THE HIDDEN EVOLUTION OF DISGUISED VISUAL CONTEXT INSIDE VLMS

Anonymous authors
Paper under double-blind review

ABSTRACT

Visual tokens enter Large Language Models (LLMs) as raw, foreign signals. How they are transformed into consistent representations and interact with the text token space depends entirely on the integration architecture. Existing architectures typically either treat visual tokens as in-context prompts within the input sequence or inject them directly into the LLM's intermediate layers. A controlled comparison and understanding of how these architectural choices affect visual information and its internal transformation to integrate with the LLM remains underexplored. In this study, we provide a fair comparison by evaluating in-context and layer-wise injection Vision-Language Model (VLM) integration paradigms under identical training conditions on benchmarks spanning across single-image, multi-image, and video understanding. Our results show that in-context injection consistently outperforms layer-wise injection, with the largest gaps on OCR and video tasks, even though both allocate attention to visual tokens in similar patterns. This raises the question of what happens to visual information inside the LLM and how the LLM utilizes it. Investigating this, we uncover a hidden evolution of visual tokens, which enter the LLM as disguised visual context from a different representation space, lacking the structure of LLM representations, before being reshaped differently according to the integration paradigm. Our analysis reveals that this reshaping leads the visual representations in the LLM to capture fundamentally different frequency characteristics, which reflect whether they encode fine-grained local detail or coarse global structure. In-context injection progressively builds high-frequency, fine-grained detail, whereas layer-wise injection remains biased toward low-frequency features. We show that this evolution inside the LLM determines what visual features the VLM can capture and utilize effectively, such as the fine-grained detail required for text-centric tasks. It also determines whether visual representations converge toward the language space. Specifically, in-context visual tokens remain separate from text tokens through the middle layers and gradually merge with them by the final layers, whereas layer-wise visual tokens remain separated across layers. Since attention patterns are similar, the performance gap must instead come from the quality of the visual representations that each paradigm shapes at every layer. We confirm this through a causal intervention by suppressing the high-frequency content in the visual tokens inside the LLM, which degrades in-context injection across tasks, especially in text-centric ones, while layer-wise injection is far less affected. This shows that high-frequency content drives the advantage of in-context injection over layer-wise injection.

1 INTRODUCTION

Vision-Language Models (VLMs) (Bai et al., 2025; Deitke et al., 2025) succeed only when an LLM can reliably use visual information during generation; critically, integration design determines what "use" even means inside the model. The common approach concatenates visual tokens with text at the input so that visual tokens participate in standard self-attention throughout the network (Li et al., 2023; Merullo et al., 2023; Eichenberg et al., 2022; Tsimpoukelli et al., 2021; Mañas et al., 2023); we call this **in-context injection**. A competing family introduces visual information at intermediate layers through dedicated mechanisms (Alayrac et al., 2022; Li et al., 2025a); we call this **layer-wise injection**. Building on top of these approaches, modern VLM frameworks further strengthen their

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

multimodal capabilities by unfreezing the LLM backbone (Liu et al., 2023; 2024a) and the pretrained vision encoder (Tong et al., 2024), driving rapid progress across image understanding (Wang et al., 2024; Lin et al., 2023), video understanding (Lin et al., 2024), and unified frameworks (Li et al., 2025b; Zhang et al., 2025a). As VLMs progress toward longer-context and text-centric domains such as video and OCR, integration becomes a key factor in controlling both computational cost and what visual information can be composed and retained.

Despite the crucial role of integration design, we still lack controlled evidence about how different integration paradigms mechanistically change visual tokens inside the LLM. Most existing VLM comparisons (Deitke et al., 2025; Tong et al., 2024; An et al., 2025) are affected by differences in training data mixtures, token budgets, model scale, and optimization. In addition, most prior analysis studies (Shukor & Cord, 2024; Chen et al., 2024; Basu et al., 2024; Neo et al., 2025) focus on a single dominant paradigm, leaving it unclear whether their conclusions about representation evolution, modality alignment, or information usage generalize to other integration paradigms. As a result, it is difficult to attribute performance differences to the integration paradigm itself, and integration choices are often made based on convenience or efficiency without a clear understanding of the tradeoffs among them.

This paper isolates integration as the key variable by evaluating representative in-context and layer-wise injection architectures under identical training conditions across single-image, multi-image, and video benchmarks. Beyond reporting benchmark outcomes, we provide a mechanistic characterization of how visual information is processed inside the LLM through five analyses:

- **Representation evolution:** we analyze how each integration paradigm transforms and reshapes visual representations to interact inside the LLM differently across layers.
- **Feature content across depth:** we examine what types of visual characteristics each approach captures across the depth of the LLM via frequency-based analysis.
- **Convergence to language space:** we study whether visual tokens in each paradigm converge toward the language space as they pass through deeper layers of the LLM.
- **Visual information usage during generation:** we analyze when the model utilizes visual information during answer generation and show that attention allocation alone is insufficient to explain the performance gaps between integration paradigms.
- **High-frequency suppression:** we apply causal intervention that suppresses the high-frequency content of visual representations, confirming that these features drive the advantage of in-context injection over layer-wise injection on text-centric tasks.

Our study reveals that the choice of integration paradigm shapes the quality of visual representations inside the LLM, which determines how effectively text tokens can utilize them during answer generation. Our analysis offers a mechanistic explanation of why different integration strategies exhibit distinct behaviors and strengths across tasks, especially in settings that require composing fine-grained or temporal evidence across many visual tokens, such as OCR and video. Together, these findings provide a principled basis for selecting integration paradigms and designing future VLM architectures beyond benchmark-driven trial and error.

2 Model Architectures and Training Data

2.1 Integration Paradigms and Architectures

To conduct a fair systematic evaluation, we standardize the training data and optimization recipe across all models, so that observed differences can be attributed primarily to the integration design rather than confounding factors. We focus on integration paradigms that are both widely adopted and representative of distinct design philosophies for delivering visual context into an LLM. Concretely, we employ three distinct baselines representing the primary vision-language integration paradigms as illustrated in Figure 1. First, in-context injection, which treats visual tokens as part of the LLM input sequence. We represent this category with **IN-CT** (Liu et al., 2024a), which concatenates projected visual tokens with the text tokens. Second, gated cross-attention layer-wise injection, which introduces visual information through dedicated cross-attention blocks inserted between the LLM

layers. We represent this category with **LW-GC** (Alayrac et al., 2022). Third, attention-only layer-wise injection, which incorporates visual information directly into the LLM's existing self-attention mechanism without introducing separate cross-attention blocks. We represent this category with **LW-AT** (Alex et al., 2025). Together, these three architectures capture the primary integration choices found in the vision-language literature.

Given an input image V, a vision encoder Φ extracts a sequence of N_v visual features, $Z_v = \Phi(V) \in \mathbb{R}^{N_v \times d_v}$, where d_v is the output dimension of the vision encoder. All three paradigms share the same connector architecture, a simple two-layer MLP with GELU activation: $\tilde{Z}_v = \Psi(Z_v) = W_{\text{out}}\,\text{GELU}(W_{\text{in}} \cdot Z_v)$, where $\tilde{Z}_v \in \mathbb{R}^{N_v \times d_t}$ denotes the projected visual tokens, d_t is the hidden dimension of the LLM, and W_{in} and W_{out} are learnable projection matrices. Bias terms are omitted for clarity. For LW-GC, we do not utilize a Perceiver (Jaegle et al., 2021) nor compress the number of visual tokens, maintaining an identical number of visual tokens across all paradigms. The three integration mechanisms are detailed as follows:

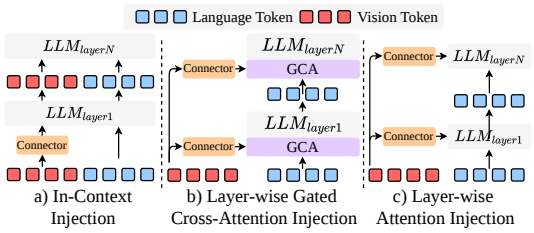


Figure 1: **Overview of the three VLM integration paradigms evaluated in this study**. (a) IN-CT concatenates visual tokens with text tokens at the input layer, (b) LW-GC introduces visual features through gated cross-attention (GCA) blocks, and (c) LW-AT injects visual features directly into the LLM's keys and values.

1. IN-CT (In-context injection): A single connector projects the visual features into the LLM embedding space, producing $\tilde{Z}_v$. Let $H_t \in \mathbb{R}^{N_t \times d_t}$ denote the embeddings of the N_t text tokens. The initial hidden state is formed by concatenating the projected visual and text tokens, $h_0 = [\tilde{Z}_v; H_t]$, after which the combined sequence is processed jointly through the LLM's standard transformer blocks, as shown in Figure 1(a).

2. LW-GC (Layer-wise with gated cross-attention): The initial hidden-state sequence contains only the text token embeddings ($h_0 = H_t$). At each layer l, a layer-specific connector Ψ_l maps visual features, which are injected via a gated cross-attention block (CA) and a gated feed-forward network (FFN), both modulated by learnable scalars α_l and β_l initialized to zero:

$$h_l^{\text{cross}} = h_l + \tanh(\alpha_l) \cdot \text{CA}(h_l,\, \tilde{Z}_v^l,\, \tilde{Z}_v^l), \tag{1}$$

$$h_l^{\text{gated}} = h_l^{\text{cross}} + \tanh(\beta_l) \cdot \text{FFN}(h_l^{\text{cross}}). \tag{2}$$

We evaluate two injection schedules: gated cross-attention blocks inserted at every LLM layer and blocks inserted once every four layers, as shown in Figure 1(b).

3. LW-AT (Layer-wise with attention-only injection): Starting with $h_0 = H_t$, layer-specific connectors map visual features that are injected solely into the LLM's existing attention, explicitly bypassing the FFNs:

$$h'_l = \text{Attn}\big(h_l,\, [\tilde{Z}_v^l; h_l],\, [\tilde{Z}_v^l; h_l]\big). \tag{3}$$

LW-AT bridges the two prior paradigms: like LW-GC, visual features are injected layer-by-layer, but like IN-CT, visual and text tokens interact simultaneously within standard attention, as shown in Figure 1(c). Further implementation details and training settings for all three paradigms are provided in Appendix B and C.

2.2 Model and Dataset

We use Qwen2.5-Instruct-0.5B, 3B, and 7B (Yang et al., 2025) and LLaMA-3.2-Instruct-1B and 3B (Grattafiori et al., 2024) as LLM backbones and SigLIP2-So400m (Tschannen et al., 2025) as the vision encoder. Training proceeds in two stages: (i) connector pretraining and (ii) supervised fine-tuning (instruction tuning). For connector pretraining, we use the BLIP-558K dataset (Liu et al., 2024a). For instruction tuning, we employ two data mixtures: (1) the **LLaVA-NeXT** instruction tuning data (Liu et al., 2024b) comprising ∼700K samples, and (2) the **LLaVA-OneVision (OV)** instruction tuning data (Li et al., 2025b), a comprehensive mixture of single images, multi-image, and

videos comprising ∼1.6M samples. Throughout the paper, we refer to these two instruction tuning mixtures as **LLaVA-NeXT** and **LLaVA-OV**, respectively.

