URL `https://huggingface.co/datasets/cmarkea/table-vqa`.

[28] Xiang Yue, Yuansheng Ni, Kai Zhang, Tianyu Zheng, Ruoqi Liu, Ge Zhang, Samuel Stevens, Dongfu Jiang, Weiming Ren, Yuxuan Sun, et al. Mmmu: A massive multi-discipline multimodal understanding and reasoning benchmark for expert agi. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9556–9567, 2024.

[29] Xiang Yue, Tianyu Zheng, Yuansheng Ni, Yubo Wang, Kai Zhang, Shengbang Tong, Yuxuan Sun, Botao Yu, Ge Zhang, Huan Sun, et al. Mmmu-pro: A more robust multi-discipline multimodal understanding benchmark. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 15134–15186, 2025.

[30] Panupong Pasupat and Percy Liang. Compositional semantic parsing on semi-structured tables. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 1470–1480, 2015.

[31] Yuze Zhao, Jintao Huang, Jinghan Hu, Xingjun Wang, Yunlin Mao, Daoze Zhang, Zeyinzi Jiang, Zhikai Wu, Baole Ai, Ang Wang, et al. Swift: a scalable lightweight infrastructure for fine-tuning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 39, pages 29733–29735, 2025.

[32] Tri Dao. Flashattention-2: Faster attention with better parallelism and work partitioning. *arXiv preprint arXiv:2307.08691*, 2023.

[33] Yuliang Liu, Biao Yang, Qiang Liu, Zhang Li, Zhiyin Ma, Shuo Zhang, and Xiang Bai. Textmonkey: An ocr-free large multimodal model for understanding document. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2026.

A Limitations

Incorporating P4P into vision encoder pre-training, evaluating vision-only backbones, and systematically scaling the visual backbone remain unexplored. We have not evaluated P4P on languages with very special layout conventions, such as right-to-left scripts (e.g., Arabic). Our training data is limited to publicly available datasets, training on larger or more diverse corpora remains unexplored.

B Layout rendering details

B.1 Data source

Our preliminary experiment uses tables from the TReB benchmark [17]. We randomly select 500 tables. All tables are rendered from their Markdown source using a custom Python renderer built on Matplotlib, with a fixed image width of 1800 px and DPI of 100.

B.2 Layout density levels

We define five layout density levels—Compact, Semi-Compact, Normal, Semi-Relaxed, and Relaxed—by jointly varying font size (11–16 pt), cell padding (0.03–0.08), row height ($0.85\times$–$1.3\times$), and column width ($0.8\times$–$1.4\times$) relative to the Normal baseline, while keeping table content identical.

To quantify layout compression, we define a Normalized Density Score (NDS):

$$\mathrm{NDS}(l) = \frac{\mathrm{FontSize}(n) \times \mathrm{Padding}(n) \times \mathrm{RowH}(n) \times \mathrm{ColW}(n)}{\mathrm{FontSize}(l) \times \mathrm{Padding}(l) \times \mathrm{RowH}(l) \times \mathrm{ColW}(l)} \tag{8}$$

where n denotes the Normal level. The resulting NDS values range from 2.49 (Compact) to 0.30 (Relaxed), spanning an $8.3\times$ density range.

C Data composition

To specifically address structural fragmentation in structured data, we curated a high-quality table-centric training set comprising approximately 60,000 samples. The dataset integrates the following sources:

- **Visual-TableQA** [26]: Features open-domain table images rendered from LaTeX, with QA pairs verified by a multi-model jury. Original high-resolution images are used to preserve visual cues.
- **Table-VQA** [27]: Consists of table images paired with corresponding LaTeX representations. QA pairs are generated using Gemini 1.5 Pro and Claude 3.5 Sonnet to ensure linguistic diversity.
- **MMMU (Table Subset)** [28]: Samples from the MMMU benchmark explicitly labeled as `Tables`, covering university-level reasoning in Business, Science, and Engineering.
- **MMMU-Pro (Table Subset)** [29]: Filtered table-related entries incorporating complex reasoning scenarios that reflect real-world user interactions.
- **WikiTableQuestions (Rendered)** [30]: Structured tables rendered into high-resolution images to evaluate the mapping of visual structures to logical relationships.

D Implementation details

This appendix provides comprehensive details regarding the experimental environment and training configurations to facilitate reproducibility.