3 Benchmark Results

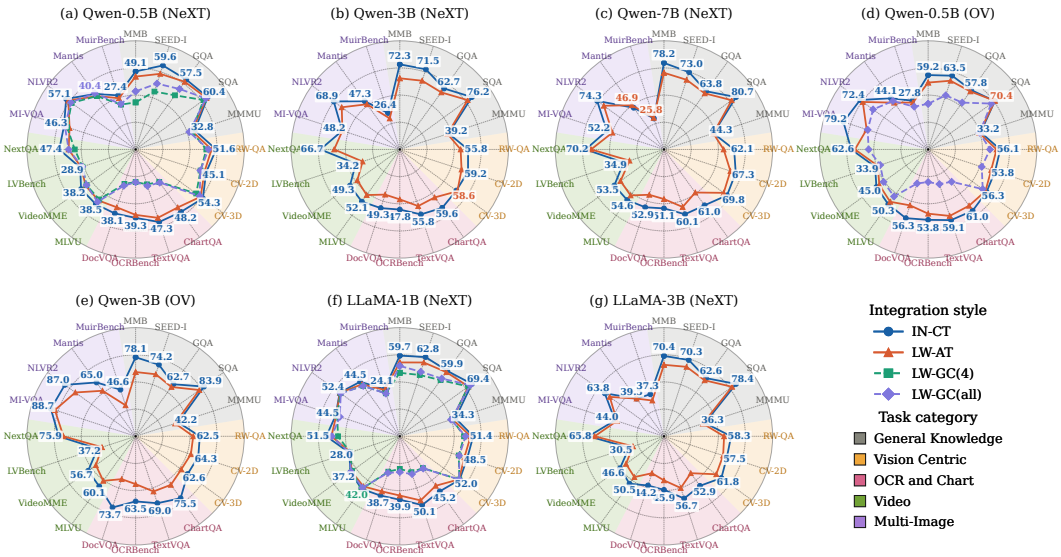


Figure 2: **Benchmark comparison of the three integration paradigms across single-image, multi-image, and video understanding**. Panels correspond to backbone and training setup: Qwen2.5-Instruct (0.5B, 3B, 7B) and LLaMA-3.2-Instruct (1B, 3B) with LLaVA-NeXT, and Qwen2.5-Instruct with LLaVA-OV on 0.5B and 3B. Axes are grouped by task type, and the annotated value on each axis is the best score on that benchmark within the panel.

We evaluate the three integration paradigms on benchmarks spanning single-image understanding (general knowledge, vision-centric, and OCR), multi-image understanding, and video understanding. As shown in Figure 2 (complete numerical results are reported in Tables 2 and 3 of Appendix D.1), IN-CT achieves the strongest overall performance, with LW-AT following closely and occasionally surpassing it on individual tasks. LW-GC, in contrast, lags behind both by a wide margin. This trend holds across both LLM backbones, all model scales, and both instruction tuning mixtures (LLaVA-NeXT and LLaVA-OV), indicating that the performance differences stem from the integration paradigm rather than from any specific LLM backbone, size, or training data. The gap between IN-CT and the layer-wise variants is largest specifically on the OCR and video tasks, while performance remains close elsewhere. Since LW-AT shares the same token interaction mechanism as IN-CT, differing only in how visual tokens are injected and passed to the LLM, this gap points directly at the injection choice. The gap in video tasks can be explained by the need to integrate information across frames. Inside the LLM, attention among visual tokens across frames is causally necessary for capturing temporal information (Kim et al., 2026), a capability that in-context injection supports through joint self-attention but that layer-wise injection cannot provide, since its visual tokens remain static and do not attend to one another. On OCR, this gap is even more striking. For OCR, both layer-wise variants consistently fall behind IN-CT by a large margin, especially LW-GC.

3.1 Effect of Text-Centric Data

Based on these observations, we hypothesize that text-centric visual data may provide limited benefit to the optimization of layer-wise injection models. To test this, we remove text-centric training data from the LLaVA-NeXT mixture (the filtered datasets are listed in Appendix C.3) and compare the resulting performance change. Figure 3 reveals a clear difference in how each architecture responds (per-benchmark results for all backbones are reported in Table 4 of Appendix D.1). For IN-CT, removing these samples leads to consistent performance decreases across nearly every benchmark, whereas for layer-wise variants the impact is more selective, with both variants even improving on

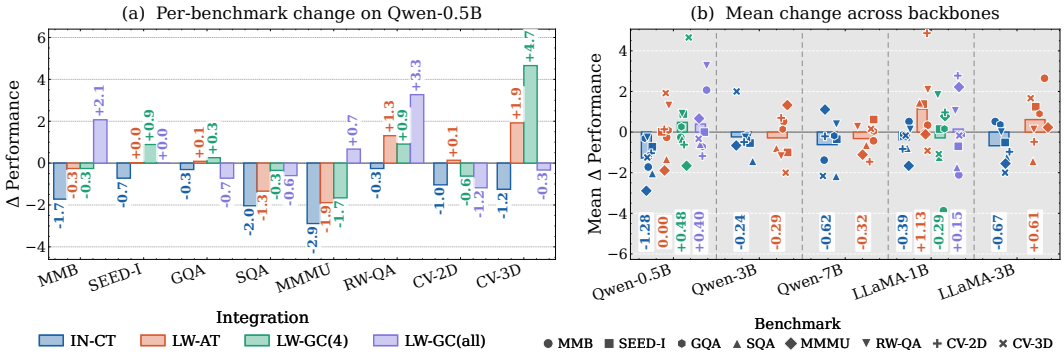


Figure 3: **Effect of removing text-centric data**. (a) Per-benchmark change (Δ score) of the Qwen 0.5B model trained on the text-centric-filtered mixture relative to the full LLaVA-NeXT mixture. (b) Mean of these Δ scores per backbone, with bars showing the mean per integration method and dots the individual benchmarks.

several benchmarks. This suggests that in-context injection can broadly leverage text-centric data to benefit general visual understanding, whereas layer-wise injection models do not utilize such data effectively, and on some tasks the data may even interfere with their optimization. We attribute this advantage to in-context visual tokens participating in the LLM's attention layers, allowing tokens from different image patches to attend to one another and progressively integrate spatially distributed information, such as characters spanning multiple patches, into coherent representations. Layer-wise injection cannot replicate this behavior, as its visual tokens never attend to one another.

4 The Hidden Evolution of Visual Context

The experimental results reveal a consistent performance gap between in-context injection and layer-wise injection, particularly in OCR and video tasks. What causes this gap? To understand this, we conduct five analyses examining (1) whether visual tokens evolve inside the LLM, (2) what features they capture, (3) whether they align with the language space, (4) when the model utilizes visual information, and (5) whether higher-frequency content causally drives performance. Unless stated otherwise, all visualizations in this section use the Qwen2.5-Instruct-0.5B backbone trained on the LLaVA-NeXT mixture, with LW-GC(all) as the gated cross-attention variant, since every-layer insertion provides multimodal interaction at every LLM layer, matching IN-CT and LW-AT. Visualizations for larger backbones, which follow the same trends, are provided in Appendix E.

4.1 Do Visual Tokens Evolve Inside the LLM?

Given that IN-CT consistently outperforms layer-wise injection, we hypothesize that visual tokens in IN-CT undergo a progressive transformation through the LLM layers, similar to how text tokens are refined. To test this, we utilize Centered Kernel Alignment (CKA) (Kornblith et al., 2019), which, unlike cosine similarity used in prior work (Shukor & Cord, 2024; Zhang et al., 2025d), is invariant to scaling and rotation, capturing structural shifts across entire token sets rather than individual vectors. Specifically, we compute CKA between the token representations of each sample at layers ℓ and ℓ', averaged over all samples (formal definitions and visualization of larger backbones are provided in Appendix E.1). As shown in Figure 4(a-b), visual tokens in IN-CT exhibit a smooth, uniform transformation across layers. Text tokens are equally smooth but evolve at different depths, shifting most in the early layers while visual representations change most in the later layers. In contrast, the visual tokens of LW-AT and LW-GC exhibit severe discontinuities across layers, even though their text tokens remain smooth. The difference lies in how visual information is delivered. In IN-CT, visual tokens reside in the residual stream, so each layer introduces only incremental changes. In layer-wise injection, visual tokens bypass the residual stream and are supplied through independent per-layer projections of the encoder features, without any constraint linking consecutive layers. While different LLM layers may require different interpretations of visual features, these results show that the two integration families arrive at fundamentally different solutions to this problem.

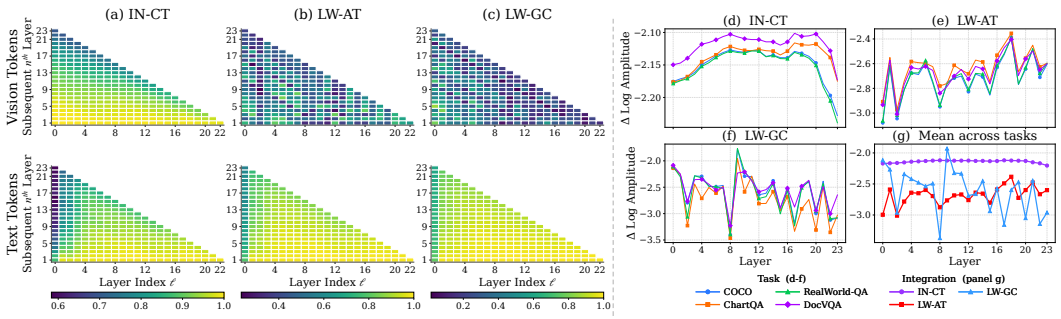


Figure 4: (Left) CKA heatmaps of visual (top row) and text (bottom row) token representations across layers on ChartQA for (a) IN-CT, (b) LW-AT, and (c) LW-GC. IN-CT evolves smoothly across layers, while LW-AT and LW-GC show severe discontinuities in their visual tokens despite smooth text tokens. (Right) Relative log amplitude of Fourier-transformed visual token representations across layers on four datasets, shown per paradigm on its own scale for (d) IN-CT, (e) LW-AT, and (f) LW-GC, and (g) averaged across tasks for all paradigms. IN-CT holds the highest amplitude, further increased for text-centric tasks, while LW-AT and LW-GC remain lower with no such task ordering.

Finding 1: Visual tokens in IN-CT semantically evolve across LLM layers, mirroring the smooth refinement of text tokens. In contrast, layer-wise injection produces discontinuous representations across layers.

4.2 What Do Visual Representations Capture?

Building on Finding 1, we next examine what visual features each paradigm captures across layers. Following the Fourier analysis framework (Park et al., 2023), we compute the relative log amplitude of Fourier-transformed hidden states at each layer, measuring the amplitude difference between the highest and lowest frequencies. Higher values indicate greater capture of higher-frequency information, while lower values indicate a bias toward lower-frequency information. As shown in Figure 4(g) (results for larger backbones are provided in Appendix E.2) , IN-CT captures and maintains higher-frequency features, holding the highest relative log amplitude across layers, whereas both layer-wise variants remain biased toward low-frequency features, with LW-AT fluctuating but relatively stable and LW-GC oscillating erratically with no coherent trend. These oscillations are consistent with the non-uniform representation shifts in Section 4.1.

We next examine the per-task frequency behavior of each paradigm in Figure 4(d-f). The IN-CT amplitude increases gradually through the early and middle layers, indicating a progressive shift toward fine-grained local details such as texture and edges. Moreover, its curves are consistently ordered by task type, with text-centric tasks such as DocVQA and ChartQA maintaining higher amplitudes than general vision tasks such as COCO and RealWorld-QA at nearly every layer, showing that IN-CT builds higher-frequency content for the inputs that demand fine-grained detail. The layer-wise variants lack this property, as their text-centric curves remain at similar amplitudes to the general vision tasks. Combined with their overall lower amplitudes observed above, this may explain why IN-CT's advantage is largest on OCR, where fine-grained detail matters most. Notably, the IN-CT amplitude drops sharply in the final layers, shifting back toward lower-frequency features. This aligns with Section 4.1, where visual representations change most in the later layers, and suggests that the LLM consolidates fine-grained details into more abstract, global representations, potentially converging toward the language space, which we investigate in Section 4.3.

Finding 2: IN-CT captures higher-frequency features, especially for tasks requiring fine-grained detail, and consolidates them into lower-frequency form in the final layers. LW-AT and LW-GC remain lower-frequency biased and show no such task differentiation.