D.1 Hardware and framework

All experiments were conducted on a high-performance server equipped with 8 $\times$ NVIDIA GeForce RTX 4090 (24GB) GPUs. We utilized the `ms-swift` fine-tuning framework [31] and full-parameter fine-tuning of 7B–8B parameter models.

D.2 Hyperparameters and optimization

To maintain consistency across diverse architectures, we standardized the training hyperparameters as follows:

- **Learning Rate:** 1×10^{-5} for the LLM backbone, ViT encoder, and Aligner.
- **Training Duration:** 4 epochs.
- **Batch Size:** A global effective batch size of 32 (8 GPUs $\times$ 1 sample/GPU $\times$ 4 gradient accumulation steps).
- **Sequence Length:** A maximum sequence length of 4096 tokens.
- **Precision:** We employed the `bfloat16` data format. Flash-Attention 2 [32] was activated to optimize the memory footprint and accelerate training speed.

D.3 Resolution strategy

During the high-resolution processing phase, the maximum input resolution was capped at $1,003,520$ pixels (approximately 1024×1024 equivalent). For models utilizing dynamic tiling (e.g., InternVL3.5-8B), the tiling strategy was configured to maintain an aspect-ratio-aware decomposition that fits within this total pixel budget.

E Evaluation benchmarks

To verify the efficacy of P4P in resolving structural fragmentation, we evaluate it on three rigorous benchmarks:

- **TReB (Table Reasoning Benchmark)**: To assess vision-based reasoning, original textual tables are rendered into high-resolution images. We evaluate four core sub-tasks within the table-understanding category: *Fact Checking*, *Plausibility Verification*, *Retrieval*, and *Summary*. This benchmark specifically tests the model's ability to maintain logical consistency across partitions.

- **ChartQAPro**: This benchmark evaluates generalization to complex chart understanding. It requires multi-step reasoning and global structural awareness, serving as a key indicator for the model's performance on visual structures that extend beyond standard training distributions.
- **TableEval**: A cross-lingual (Simplified/Traditional Chinese and English) benchmark that encompasses diverse domains such as finance and academic reports. Similar to TReB, tables are rendered into images to evaluate the robustness of multimodal structural parsing across different languages and document styles.

F Baseline implementation details

We implement SWA (Shifted Window Attention) [20, 33] and SRA (SliceRestore Adapter) [21] as cross-partition communication baselines across all three architectures. Both methods were originally designed for specific model architectures; adapting them to Qwen2.5-VL, MiniCPM-V-4.5, and InternVL3.5-8B requires careful engineering while preserving the core mechanisms of each method.

F.1 SWA implementation

Our SWA implementation reproduces all core components of TextMonkey's shifted window attention [33]: (i) dimension reduction via a linear projection, (ii) Swin-style cyclic-shifted window self-attention with learnable relative position bias [20], and (iii) an output projection with zero-initialized weights. Positional information is encoded via learnable absolute position embeddings. All three architectures share the identical `ShiftedWindowBlock` module.

Three adaptations were necessary:

- **Dynamic resolution (all):** TextMonkey uses a fixed 896×896 input (64×64 tokens). We apply bicubic interpolation to the position embedding for dynamic resolutions, following standard ViT practice [10].
- **Layer placement (all):** TextMonkey inserts SWA at specific intervals within its ViT encoder. We insert at the same layers as P4P so that the comparison isolates the *cross-partition mechanism* while holding insertion depth constant.
- **Tile merging (MiniCPM, InternVL):** These models produce separate per-tile feature maps. We merge them into a unified 2D grid before the Swin block, as cyclic shift requires a contiguous spatial layout. InternVL CLS tokens are excluded and receive a zero delta.

The cyclic shift is half the window size, and the MLP expansion ratio is 2, following TextMonkey. The attention bottleneck is specified relative to the target encoder's hidden dimension to scale the added capacity across backbones. The same fixed configuration is used across evaluation benchmarks.

F.2 SRA implementation

For MiniCPM-V-4.5 and InternVL3.5-8B, our SRA faithfully reproduces HiRes-LLaVA's full pipeline [21]: (i) spatial merging into a 2D layout, (ii) 3×3 depthwise convolution for local fusion, (iii) downsampled multi-head self-attention for global fusion with bilinear upsampling, and (iv) residual addition.