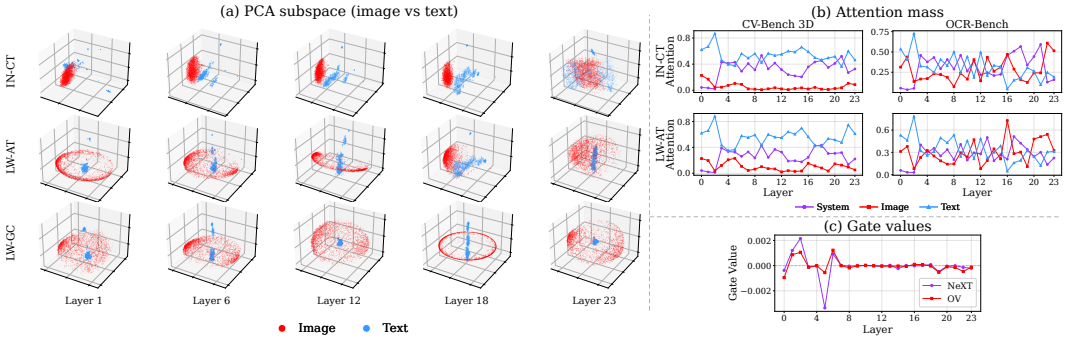


Figure 5: (Left) 3D PCA projections of image (red) and text (blue) token representations across layers for the three integration paradigms (rows: IN-CT, LW-AT, LW-GC), shown in (a). In IN-CT, the two modalities progressively merge into a shared space, whereas in LW-AT and LW-GC the image tokens remain separated from the text tokens throughout all layers. (Right) Visual information utilization. (b) Attention mass of generated tokens toward system, image, and text tokens across layers for IN-CT and LW-AT on a general and a text-centric task. (c) Learned gate values of LW-GC across layers under both instruction tuning mixtures.

4.3 Do Visual Tokens Align with the Language Representation Space?

Following Section 4.2, we investigate whether the final-layer consolidation reflects convergence toward the language representation space. To test this, we project image and text hidden states at each layer into a shared 3D PCA space. As shown in Figure 5(a), the paradigms exhibit fundamentally different behaviors (visualizations of every layer and larger backbones are provided in Appendix E.4). In IN-CT, image tokens already carry rich semantic structure from the vision encoder in the early layers, while text tokens have yet to form meaningful representations. As text tokens acquire semantics in the middle layers, both modalities occupy narrow modality-specific cones, consistent with the modality gap (Liang et al., 2022) observed in MLLMs (Shukor & Cord, 2024). However, as tokens propagate deeper, these cones progressively merge, converging into a shared representational space by the final layers. In contrast, neither layer-wise paradigm exhibits this convergence. In both LW-AT and LW-GC, image representations remain separated from the text cluster at every depth. This is expected, as layer-wise injection is not designed to progressively transform visual tokens through the LLM, and the representations are therefore never reshaped toward the language space, consistent with the discontinuous representations observed in Section 4.1.

Finding 3: Visual tokens in IN-CT progressively converge toward the language space across LLM layers, while layer-wise tokens remain separated from it.

4.4 When Is Visual Information Used?

Sections 4.1–4.3 established that IN-CT, LW-AT, and LW-GC differ fundamentally in their representations. How does the LLM utilize these representations during generation? For IN-CT and LW-AT, we quantify visual information utilization by generated tokens at each layer through Attention Mass, the total attention weight directed toward visual tokens. For a generated token at position t, the attention mass for a segment S at layer ℓ is

$$\text{Mass}_S^{(\ell)} = \frac{1}{H} \sum_{h=1}^{H} \sum_{j \in S} \alpha_{t,j}^{(\ell,h)} \tag{4}$$

where H is the number of attention heads and $\alpha_{t,j}^{(\ell,h)}$ is the attention weight from position t to position j in head h of layer ℓ.

We confirm prior findings (Chen et al., 2024; Zhang et al., 2025c) that IN-CT attends to visual tokens mainly in shallow layers, diminishing with depth, on general tasks such as CV-Bench 3D as shown

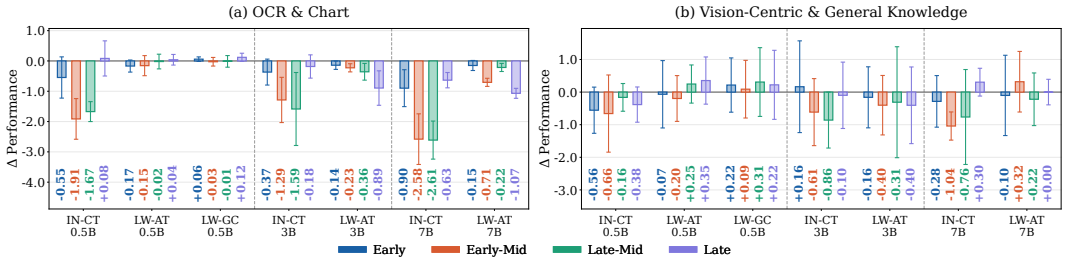


Figure 6: **Performance change from low-pass filtering the visual token hidden states at four decoder depth groups, shown for each model and scale on (a) OCR & Chart and (b) Vision-Centric and General Knowledge tasks.** Bars report the mean change relative to the unfiltered model and error bars denote variation across benchmarks within each group. IN-CT degrades most, with damage concentrated in the Early-Mid and Late-Mid groups on OCR & Chart, while LW-AT is far less affected and General Knowledge remains largely unchanged.

in Figure 5(b), but find this pattern task-dependent (visualizations on additional tasks and for larger backbones are provided in Appendix E.3). In text-centric tasks such as OCR-Bench, attention to image tokens remains high through middle and deep layers. For LW-AT, the attention distribution is surprisingly similar to IN-CT. Yet IN-CT consistently outperforms LW-AT. We attribute this gap to the quality of information at each layer rather than attention allocation. Specifically, IN-CT captures higher-frequency features (Section 4.2), providing fine-grained details critical for OCR, while LW-AT remains low-frequency biased. We further hypothesize that IN-CT benefits on OCR from language-space convergence (Section 4.3), as the LLM reads text-like semantics rather than raw visual features. For LW-GC, fixed learned gate values control the visual contribution regardless of input, as shown in Figure 5(c). The gates are active only in early layers and near zero in the deeper layers, for both NeXT and OV. This explains why LW-AT outperforms LW-GC despite both being low-frequency biased, as it can adapt its visual contribution through attention. We hypothesize that variants with dynamic gating (Ye et al., 2025) could resolve this, but would still face the same frequency limitations as LW-AT.

Finding 4: Visual information utilization by generated tokens varies across layers and is task-dependent. The performance gap cannot be explained by attention allocation alone but also reflects the quality of representations at each layer.

4.5 Do High-Frequency Features Provide the Benefit on Text-Centric Tasks?

Since IN-CT builds higher-frequency features (Section 4.2) and its advantage over the layer-wise variants arises from representation quality (Section 4.4), this content may itself underlie the text-centric advantage. To test this directly, we suppress the high-spatial-frequency content of visual representations inside the LLM while preserving their coarse structure and measure the downstream effect. Let $Z_\ell \in \mathbb{R}^{H\times W\times d}$ denote the visual-token hidden states at decoder layer ℓ on their spatial grid. We apply an ideal low-pass filter in the 2-D Fourier domain

$$\tilde{Z}_\ell = \mathcal{F}^{-1}\big(M_\rho \odot \mathcal{F}(Z_\ell)\big),$$
$$M_\rho(u,v) = \begin{cases} 1 & \text{if } \sqrt{u^2+v^2} \le \rho \cdot k_{\max} \\ 0 & \text{otherwise} \end{cases} \tag{5}$$

where $\mathcal{F}$ is the per-channel 2-D DFT, (u, v) are the spatial frequencies measured from the spectrum center, and $k_{\max}$ is the highest representable frequency. We set the cutoff $\rho = 0.9$, removing only the highest-frequency band. To localize where fine detail matters, we filter one of four decoder depth groups (Early, Early-Mid, Late-Mid, Late) at a time and report performance change per task category, as shown in Figure 6 (per-benchmark results in Appendix F).

The results reveal a consistent asymmetry between the integration strategies. At every scale, the largest performance drops belong to IN-CT, while LW-AT degrades far less at matched scales and

depths, and LW-GC is nearly unaffected. This asymmetry follows directly from Section 4.2, as the filter suppresses precisely the content only IN-CT constructs. The degradation is most severe on OCR & Chart and lesser on Vision-Centric and General Knowledge. Consistent with Section 4.4, although IN-CT and LW-AT exhibit similar attention patterns, the intervention harms IN-CT far more, as the two differ in the quality of their visual representations. The tasks least reliant on fine detail show only minor degradation as shown in Figure 6, confirming the intervention removes fine-grained information rather than broadly disrupting computation. These results thus causally confirm that the representation quality separating IN-CT from layer-wise injection is its higher-frequency content, a key source of its text-centric advantage.

Finding 5: Suppressing high-frequency visual content selectively harms IN-CT on text-centric tasks, confirming this content drives its text-centric advantage.

5 Ablation Study

Our analysis has shown that IN-CT and LW-AT capture different frequency characteristics, with IN-CT capturing higher-frequency features while LW-AT maintains lower-frequency representations. To explore whether combining both paradigms can leverage these properties, we construct a hybrid model, IN-CT + LW-AT, applying both integration strategies simultaneously. Specifically, layer-wise attention injection is interleaved before the in-context tokens with a 1:1 token mapping (full details in Appendix B.4). As shown in Figure 7, the hybrid achieves the best performance across most benchmarks, outperforming either paradigm alone. However, it doubles the visual token count, making it computationally impractical. We present this as an exploratory finding, motivating future architectures that could combine both strategies efficiently.

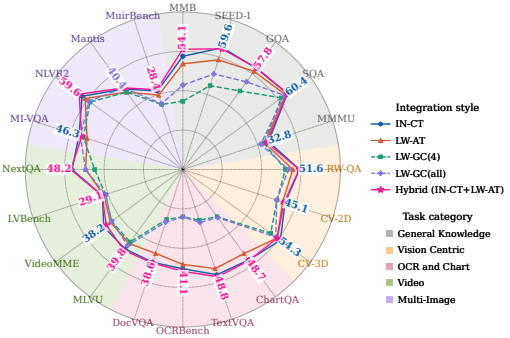


Figure 7: Benchmark performance of the hybrid IN-CT + LW-AT model compared with the individual integration paradigms, where the annotated value on each axis is the best score on that benchmark (numerical results are reported in Table 5 of Appendix D.2).

6 Conclusion

We presented a systematic comparison of in-context and layer-wise integration paradigms in VLMs under identical training conditions. Through five analyses, we uncovered that in-context injection enables a hidden evolution of visual tokens, progressively shifting toward high-frequency features before converging with the language space, while layer-wise injection remains biased toward low-frequency features and separated from it. These representational differences, rather than attention allocation, drive the performance gap between the paradigms, as causally confirmed by our frequency intervention. Our hybrid experiment suggests a promising direction for future architectures that combine representational depth with computational efficiency. Though our study focuses on vision-language, these findings are relevant to the broader MLLM community. Different modalities may benefit from different frequency characteristics. In particular, each modality could be routed through either the efficient layer-wise pathway or the more expensive in-context pathway according to its representation's detail requirements, allowing future architectures to scale across modalities without the prohibitive bottleneck of universal in-context processing. We hope this work provides actionable insights for next-generation VLMs and MLLMs.

Limitations: All experiments use a constant-epoch budget, which does not equalize compute, as IN-CT runs visual tokens through quadratic attention and FFNs, consuming substantially more per epoch than the layer-wise variants. Under equal compute, the layer-wise methods could train longer, potentially narrowing the benchmark gaps. The representational differences we identify arise from the architecture itself and are not expected to disappear with additional training.