Three adaptations were necessary:

- **1D variant (Qwen only):** For Qwen2.5-VL, since no 2D tile grid exists, we implement a 1D variant that applies depthwise Conv1D over the full token sequence, providing local feature mixing without explicit boundary awareness.
- **Grid layout inference (MiniCPM, InternVL):** HiRes-LLaVA knows the slice layout from its pipeline; we infer the 2D grid at runtime from the number of partitions.
- **Zero-initialization (all):** In the original HiRes-LLaVA, SRA is introduced during instruction tuning with the vision encoder frozen; its parameters are randomly initialized but the

residual connection (Eq. 2 in [21]) ensures pretrained features are preserved. Since we train all parameters jointly from the start, we use zero-initialized output projections to guarantee the same property: SRA contributes zero at initialization, so pretrained behavior is exactly preserved before any gradient update. This is strictly safer than random initialization, which would inject noise into pretrained features at the first forward pass.

For MiniCPM and InternVL, the local and global branches operate in parallel on the same reconstructed feature map; their outputs are added before a single zero-initialized output projection. For Qwen, the 1D SRA-style variant uses depthwise sequential mixing and a lightweight bottleneck projection. These are adapted baselines rather than unmodified implementations in their original architectures.

F.3 Controlled training protocol

SWA, SRA, and P4P use the same training data, full-parameter fine-tuning procedure, learning rate, batch size, resolution budget, and evaluation protocol. Each baseline uses a single fixed configuration across all benchmarks. The newly introduced SWA and SRA output projections are zero-initialized to preserve pretrained behavior at initialization. These controls improve comparability.

G Additional cross-partition comparisons

G.1 Overlapping windows and boundary-only communication

We compare P4P with overlapping windows and a boundary-only communication variant on Qwen2.5-VL-7B, using the same training data and training setup. The overlapping-window baseline provides a direct comparison with a simple way of extending local receptive fields. The boundary-only variant restricts the transmitted information to boundary-token content without the partition-level context used by P4P. Table 3 reports all seven evaluation metrics.

Table 3: Additional comparisons on Qwen2.5-VL-7B. All values are accuracy (%). Factoid, Conv., Hypoth., FC, and MCQ denote the five ChartQAPro categories.

Method	TReB	TableEval	Factoid	Conv.	Hypoth.	FC	MCQ
Overlapping windows	38.22	33.23	26.49	31.75	29.22	39.26	31.20
Boundary-only	40.01	34.13	30.36	30.87	36.59	39.75	35.33
P4P	47.70	50.71	27.38	42.48	42.88	40.98	34.11

P4P achieves higher TReB and TableEval accuracy than both alternatives. In Qwen, the post-window-attention feature at an anchor position has access to the entire window, so selecting an anchor near a boundary does not restrict its representation to boundary content. The boundary-only comparison supports the benefit of this contextualized information on table tasks, although P4P does not outperform the boundary-only variant on every ChartQAPro category.

G.2 Efficiency–accuracy trade-off

Table 4 reports throughput and peak memory for P4P and overlapping windows. P4P is slower and uses more memory in all three measured benchmark settings. Its advantage in this comparison is higher accuracy, particularly on TReB and TableEval, rather than lower raw computational cost. The measurements should therefore be interpreted as an efficiency–accuracy trade-off at the evaluated configurations, not as an accuracy-matched efficiency comparison.

Table 4: Inference measurements for Qwen2.5-VL-7B. Throughput is in samples/s and peak memory is in GiB.

Benchmark	Method	Throughput	Peak memory
TReB	Overlapping windows	1.037	16.59
	P4P	0.945	16.99
TableEval	Overlapping windows	0.746	17.02
	P4P	0.698	17.63
ChartQAPro	Overlapping windows	1.552	16.91
	P4P	1.386	17.44

H Additional anchor and layer ablations

H.1 Top-K ratio

The main experiments use a Top-K ratio of 15%. Table 5 compares ratios of 10%, 15%, and 20% on Qwen2.5-VL-7B. For Qwen, this ratio is applied to the boundary candidate set, as specified in Algorithm 1. The default 15% achieves the highest TReB and TableEval accuracy. ChartQAPro categories prefer different ratios, so the results do not suggest a universally optimal anchor budget or a monotonic accuracy improvement with more anchors.

Table 5: Top-K ratio ablation on Qwen2.5-VL-7B. All values are accuracy (%); the last five columns are ChartQAPro categories.