AI USE STATEMENT

In this work, we used generative AI tools only to aid and polish the writing of the paper, namely grammar, formatting, and clarity and flow. We have not used generative AI tools for any task with required disclosure. All intellectual and experimental contributions were made by the authors. We take responsibility for the final content of this work, including text produced with the aid of generative AI.

REPRODUCIBILITY STATEMENT

All LLM backbones, the vision encoder, training datasets, and benchmarks used in this work are publicly available. The three integration architectures are described in Section 2.1, with implementation details in Appendix B. Training details, including input image processing, hyperparameters, computing infrastructure, and the training data mixtures with the list of filtered datasets, are provided in Appendix C. The analysis methods are defined in Section 4 and Appendix E.1, and the evaluation setup is described in Appendix C.4. Code and pretrained models will be made available upon acceptance.

REFERENCES

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

Tony Alex, Wish Suharitdamrong, Sara Atito, Armin Mustafa, Philip JB Jackson, Imran Razzak, and Muhammad Awais. Pal: Probing audio encoders via llms-audio information transfer into llms. *arXiv preprint arXiv:2506.10423*, 2025.

Xiang An, Yin Xie, Kaicheng Yang, Wenkang Zhang, Xiuwei Zhao, Zheng Cheng, Yirui Wang, Songcen Xu, Changrui Chen, Didi Zhu, et al. Llava-onevision-1.5: Fully open framework for democratized multimodal training. *arXiv preprint arXiv:2509.23661*, 2025.

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

Samyadeep Basu, Martin Grayson, Cecily Morrison, Besmira Nushi, Soheil Feizi, and Daniela Massiceti. Understanding information storage and transfer in multi-modal large language models. *Advances in Neural Information Processing Systems*, 37:7400–7426, 2024.

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

Matt Deitke, Christopher Clark, Sangho Lee, Rohun Tripathi, Yue Yang, Jae Sung Park, Mohammadreza Salehi, Niklas Muennighoff, Kyle Lo, Luca Soldaini, et al. Molmo and pixmo: Open weights and open data for state-of-the-art vision-language models. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pp. 91–104, 2025.

Haiwen Diao, Xiaotong Li, Yufeng Cui, Yueze Wang, Haoge Deng, Ting Pan, Wenxuan Wang, Huchuan Lu, and Xinlong Wang. Evev2: Improved baselines for encoder-free vision-language models. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 21014–21025, 2025.

Constantin Eichenberg, Sidney Black, Samuel Weinbach, Letitia Parcalabescu, and Anette Frank. Magma–multimodal augmentation of generative models through adapter-based finetuning. In *Findings of the association for computational linguistics: EMNLP 2022*, pp. 2416–2428, 2022.

Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Alex Vaughan, et al. The llama 3 herd of models. *arXiv preprint arXiv:2407.21783*, 2024.

Wenyi Hong, Weihan Wang, Qingsong Lv, Jiazheng Xu, Wenmeng Yu, Junhui Ji, Yan Wang, Zihan Wang, Yuxiao Dong, Ming Ding, et al. Cogagent: A visual language model for gui agents. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 14281–14290, 2024.

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

Nick Jiang, Anish Kachinthaya, Suzanne Petryk, and Yossi Gandelsman. Interpreting and editing vision-language representations to mitigate hallucinations. In *International Conference on Learning Representations*, volume 2025, pp. 63582–63605, 2025a.

Zhangqi Jiang, Junkai Chen, Beier Zhu, Tingjin Luo, Yankun Shen, and Xu Yang. Devils in middle layers of large vision-language models: Interpreting, detecting and mitigating object hallucinations via attention lens. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 25004–25014, 2025b.

Omri Kaduri, Shai Bagon, and Tali Dekel. What's in the image? a deep-dive into the vision of vision language models. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pp. 14549–14558, 2025.

Kushal Kafle, Brian Price, Scott Cohen, and Christopher Kanan. Dvqa: Understanding data visualizations via question answering. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 5648–5656, 2018.

Seil Kang, Jinyeong Kim, Junhyeok Kim, and Seong Jae Hwang. See what you are told: Visual attention sink in large multimodal models. In *International Conference on Learning Representations*, volume 2025, pp. 87676–87703, 2025.

Aniruddha Kembhavi, Mike Salvato, Eric Kolve, Minjoon Seo, Hannaneh Hajishirzi, and Ali Farhadi. A diagram is worth a dozen images. In *European conference on computer vision*, pp. 235–251. Springer, 2016.

Geewook Kim, Teakgyu Hong, Moonbin Yim, JeongYeon Nam, Jinyoung Park, Jinyeong Yim, Wonseok Hwang, Sangdoo Yun, Dongyoon Han, and Seunghyun Park. Ocr-free document understanding transformer. In *European Conference on Computer Vision*, pp. 498–517. Springer, 2022.

Minji Kim, Taekyung Kim, and Bohyung Han. Map the flow: Revealing hidden pathways of information in videollms. In *International Conference on Learning Representations (ICLR)*, 2026.

Simon Kornblith, Mohammad Norouzi, Honglak Lee, and Geoffrey Hinton. Similarity of neural network representations revisited. In Kamalika Chaudhuri and Ruslan Salakhutdinov (eds.), *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pp. 3519–3529. PMLR, 09–15 Jun 2019. URL https://proceedings.mlr.press/v97/kornblith19a.html.

Hugo Laurençon, Lucile Saulnier, Léo Tronchon, Stas Bekman, Amanpreet Singh, Anton Lozhkov, Thomas Wang, Siddharth Karamcheti, Alexander Rush, Douwe Kiela, et al. Obelics: An open web-scale filtered dataset of interleaved image-text documents. *Advances in Neural Information Processing Systems*, 36:71683–71702, 2023.

Hugo Laurençon, Léo Tronchon, Matthieu Cord, and Victor Sanh. What matters when building vision-language models? *Advances in Neural Information Processing Systems*, 37:87874–87907, 2024.

Bo Li, Yuanhan Zhang, Liangyu Chen, Jinghao Wang, Fanyi Pu, Joshua Adrian Cahyono, Jingkang Yang, Chunyuan Li, and Ziwei Liu. Otter: A multi-modal model with in-context instruction tuning. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 47(9):7543–7557, 2025a.

Bo Li, Yuanhan Zhang, Dong Guo, Renrui Zhang, Feng Li, Hao Zhang, Kaichen Zhang, Peiyuan Zhang, Yanwei Li, Ziwei Liu, and Chunyuan Li. LLaVA-onevision: Easy visual task transfer. *Transactions on Machine Learning Research*, 2025b. ISSN 2835-8856. URL https://openreview.net/forum?id=zKv8qULV6n.

Junnan Li, Dongxu Li, Silvio Savarese, and Steven Hoi. Blip-2: Bootstrapping language-image pre-training with frozen image encoders and large language models. In *International conference on machine learning*, pp. 19730–19742. PMLR, 2023.

Victor Weixin Liang, Yuhui Zhang, Yongchan Kwon, Serena Yeung, and James Y Zou. Mind the gap: Understanding the modality gap in multi-modal contrastive representation learning. *Advances in Neural Information Processing Systems*, 35:17612–17625, 2022.

Bin Lin, Yang Ye, Bin Zhu, Jiaxi Cui, Munan Ning, Peng Jin, and Li Yuan. Video-llava: Learning united visual representation by alignment before projection. In *Proceedings of the 2024 conference on empirical methods in natural language processing*, pp. 5971–5984, 2024.

Junyan Lin, Haoran Chen, Yue Fan, Yingqi Fan, Xin Jin, Hui Su, Jinlan Fu, and Xiaoyu Shen. Multi-layer visual feature fusion in multimodal llms: Methods, analysis, and best practices. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pp. 4156–4166, 2025.

Ziyi Lin, Chris Liu, Renrui Zhang, Peng Gao, Longtian Qiu, Han Xiao, Han Qiu, Chen Lin, Wenqi Shao, Keqin Chen, et al. Sphinx: The joint mixing of weights, tasks, and visual embeddings for multi-modal large language models. *arXiv preprint arXiv:2311.07575*, 2023.

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

Haotian Liu, Chunyuan Li, Yuheng Li, and Yong Jae Lee. Improved baselines with visual instruction tuning. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 26296–26306, 2024a.

Haotian Liu, Chunyuan Li, Yuheng Li, Bo Li, Yuanhan Zhang, Sheng Shen, and Yong Jae Lee. Llava-next: Improved reasoning, ocr, and world knowledge, January 2024b. URL https://llava-vl.github.io/blog/2024-01-30-llava-next/.

Gen Luo, Xue Yang, Wenhan Dou, Zhaokai Wang, Jiawen Liu, Jifeng Dai, Yu Qiao, and Xizhou Zhu. Mono-internvl: Pushing the boundaries of monolithic multimodal large language models with endogenous visual pre-training. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 24960–24971, 2025.

Oscar Mañas, Pau Rodriguez Lopez, Saba Ahmadi, Aida Nematzadeh, Yash Goyal, and Aishwarya Agrawal. Mapl: Parameter-efficient adaptation of unimodal pre-trained models for vision-language few-shot prompting. In *Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics*, pp. 2523–2548, 2023.

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

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

Lingchen Meng, Jianwei Yang, Rui Tian, Xiyang Dai, Zuxuan Wu, Jianfeng Gao, and Yu-Gang Jiang. Deepstack: Deeply stacking visual tokens is surprisingly simple and effective for lmms. *Advances in Neural Information Processing Systems*, 37:23464–23487, 2024.

Jack Merullo, Louis Castricato, Carsten Eickhoff, and Ellie Pavlick. Linearly mapping from image to text space. In *The Eleventh International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=8tYRqb05pVn.

Anand Mishra, Shashank Shekhar, Ajeet Kumar Singh, and Anirban Chakraborty. Ocr-vqa: Visual question answering by reading text in images. In *2019 international conference on document analysis and recognition (ICDAR)*, pp. 947–952. IEEE, 2019.

Clement Neo, Luke Ong, Philip Torr, Mor Geva, David Krueger, and Fazl Barez. Towards interpreting visual information processing in vision-language models. In *International Conference on Learning Representations*, volume 2025, pp. 57172–57189, 2025.

Namuk Park, Wonjae Kim, Byeongho Heo, Taekyung Kim, and Sangdoo Yun. What do self-supervised vision transformers learn? In *The Eleventh International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=azCKuYyS74.

Mustafa Shukor and Matthieu Cord. Implicit multimodal alignment: On the generalization of frozen llms to multimodal inputs. *Advances in Neural Information Processing Systems*, 37:130848–130886, 2024.

Mustafa Shukor, Corentin Dancette, and Matthieu Cord. ep-alm: Efficient perceptual augmentation of language models. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 22056–22069, 2023.

Peter Tong, Ellis Brown, Penghao Wu, Sanghyun Woo, Adithya Jairam Vedagiri Iyer, Sai Charitha Akula, Shusheng Yang, Jihan Yang, Manoj Middepogu, Ziteng Wang, et al. Cambrian-1: A fully open, vision-centric exploration of multimodal llms. *Advances in Neural Information Processing Systems*, 37:87310–87356, 2024.

Michael Tschannen, Alexey Gritsenko, Xiao Wang, Muhammad Ferjad Naeem, Ibrahim Alabdulmohsin, Nikhil Parthasarathy, Talfan Evans, Lucas Beyer, Ye Xia, Basil Mustafa, et al. Siglip 2: Multilingual vision-language encoders with improved semantic understanding, localization, and dense features. *arXiv preprint arXiv:2502.14786*, 2025.

Maria Tsimpoukelli, Jacob L Menick, Serkan Cabi, SM Eslami, Oriol Vinyals, and Felix Hill. Multimodal few-shot learning with frozen language models. *Advances in Neural Information Processing Systems*, 34:200–212, 2021.