Ratio	TReB	TableEval	Factoid	Conv.	Hypoth.	FC	MCQ
10%	46.13	45.94	27.19	41.19	41.63	43.24	36.26
15%	47.70	50.71	27.38	42.48	42.88	40.98	34.11
20%	47.13	47.56	25.89	40.87	43.69	43.90	36.82

H.2 Insertion depth

Table 6 compares all-layer insertion with early-only, middle-only, and late-only insertion on Qwen2.5-VL-7B. Later insertion generally yields stronger table performance: the late-only variant reaches 46.70 on TReB and 46.57 on TableEval, compared with 46.20/44.57 for early-only and 46.20/45.49 for middle-only. All-layer insertion achieves the highest scores on these two benchmarks. Results across ChartQAPro categories are mixed, indicating that the preferred insertion position also depends on the task.

Table 6: Insertion-depth ablation on Qwen2.5-VL-7B. All values are accuracy (%); the last five columns are ChartQAPro categories. The all-layer row is the main P4P configuration.

Position	TReB	TableEval	Factoid	Conv.	Hypoth.	FC	MCQ
All-layer	47.70	50.71	27.38	42.48	42.88	40.98	34.11
Early-only	46.20	44.57	25.96	41.51	35.61	41.70	33.61
Middle-only	46.20	45.49	26.10	42.08	38.18	39.44	33.18
Late-only	46.70	46.57	26.96	41.60	40.22	40.42	34.96

I General multimodal evaluation and inference measurements

We evaluate on MMBench v1.1, MMStar, and MME to examine whether table-focused fine-tuning with P4P affects broader multimodal capabilities. Full MMMU and MMMU-Pro are excluded because their table-oriented samples were used to construct the training data. Table 7 compares the original Base model, Direct SFT, and P4P for all three backbones.

Table 7: General multimodal evaluation. MMBench v1.1, MMStar, and MME Acc. are percentages; MME Score is reported in benchmark score units.

Method	MMBench v1.1	MMStar	MME Score	MME Acc.
Qwen2.5-VL-7B				
Base	82.10	62.67	2296	86.98
Direct SFT	80.00	61.08	2274	84.96
P4P	81.10	61.72	2275	85.00
MiniCPM-V-4.5				
Base	84.60	71.10	2384	88.88
Direct SFT	85.20	68.82	2348	87.95
P4P	85.20	69.53	2377	88.84
InternVL3.5-8B				
Base	82.50	68.69	2365	87.91
Direct SFT	80.90	65.83	2338	86.90
P4P	82.10	66.11	2340	87.28

Across these reported metrics, P4P matches or improves upon Direct SFT. However, several scores remain below those of the original Base models. These results support preservation relative to the same table-focused fine-tuning baseline, rather than a claim of universally unchanged performance relative to the pretrained model. They also do not establish effectiveness on vision-only backbones.

Table 8: Reported inference throughput (samples/s) and peak memory (GiB). These are separate measurements from the benchmark-specific overlapping-window comparison in Table 4.

Model	Method	Throughput	Peak memory
Qwen2.5-VL-7B	Base	5.45	16.28
	Direct SFT	5.45	16.28
	P4P	4.93	16.93
MiniCPM-V-4.5	Base	8.48	17.71
	Direct SFT	8.48	17.71
	P4P	7.83	17.89
InternVL3.5-8B	Base	4.77	16.82
	Direct SFT	4.79	16.82
	P4P	4.33	17.02

Relative to Direct SFT, throughput decreases by approximately 9.5%, 7.7%, and 9.6% for Qwen, MiniCPM, and InternVL, respectively. Peak memory increases by 0.65, 0.18, and 0.20 GiB. Throughput measures processing rate and is not a direct measurement of per-request latency.

J Evaluation on Qwen2.5-VL-32B

To complement the main 7–8B experiments, we fine-tune Qwen2.5-VL-32B with P4P and compare it with the original Base model and Direct SFT. Table 9 reports the complete results on the same structured-content benchmarks.

Table 9: Results on Qwen2.5-VL-32B. All values are accuracy (%); the last five columns are ChartQAPro categories.

Method	TReB	TableEval	Factoid	Conv.	Hypoth.	FC	MCQ
Base	59.54	48.32	39.17	42.82	43.81	54.10	55.79
Direct SFT	61.91	49.02	39.14	43.29	50.03	60.25	53.04
P4P	63.43	52.40	40.80	45.74	54.06	59.02	57.94

P4P improves on Direct SFT on TReB, TableEval, and four of the five ChartQAPro categories, with a lower score on Fact Checking. The improvements on TReB and TableEval are 1.52 and 3.38 percentage points, respectively. This provides evidence beyond the 7–8B setting, while leaving vision-encoder pre-training and systematic scaling of the visual backbone for future work.