Théophane Vallaeys, Mustafa Shukor, Matthieu Cord, and Jakob Verbeek. Improved baselines for data-efficient perceptual augmentation of llms. In Alessio Del Bue, Cristian Canton, Jordi Pont-Tuset, and Tatiana Tommasi (eds.), *Computer Vision – ECCV 2024 Workshops*, pp. 369–387, Cham, 2025. Springer Nature Switzerland. ISBN 978-3-031-92089-9.

Han Wang, Yongjie Ye, Bingru Li, Yuxiang Nie, Jinghui Lu, Jingqun Tang, Yanjie Wang, and Can Huang. Vision as lora. *arXiv preprint arXiv:2503.20680*, 2025.

Weihan Wang, Qingsong Lv, Wenmeng Yu, Wenyi Hong, Ji Qi, Yan Wang, Junhui Ji, Zhuoyi Yang, Lei Zhao, Xixuan Song, et al. Cogvlm: Visual expert for pretrained language models. *Advances in Neural Information Processing Systems*, 37:121475–121499, 2024.

An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, et al. Qwen2.5 technical report. *arXiv preprint arXiv:2412.15115*, 2025.

Jiabo Ye, Haiyang Xu, Haowei Liu, Anwen Hu, Ming Yan, Qi Qian, Ji Zhang, Fei Huang, and Jingren Zhou. mplug-owl3: Towards long image-sequence understanding in multi-modal large language models. In *International Conference on Learning Representations*, volume 2025, pp. 98891–98913, 2025.

Zhuoran Yu and Yong Jae Lee. How multimodal LLMs solve image tasks: A lens on visual grounding, task reasoning, and answer decoding. In *Second Conference on Language Modeling*, 2025. URL https://openreview.net/forum?id=kjNJYWvfPA.

Tongtian Yue, Longteng Guo, Yepeng Tang, Zijia Zhao, Xinxin Zhu, Hua Huang, and Jing Liu. Lavi: Efficient large vision-language models via internal feature modulation. *arXiv preprint arXiv:2506.16691*, 2025.

Boqiang Zhang, Kehan Li, Zesen Cheng, Zhiqiang Hu, Yuqian Yuan, Guanzheng Chen, Sicong Leng, Yuming Jiang, Hang Zhang, Xin Li, et al. Videollama 3: Frontier multimodal foundation models for image and video understanding. *arXiv preprint arXiv:2501.13106*, 2025a.

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

Shaolei Zhang, Qingkai Fang, Zhe Yang, and Yang Feng. LLaVA-mini: Efficient image and video large multimodal models with one vision token. In *The Thirteenth International Conference on Learning Representations*, 2025c. URL https://openreview.net/forum?id=UQJ7CDW8nb.

Xuange Zhang, Dengjie Li, Bo Liu, Zenghao Bao, Yao Zhou, Baisong Yang, Zhongying Liu, Yujie Zhong, and Tongtong Yuan. Layer-wise vision injection with disentangled attention for efficient lvlms. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 7004–7013, 2025d.

Zhi Zhang, Srishti Yadav, Fengze Han, and Ekaterina Shutova. Cross-modal information flow in multimodal large language models. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pp. 19781–19791, 2025e.

APPENDIX

APPENDIX CONTENTS

§A: Background

§A.1: Integration in MLLMs

§A.2: Previous Work in MLLM Analysis

§B: Implementation Details

§B.1: IN-CT

§B.2: LW-GC

§B.3: LW-AT

§B.4: IN-CT + LW-AT

§C: Training Details

§C.1: Input Image Processing

§C.2: Hyperparameters and Computing Infrastructure

§C.3: Training Dataset Details

§C.4: Evaluation Details

§D: Full Benchmark Results

§D.1: Benchmark Comparison and Effect of Text-Centric Data

§D.2: Hybrid Integration

§E: Additional Analysis Results

§E.1: Centered Kernel Alignment (CKA)

§E.2: Fourier Analysis

§E.3: Attention Mass

§E.4: PCA Visualizations

§F: Detailed Frequency Intervention Results

A BACKGROUND

A.1 INTEGRATION IN MLLMS

The architecture of Multimodal Large Language Models (MLLMs) is largely defined by how modality tokens are delivered to the LLM backbone. Existing approaches can be broadly categorized into three paradigms. In-context injection (Liu et al., 2024a; Zhang et al., 2025a) concatenates modality tokens with text tokens at the LLM input layer. Layer-wise injection introduces modality information at intermediate LLM layers through gated cross-attention (Alayrac et al., 2022; Laurençon et al., 2023), adaptive gating (Ye et al., 2025), LayerNorm injection (Yue et al., 2025), or direct injection into the LLM's attention (Alex et al., 2025; Zhang et al., 2025d). Hybrid approaches (Hong et al., 2024; Alex et al., 2025) feed compressed modality tokens as an in-context prompt while injecting finer-grained representations at intermediate layers. Orthogonal to these integration paradigms, other works explore visual feature enhancement, injecting features from different levels of the vision encoder into different LLM layers (Meng et al., 2024; Lin et al., 2025), or bypass the external vision encoder entirely through encoder-free approaches (Diao et al., 2025; Luo et al., 2025; Wang et al., 2025), processing raw pixels directly in the LLM. In this study, we focus on in-context and layer-wise injection within the vision-language setting, the two most widely employed integration strategies.

A.2 Previous Work in MLLM Analysis

A growing body of work analyzes how visual information is processed inside the LLM of MLLMs, revealing an implicit multimodal alignment despite a persistent modality gap (Shukor & Cord, 2024), visual representations becoming increasingly interpretable across layers via the logit lens (Neo et al., 2025; Jiang et al., 2025a;b), and middle layers critical for cross-modal interaction (Kaduri et al., 2025; Jiang et al., 2025b; Yu & Lee, 2025; Basu et al., 2024). Further studies show that visual tokens are primarily utilized in shallow layers, diminishing with depth (Chen et al., 2024), and exhibit a visual attention sink (Kang et al., 2025), while others map the cross-modal information flow (Zhang et al., 2025e; Kim et al., 2026). These works predominantly focus on a single integration paradigm, typically in-context injection. Closest to our study, Shukor et al. (2023) freeze most parameters and train only a linear projection with a single prepended token, Vallaeys et al. (2025) train only lightweight adaptation modules with a frozen LLM and encoder, and Laurençon et al. (2024) rely on LoRA fine-tuning. In contrast, our comparison performs full end-to-end fine-tuning of the vision encoder, projector, and LLM, reflecting modern VLM training practice. While these works compare integration mechanisms at the benchmark level, a systematic comparison of how different integration strategies affect visual information processing within the LLM remains unexplored.

B Implementation Details

All three paradigms share the same vision encoder (SigLIP2-So400m (Tschannen et al., 2025)) and connector architecture (MLP with GELU activation), and maintain an identical number of visual tokens to ensure that any observed performance differences are attributable solely to the integration mechanism. The implementation details of each paradigm are described below.

B.1 IN-CT

Our IN-CT implementation directly follows Liu et al. (2024a). Visual and text tokens are concatenated into a single sequence of length $N_v + N_t$ and processed jointly through all LLM layers, yielding an attention complexity of $\mathcal{O}((N_v + N_t)^2)$ and an FFN cost proportional to $N_v + N_t$ at every layer. Since $N_v \gg N_t$ in practice, both costs are dominated by the visual token count.

B.2 LW-GC

Our LW-GC implementation follows Alayrac et al. (2022), retaining the zero initialization of the learnable gates α_l and β_l in Eqs. (1) and (2) as in the original design. However, unlike the original Flamingo training procedure which keeps the LLM frozen, we unfreeze the LLM during supervised fine-tuning to ensure a fair comparison with IN-CT and LW-AT. During the projector pretraining stage, we train the MLP projector along with the gated cross-attention modules while keeping the LLM and vision encoder frozen. Additionally, we do not utilize a Perceiver (Jaegle et al., 2021) nor compress the number of visual tokens, maintaining an identical visual token count across all three paradigms. Since the initial hidden state contains only text tokens, the standard self-attention operates over N_t tokens with complexity $\mathcal{O}(N_t^2)$, while the gated cross-attention adds a cost of $\mathcal{O}(N_t \cdot N_v)$, avoiding the $\mathcal{O}(N_v^2)$ term present in IN-CT. We experiment with two insertion schedules, inserting blocks at every layer and at every fourth layer. Every-layer insertion consistently yields better performance, which we adopt as the default configuration in our main comparisons.

B.3 LW-AT

Our LW-AT implementation, defined in Equation (3), appends visual tokens to keys and values only, never forming queries, and bypasses the LLM's FFN sublayers entirely, as shown in Figure 8. This reduces the per-layer attention complexity from $\mathcal{O}((N_v + N_t)^2)$ in IN-CT to $\mathcal{O}((N_v + N_t) \cdot N_t)$, and since $N_v \gg N_t$ in practice, the savings are substantial. The FFN cost per layer is similarly reduced, as it operates over N_t tokens rather than $N_v + N_t$. Since this is an architectural rather than a training-only modification, these efficiency gains apply at both training and inference time. We follow Alex et al. (2025) in adopting their position ID scheme, where visual tokens are assigned the same position IDs in the key-value sequence as they would occupy in IN-CT, as shown in Figure 8, ensuring a direct and fair architectural comparison between LW-AT and IN-CT.

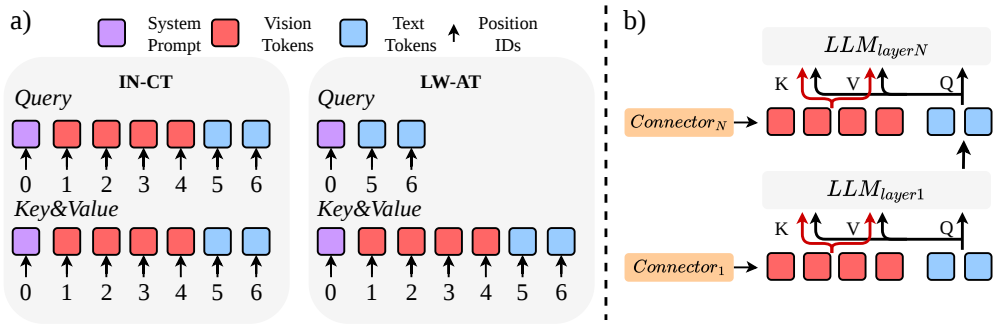


Figure 8: (a) Position ID scheme used in LW-AT compared to IN-CT. Visual tokens are assigned the same position IDs in the key-value sequence as they would occupy in IN-CT, preserving the relative positional relationship between visual and text tokens across both paradigms. (b) Architecture of LW-AT, where layer-specific connectors project visual features into the keys and values of each LLM layer, bypassing the FFN sublayers entirely.

B.4 IN-CT + LW-AT

We provide full implementation details for the hybrid model introduced in the main paper. The hybrid model (IN-CT + LW-AT) applies both integration strategies simultaneously. As illustrated in Figure 9, IN-CT vision tokens are concatenated with text tokens at the input following the standard in-context injection pipeline, while LW-AT vision tokens are additionally injected into the keys and values of each LLM layer. The two sets of visual tokens are processed by separate connectors but share the same vision encoder features. Figure 10 illustrates the resulting token sequence and position ID assignment. IN-CT vision tokens retain their original position IDs within the input sequence, while LW-AT vision tokens are assigned position IDs in the key-value sequence following the same scheme as the standalone LW-AT model, preserving the relative positional structure of each paradigm independently.

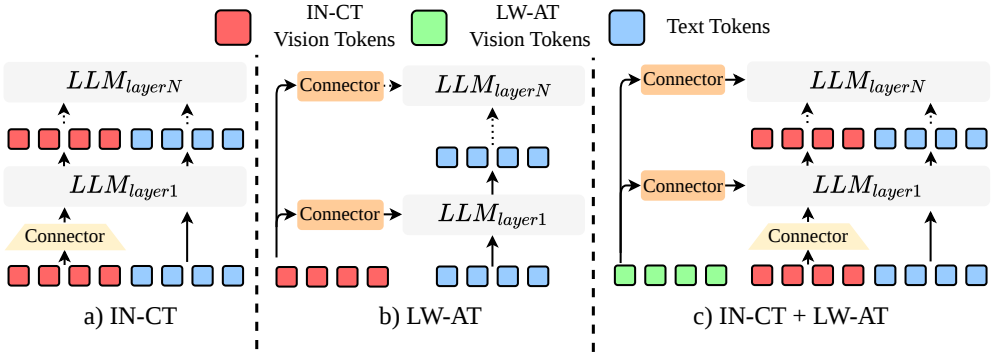


Figure 9: Architecture overview of the hybrid IN-CT + LW-AT model compared to standalone IN-CT and LW-AT. IN-CT vision tokens are concatenated at the input, while LW-AT vision tokens are injected into the keys and values at every layer.

C Training Details

C.1 Input Image Processing

For training with the 700K and text-centric-filtered 500K mixtures, which consist entirely of single images, we apply standard image resizing to 384×384. For training with the LLaVA-OV mixture, which combines high-resolution single images, multi-image, and video data, we follow the AnyRes

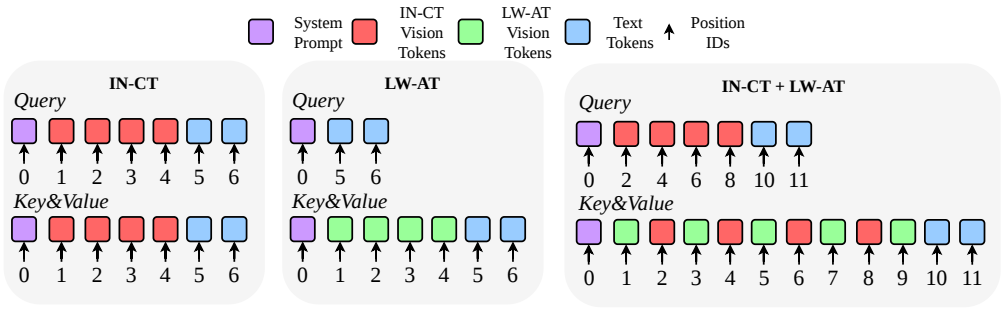


Figure 10: Token sequence and position ID assignment for IN-CT, LW-AT, and the hybrid IN-CT + LW-AT model. Each paradigm preserves its original position ID scheme independently within the hybrid architecture.

input processing pipeline from Li et al. (2025b). Specifically, high-resolution single images are processed using the same adaptive tiling strategy, while multi-image and video inputs follow the same frame sampling and sequence construction as in the original LLaVA-OV setup. This input processing pipeline is applied identically across all three integration paradigms (IN-CT, LW-AT, and LW-GC), ensuring that any observed performance differences are attributable solely to the integration mechanism rather than to differences in visual input handling.

C.2 Hyperparameters and Computing Infrastructure

All models are trained using the same two-stage pipeline and hyperparameter configuration, summarized in Table 1. All paradigms share identical settings across both stages, ensuring that any observed differences are attributable solely to the integration mechanism. All experiments are conducted on NVIDIA H200 GPUs, with the per-stage GPU allocation listed in the same table.

C.3 Training Dataset Details

For Stage 1 pretraining, we use LLaVA-Pretrain (BLIP-558K) (Liu et al., 2023), a standard image-caption dataset used to align the visual connector with the LLM. For Stage 2 supervised fine-tuning, we experiment with three mixtures. LLaVA-NeXT SFT (700K) (Liu et al., 2024b) is a general visual instruction tuning mixture covering diverse question answering, reasoning, and conversation tasks. The Text-Centric-Filtered (500K) mixture is derived from LLaVA-NeXT SFT by removing text-centric samples, including pure OCR datasets as well as samples containing substantial situated text such as OCR-VQA (Mishra et al., 2019), DocVQA (Mathew et al., 2021), AI2D (Kembhavi et al., 2016), ChartQA (Masry et al., 2022), DVQA (Kafle et al., 2018), and Synthdog (Kim et al., 2022), resulting in approximately 500K samples after filtering. LLaVA-OV (1.6M) (Li et al., 2025b) is a large-scale mixture covering single-image, multi-image, and video data.

C.4 Evaluation Details

All benchmarks are evaluated with the open source lmms-eval framework (Zhang et al., 2025b), using its default prompts and metrics for each benchmark. The same evaluation settings are applied to all models and all three integration paradigms.

D Full Benchmark Results

This section reports the exact numbers behind the benchmark figures in the main paper. All scores are obtained with the evaluation setup described in Appendix C.4.

Table 1: Training hyperparameters, configurations, and compute allocation for the two-stage pipeline. In Stage 1, only the connector is trained to align visual features. In Stage 2, the entire architecture (vision encoder, projector, and LLM) undergoes full-parameter instruction tuning. For Stage 2, we experiment with two SFT mixtures: LLaVA-NeXT (700K) and LLaVA-OV (1.6M).

Configuration	**Stage 1: Pretraining**	**Stage 2: Supervised Fine-Tuning**
Trainable modules		
Vision encoder (SigLIP2-So400m)	Frozen	Trainable
MLP projector	Trainable	Trainable
LLM (Qwen2.5 / Llama-3.2)	Frozen	Trainable
Training configuration		
Dataset	LLaVA-Pretrain	LLaVA-NeXT / LLaVA-OV
Global batch size	256	256
Learning rate (projector)	1×10^{-3}	1×10^{-5}
Learning rate (VE)	–	2×10^{-6}
Learning rate (LLM)	–	1×10^{-5}
Optimizer	AdamW	AdamW
Weight decay	0.0	0.0
LR schedule	Cosine decay	Cosine decay
Warmup ratio	0.03	0.03
Epochs	1	1
Distributed training	DeepSpeed ZeRO-2	DeepSpeed ZeRO-2
Image resolution	384×384	384×384 (LLaVA-NeXT) AnyRes (LLaVA-OV)
Precision	bfloat16	bfloat16
GPU	NVIDIA H200	NVIDIA H200
GPU count	4 (1 node)	16 (4 nodes)

D.1 Benchmark Comparison and Effect of Text-Centric Data

Figure 2 in the main paper summarizes benchmark performance as radar charts, where only the best score on each benchmark is annotated. Tables 2 and 3 report the exact scores of every integration paradigm for all backbones, model scales, and instruction tuning mixtures on single-image benchmarks and on multi-image and video benchmarks, respectively. In addition, Table 4 reports the scores after removing text-centric training data from the LLaVA-NeXT mixture, together with the change relative to the full mixture, corresponding to Figure 3 and Section 3.

Table 2: Full single-image benchmark results for all integration paradigms, backbones, and instruction tuning mixtures, grouped into General Knowledge, Vision-Centric, and OCR & Chart. Results are grouped by instruction tuning mixture (LLaVA-NeXT and LLaVA-OneVision). Within each backbone and mixture, **bold** and underline denote the best and second best results, respectively; second best is marked only for backbones with more than two integration methods. The best integration method for each backbone is also shaded.

LLM Backbone	Integration Method	General Knowledge					Vision Centric			OCR & Chart			
		MMB	SEED-I	GQA	SQA	MMMU	RW-QA	CV-2D	CV-3D	ChartQA	TextVQA	OCRBench	DocVQA
Instruction tuning on LLaVA-NeXT													
Qwen2.5-0.5B	**IN-CT**	**49.1**	**59.6**	**57.5**	**60.4**	**32.8**	**51.6**	**45.1**	**54.3**	**48.2**	**47.3**	**39.3**	**38.1**
	LW-AT	43.8	50.6	54.8	58.9	30.0	46.9	43.3	51.5	42.6	42.8	36.2	31.5
	LW-GC(4)	17.4	31.5	37.7	55.2	29.4	41.6	39.1	45.9	10.4	6.8	2.6	6.3
	LW-GC(all)	28.7	40.1	45.7	58.1	27.0	43.8	38.9	50.3	11.2	8.5	2.8	8.3
Qwen2.5-3B	**IN-CT**	**72.3**	**71.5**	**62.7**	**76.2**	**39.2**	**55.8**	**59.2**	57.4	**59.6**	**55.8**	**47.8**	**49.3**
	LW-AT	58.6	59.7	58.0	71.1	38.1	48.5	52.5	**58.6**	47.0	47.2	37.8	35.4
Qwen2.5-7B	**IN-CT**	**78.2**	**73.0**	**63.8**	**80.7**	**44.3**	**62.1**	**67.3**	**69.8**	**61.0**	**60.1**	**51.1**	**52.9**
	LW-AT	68.5	65.1	61.3	75.4	44.2	51.8	61.8	65.7	45.5	52.6	41.9	40.0
LLaMA-3.2-1B	**IN-CT**	**59.7**	**62.8**	**59.9**	**69.4**	**34.3**	**51.4**	**48.5**	**52.0**	**45.2**	**50.1**	**39.9**	**38.7**
	LW-AT	51.9	56.2	58.1	64.4	30.7	48.4	45.1	51.2	37.3	44.5	35.1	33.6
	LW-GC(4)	40.0	41.1	46.8	64.3	30.3	40.9	38.3	51.6	12.2	10.4	4.5	9.8
	LW-GC(all)	47.9	45.0	49.0	67.1	29.0	42.6	38.9	51.4	12.8	12.7	8.2	11.3
LLaMA-3.2-3B	**IN-CT**	**70.4**	**70.3**	**62.6**	**78.4**	**36.3**	**58.3**	**57.5**	**61.8**	**52.9**	**56.7**	**45.9**	**44.2**
	LW-AT	62.3	63.7	60.2	73.9	34.3	51.9	53.8	56.0	38.6	46.6	37.2	32.4
Instruction tuning on LLaVA-OneVision (OV)													
Qwen2.5-0.5B	**IN-CT**	**59.2**	**63.5**	**57.8**	70.0	**33.2**	**56.1**	**53.8**	**56.3**	**61.0**	**59.1**	**53.8**	**56.3**
	LW-AT	49.0	56.3	55.6	**70.4**	31.1	51.1	48.2	54.4	52.7	53.1	45.4	38.0
	LW-GC(all)	20.2	35.4	36.8	64.2	31.6	42.2	34.7	49.8	12.4	7.6	2.4	6.7
Qwen2.5-3B	**IN-CT**	**78.1**	**74.2**	**62.7**	**83.9**	**42.2**	**62.5**	**64.3**	**62.6**	**75.5**	**69.0**	**63.5**	**73.7**
	LW-AT	62.6	63.6	59.1	77.8	38.7	55.6	56.1	54.3	58.1	56.0	45.3	42.4

Table 3: Full multi-image and video benchmark results for all integration paradigms, backbones, and instruction tuning mixtures. Results are grouped by instruction tuning mixture (LLaVA-NeXT and LLaVA-OneVision). Within each backbone and mixture, **bold** and underline denote the best and second best results, respectively; second best is marked only for backbones with more than two integration methods. The best integration method for each backbone is also shaded.