NeurIPS Paper Checklist

1. **Claims**

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

   Answer: [Yes]

   Justification: The abstract and introduction (Section 1) clearly state three contributions: (1) identifying structural fragmentation in patch-based MLLMs, (2) proposing the P4P mechanism with bridging anchors and gated fusion, and (3) cross-architecture validation on Qwen, MiniCPM, and InternVL. These claims are directly supported by the experimental results in Section 4.

   Guidelines:

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

2. **Limitations**

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

   Answer: [Yes]

   Justification: Limitations are discussed in Appendix A.

   Guidelines:

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

3. **Theory assumptions and proofs**

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

   Answer: [N/A]

   Justification: The paper does not include formal theoretical results or proofs. The method is primarily empirical, with equations describing the architecture design (Section 3) rather than theoretical claims.

   Guidelines:

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

4. **Experimental result reproducibility**

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

   Answer: [Yes]

   Justification: The paper provides the full algorithm (Algorithm 1), detailed architecture descriptions (Section 3), training hyperparameters (Appendix D), data composition (Appendix C), benchmark details (Appendix E), and relevant implementation code in the supplementary material.

   Guidelines:

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

(c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).

(d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

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

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

   Answer: [Yes]

   Justification: Relevant implementation code and example training and deployment instructions are provided in the supplementary material. The accompanying README specifies the scope of this release. Training data consists of publicly available benchmarks (Visual-TableQA, Table-VQA, MMMU, MMMU-Pro, WikiTableQuestions) as detailed in Appendix C.

   Guidelines:

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

6. **Experimental setting/details**

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

   Answer: [Yes]

   Justification: Section 4.1 describes the experimental setup. Appendix D provides hardware details (8×RTX 4090), framework (ms-swift), hyperparameters (lr=1×10^{-5}, 4 epochs, batch size 32, max seq length 4096, bfloat16, Flash-Attention 2), and resolution strategy (max 1,003,520 pixels).

   Guidelines:

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

7. **Experiment statistical significance**

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

   Answer: [Yes]

   Justification: All reported results are averaged over three independent runs.

   Guidelines:

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

8. **Experiments compute resources**

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

   Answer: [Yes]

   Justification: Appendix D.1 specifies the hardware (8×NVIDIA RTX 4090, 24GB each). Table 2 reports training speed (iterations/second) for different configurations. The training framework and optimization strategy are fully described.

   Guidelines:

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

9. **Code of ethics**

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

   Answer: [Yes]

   Justification: The research conforms with the NeurIPS Code of Ethics. The work focuses on improving visual understanding of structured documents and does not involve human subjects, private data, or potentially harmful applications.

   Guidelines:

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

10. **Broader impacts**

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

    Answer: [N/A]

    Justification: This work is foundational research on improving visual encoder architectures. It does not introduce new capabilities for generating content, surveillance, or other applications with direct negative societal impact.

    Guidelines:

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

11. **Safeguards**

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

    Answer: [N/A]

    Justification: The paper proposes a lightweight attention module for existing vision encoders. The proposed module does not introduce a new high-risk data or model release.

    Guidelines:

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

12. **Licenses for existing assets**

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

Answer: [Yes]

Justification: All models (Qwen2.5-VL under Apache 2.0, InternVL3.5 under MIT, MiniCPM-V under Apache 2.0) and the training framework (ms-swift under Apache 2.0) permit research use. Evaluation benchmarks MMMU/MMMU-Pro (Apache 2.0), ChartQAPro (MIT), and WikiTableQuestions (CC-BY 4.0) explicitly allow academic use. TReB, TableEval, and Table-VQA-Bench are distributed via gated access on HuggingFace with terms permitting research use; we comply with their access agreements.

Guidelines:

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

13. **New assets**

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

Answer: [Yes]

Justification: Relevant implementation code is provided in the supplementary material, together with a README describing the included components, installation, training, and deployment.

Guidelines:

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

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

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

Answer: [N/A]

Justification: The paper does not involve crowdsourcing or research with human subjects. All evaluations use automated benchmarks with programmatic scoring.

Guidelines:

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

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

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

    Answer: [N/A]

    Justification: The paper does not involve human subjects research. No IRB approval was required.

    Guidelines:

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

16. **Declaration of LLM usage**

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

    Answer: [No]

    Justification: LLMs are not used as a core component of the proposed method. The P4P mechanism is a deterministic attention module.

    Guidelines:

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