LLM Backbone	Integration Method	Multi-Image				Video			
		MI-VQA	NLVR2	Mantis	MuirBench	MLVU	VideoMME	LVBench	NextQA
Instruction tuning on LLaVA-NeXT									
Qwen2.5-0.5B	**IN-CT**	**46.3**	**57.1**	39.5	**27.4**	**38.5**	**38.2**	**28.9**	**47.4**
	LW-AT	40.3	54.4	36.7	24.4	33.8	32.4	27.9	37.8
	LW-GC(4)	43.0	50.2	36.7	17.5	32.5	31.4	26.5	31.5
	LW-GC(all)	45.3	51.1	**40.4**	18.1	35.5	31.3	26.7	37.8
Qwen2.5-3B	**IN-CT**	**48.2**	**68.9**	**47.3**	**26.4**	**52.1**	**49.3**	**34.2**	**66.7**
	LW-AT	48.0	59.2	43.6	20.8	44.1	40.3	27.2	53.2
Qwen2.5-7B	**IN-CT**	**52.2**	**74.3**	43.8	25.2	**54.6**	**53.5**	**34.9**	**70.2**
	LW-AT	46.5	66.2	**46.9**	**25.8**	49.0	45.9	28.8	64.8
LLaMA-3.2-1B	**IN-CT**	**44.5**	**52.4**	**44.5**	**24.1**	41.7	**37.2**	**28.0**	**51.5**
	LW-AT	43.8	52.1	40.9	23.0	38.2	35.7	26.5	45.6
	LW-GC(4)	42.7	51.4	38.6	19.0	**42.0**	34.9	27.0	38.0
	LW-GC(all)	39.5	50.9	38.6	18.6	40.1	36.4	27.8	45.3
LLaMA-3.2-3B	**IN-CT**	**44.0**	**63.8**	**39.5**	**37.3**	**50.5**	**46.6**	**30.5**	**65.8**
	LW-AT	42.2	58.2	**39.5**	31.3	43.0	39.8	26.7	60.3
Instruction tuning on LLaVA-OneVision (OV)									
Qwen2.5-0.5B	**IN-CT**	**79.2**	**72.4**	**44.1**	**27.8**	**50.3**	**45.0**	**33.9**	**62.6**
	LW-AT	46.0	67.9	36.7	25.1	46.0	40.1	30.1	57.6
	LW-GC(all)	44.5	50.6	39.5	19.2	37.0	33.0	26.6	39.0
Qwen2.5-3B	**IN-CT**	**88.7**	**87.0**	**65.0**	**46.6**	**60.1**	**56.7**	**37.2**	**75.9**
	LW-AT	84.3	73.4	54.2	29.3	52.9	46.7	32.2	70.9

Table 4: Performance of integration paradigms on General Knowledge and Vision-Centric benchmarks when trained on the text-centric-filtered LLaVA-NeXT mixture. Parentheses show the change relative to the full LLaVA-NeXT mixture (green for improvement, red for decline), corresponding to Figure 3.

LLM Backbone	Integration Method	General Knowledge					Vision-Centric		
		MMB	SEED-I	GQA	SQA	MMMU	RW-QA	CV-2D	CV-3D
Instruction tuning on text-centric-filtered LLaVA-NeXT									
Qwen2.5-0.5B	IN-CT	47.4 (-1.7)	58.9 (-0.7)	57.2 (-0.3)	58.4 (-2.0)	29.9 (-2.9)	51.4 (-0.3)	44.1 (-1.0)	53.1 (-1.2)
	LW-AT	43.6 (-0.3)	50.6 (0.0)	54.8 (+0.1)	57.5 (-1.3)	28.1 (-1.9)	48.2 (+1.3)	43.5 (+0.1)	53.4 (+1.9)
	LW-GC(4)	17.2 (-0.3)	32.4 (+0.9)	37.9 (+0.3)	54.9 (-0.3)	27.8 (-1.7)	42.5 (+0.9)	38.5 (-0.6)	50.6 (+4.7)
	LW-GC(all)	30.8 (+2.1)	40.1 (0.0)	44.9 (-0.7)	57.5 (-0.6)	27.7 (+0.7)	47.1 (+3.3)	37.8 (-1.2)	50.0 (-0.3)
Qwen2.5-3B	IN-CT	72.1 (-0.2)	70.9 (-0.5)	62.4 (-0.3)	74.8 (-1.4)	38.6 (-0.7)	55.6 (-0.3)	58.7 (-0.5)	59.4 (+2.0)
	LW-AT	59.1 (+0.5)	58.7 (-1.0)	58.1 (+0.1)	70.3 (-0.8)	39.4 (+1.3)	47.3 (-1.2)	53.2 (+0.7)	56.6 (-2.0)
Qwen2.5-7B	IN-CT	76.8 (-1.4)	72.6 (-0.3)	63.7 (-0.2)	78.5 (-2.2)	45.4 (+1.1)	62.5 (+0.4)	67.0 (-0.2)	67.7 (-2.2)
	LW-AT	68.0 (-0.4)	65.7 (+0.6)	61.3 (0.0)	74.7 (-0.7)	43.1 (-1.1)	52.0 (+0.3)	60.4 (-1.5)	65.8 (+0.2)
LLaMA-3.2-1B	IN-CT	60.2 (+0.5)	62.6 (-0.2)	59.7 (-0.1)	68.6 (-0.8)	32.7 (-1.7)	51.5 (+0.1)	47.6 (-0.8)	51.8 (-0.2)
	LW-AT	52.2 (+0.3)	57.6 (+1.4)	58.1 (0.0)	65.8 (+1.4)	30.6 (-0.1)	50.5 (+2.1)	50.0 (+4.9)	50.3 (-0.9)
	LW-GC(4)	36.2 (-3.9)	41.3 (+0.2)	46.9 (+0.2)	63.0 (-1.2)	31.1 (+0.8)	42.8 (+1.8)	39.2 (+1.0)	50.5 (-1.1)
	LW-GC(all)	45.8 (-2.1)	44.3 (-0.7)	48.8 (-0.1)	65.4 (-1.7)	31.2 (+2.2)	43.7 (+1.0)	41.7 (+2.8)	51.3 (-0.2)
LLaMA-3.2-3B	IN-CT	70.9 (+0.5)	69.8 (-0.5)	63.0 (+0.4)	77.1 (-1.2)	34.8 (-1.5)	58.3 (0.0)	56.5 (-1.0)	59.8 (-2.0)
	LW-AT	64.9 (+2.6)	65.0 (+1.2)	61.1 (+0.9)	72.4 (-1.4)	34.6 (+0.2)	52.0 (+0.1)	53.3 (-0.5)	57.7 (+1.7)

D.2 HYBRID INTEGRATION

Table 5 reports the full benchmark results of the hybrid IN-CT + LW-AT model introduced in the ablation study, corresponding to Figure 7. Implementation details of the hybrid model are provided in Appendix B.4.

Table 5: Benchmark performance of the hybrid IN-CT + LW-AT model compared with the individual integration paradigms, all using the Qwen2.5-0.5B backbone trained on the LLaVA-NeXT mixture, corresponding to Figure 7. The hybrid applies both integration strategies simultaneously and therefore doubles the number of visual tokens. The last row reports the average (Avg) over all 20 benchmarks. **bold** and underline denote the best and second best results, respectively. The best integration method overall is also shaded.

Category	Benchmark	IN-CT	LW-AT	LW-GC(4)	LW-GC(all)	Hybrid (IN-CT + LW-AT)
General Knowledge	MMB	49.1	43.8	17.4	28.7	**54.1**
	SEED-I	**59.6**	50.6	31.5	40.1	58.7
	GQA	57.5	54.8	37.7	45.7	**57.8**
	SQA	**60.4**	58.9	55.2	58.1	**60.4**
	MMMU	**32.8**	30.0	29.4	27.0	31.3
Vision Centric	RW-QA	**51.6**	46.9	41.6	43.8	51.4
	CV-2D	**45.1**	43.3	39.1	38.9	44.4
	CV-3D	**54.3**	51.5	45.9	50.3	51.6
OCR & Chart	ChartQA	48.2	42.6	10.4	11.2	**48.7**
	TextVQA	47.3	42.8	6.8	8.5	**48.8**
	OCRBench	39.3	36.2	2.6	2.8	**41.1**
	DocVQA	38.1	31.5	6.3	8.3	**38.6**
Video	MLVU	38.5	33.8	32.5	35.5	**39.8**
	VideoMME	**38.2**	32.4	31.4	31.3	36.0
	LVBench	28.9	27.9	26.5	26.7	**29.1**
	NextQA	47.4	37.8	31.5	37.8	**48.2**
Multi-Image	MI-VQA	**46.3**	40.3	43.0	45.3	44.3
	NLVR2	57.1	54.4	50.2	51.1	**59.6**
	Mantis	39.5	36.7	36.7	**40.4**	**40.4**
	MuirBench	27.4	24.4	17.5	18.1	**28.4**
Avg		45.3	41.0	29.7	32.5	**45.6**

E Additional Analysis Results

This section provides additional results for the analyses in Section 4. Unless stated otherwise, the main paper uses the Qwen2.5-Instruct-0.5B backbone trained on the LLaVA-NeXT mixture; here we additionally report results for larger backbones.

E.1 Centered Kernel Alignment (CKA)

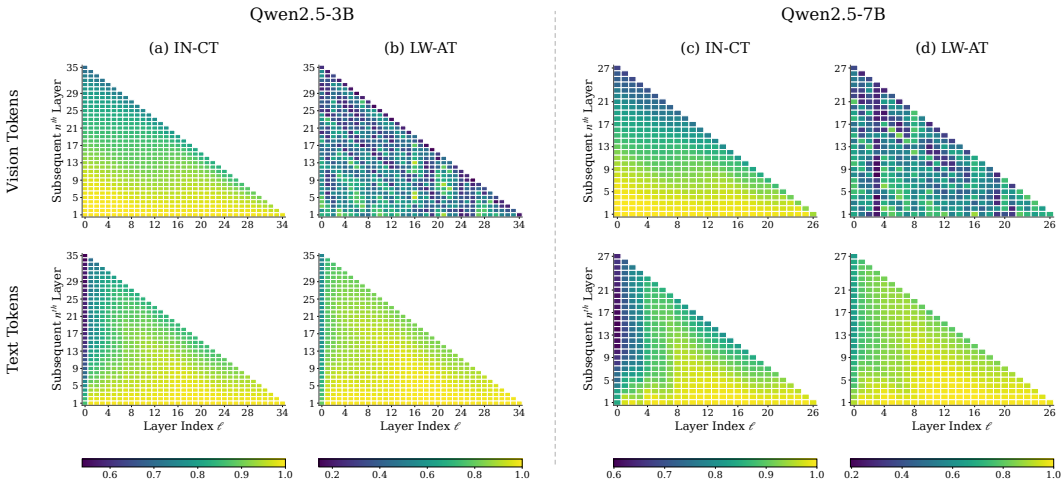


Figure 11: CKA heatmaps of visual and text token representations across layers for the Qwen2.5-3B and Qwen2.5-7B backbones on ChartQA. As with the 0.5B backbone, IN-CT evolves smoothly across layers, while the layer-wise variants show discontinuities in their visual tokens.

We employ Centered Kernel Alignment (CKA) (Kornblith et al., 2019) to measure how token representations evolve across layers. Here we provide the formal definition.

Formal definition. Given two representation matrices $X \in \mathbb{R}^{T\times D}$ and $Y \in \mathbb{R}^{T\times D}$, where T is the number of tokens and D is the hidden dimension, we define their kernel matrices as $K = XX^\top$ and $L = YY^\top$ using a linear kernel. CKA is then computed via the Hilbert-Schmidt Independence Criterion (HSIC):

$$\mathrm{CKA}(K, L) = \frac{\mathrm{HSIC}(K, L)}{\sqrt{\mathrm{HSIC}(K, K) \cdot \mathrm{HSIC}(L, L)}} \tag{6}$$

where HSIC measures the statistical dependence between the two feature spaces. For centered kernel matrices $\tilde{K} = HKH$ and $\tilde{L} = HLH$, with $H = I_T - \frac{1}{T}\mathbf{1}\mathbf{1}^\top$ being the centering matrix, the empirical HSIC is given by:

$$\mathrm{HSIC}(K, L) = \frac{1}{(T-1)^2}\,\mathrm{tr}(\tilde{K}\tilde{L}) \tag{7}$$

We apply CKA at the token level to measure how the internal structure of visual or text token representations shifts across layers. For a batch of N samples, let $H_i^{(\ell)} \in \mathbb{R}^{T\times D}$ denote the hidden states of sample i at layer ℓ. We treat the T tokens as data points and compute:

$$\mathrm{CKA}_{\text{token}}(\ell, \ell') = \frac{1}{N}\sum_{i=1}^{N} \mathrm{CKA}(H_i^{(\ell)}, H_i^{(\ell')}) \tag{8}$$

A high $\mathrm{CKA}_{\text{token}}(\ell, \ell')$ indicates that the pairwise relationships among tokens are largely preserved between layers ℓ and ℓ', whereas a low value signals a structural reorganization of the token representations.

Beyond the Qwen2.5-0.5B backbone used in the main paper, Figure 11 reports the CKA heatmaps for the Qwen2.5-3B and Qwen2.5-7B backbones. Both scales follow the same trend as the 0.5B

backbone, with visual tokens in IN-CT evolving smoothly across layers, while those in the layer-wise variants show clear discontinuities.

E.2 Fourier Analysis

Section 4.2 of the main paper reports the relative log amplitude of Fourier-transformed visual token representations for the Qwen2.5-0.5B backbone. Figures 12 and 13 report the same analysis for the Qwen2.5-3B and Qwen2.5-7B backbones, respectively. Both scales follow the same trend as the 0.5B backbone, with IN-CT capturing higher-frequency features than the layer-wise variants.

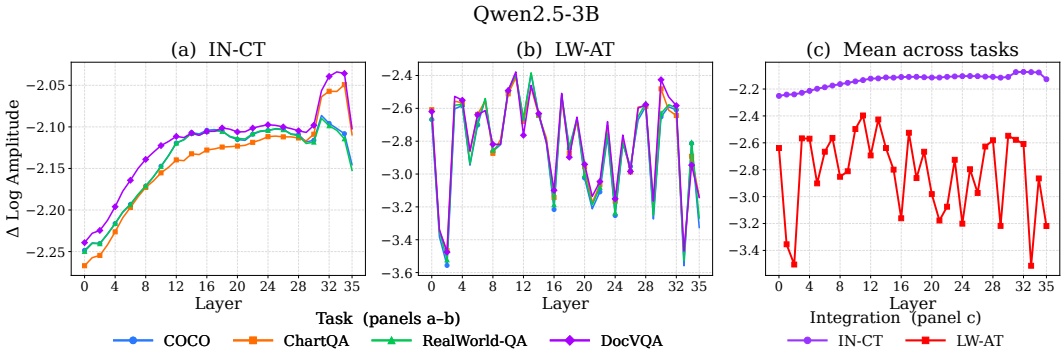


Figure 12: Relative log amplitude of Fourier-transformed visual token representations across layers for the Qwen2.5-3B backbone. As with the 0.5B backbone, IN-CT holds the highest amplitude, while the layer-wise variants remain biased toward low-frequency features.

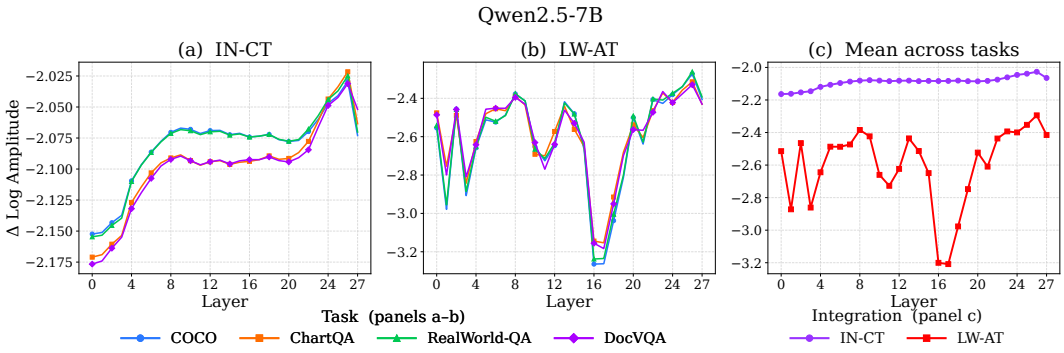


Figure 13: Relative log amplitude of Fourier-transformed visual token representations across layers for the Qwen2.5-7B backbone. As with the 0.5B backbone, IN-CT holds the highest amplitude, while the layer-wise variants remain biased toward low-frequency features.

E.3 Attention Mass

Section 4.4 of the main paper reports attention mass for IN-CT and LW-AT on one general and one text-centric benchmark with the Qwen2.5-0.5B backbone. Figure 14 extends the same analysis to additional benchmarks. The task-dependent pattern holds throughout. Attention to visual tokens stays low across all layers on non-OCR benchmarks but remains substantially higher through the middle and deep layers on text-centric ones. The two paradigms allocate attention almost identically, reinforcing that the performance gap between them reflects representation quality rather than how much attention the visual tokens receive. Figure 15 shows the same analysis for the Qwen2.5-3B backbone, which follows the same trend.

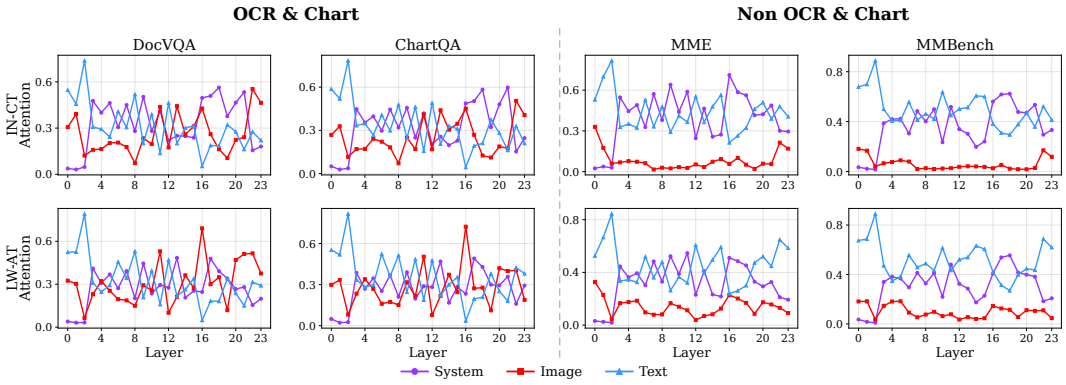


Figure 14: Attention mass allocated to visual tokens during generation across layers on additional benchmarks for both IN-CT and LW-AT. Consistent with the main paper, attention patterns are task-dependent, with text-centric tasks maintaining substantially higher attention to visual tokens through the deeper layers than non-OCR tasks.

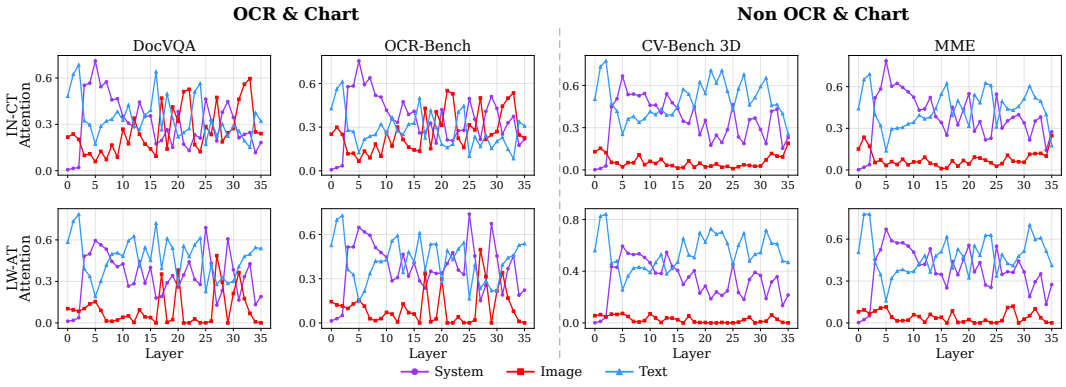


Figure 15: Attention mass allocated to visual tokens during generation across layers for both IN-CT and LW-AT with the Qwen2.5-3B backbone. As with the 0.5B backbone, attention patterns are task-dependent, and the two paradigms allocate attention similarly.

E.4 PCA VISUALIZATIONS

Figure 5 in the main paper shows PCA projections at five representative layers. Figures 16 to 18 show every layer for each paradigm with the Qwen2.5-0.5B backbone. The convergence in IN-CT is visible across the full depth rather than only at the layers selected for display, and no such convergence appears at any depth in either layer-wise variant. Figures 19, to 22 show the same projections for IN-CT and LW-AT with the Qwen2.5-3B backbone, which follow the same trend.

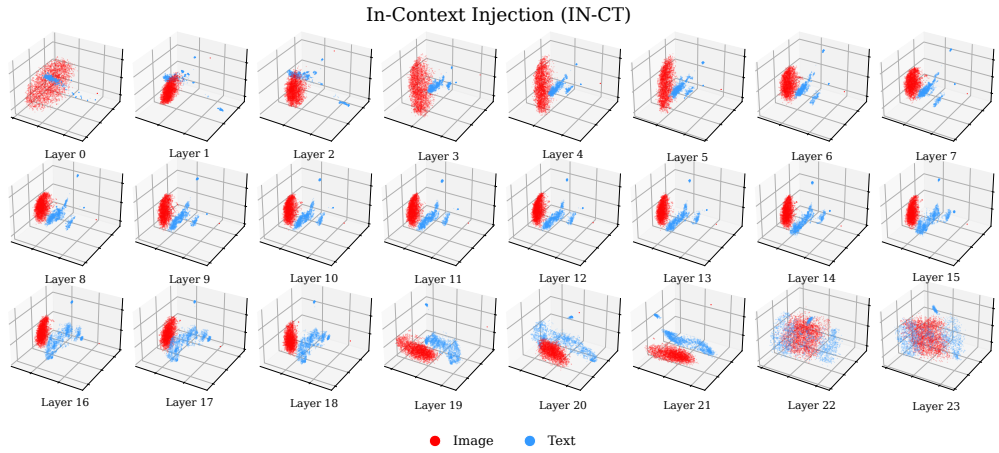


Figure 16: 3D PCA projections of image (red) and text (blue) token representations at every layer for IN-CT with the Qwen2.5-0.5B backbone. Visual tokens progressively converge toward the text token subspace, with clear merging in the final layers.

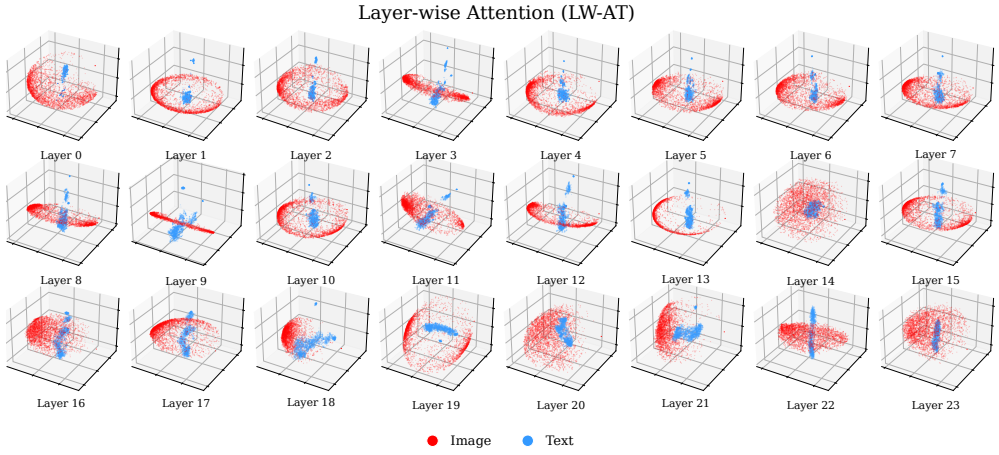


Figure 17: 3D PCA projections of image (red) and text (blue) token representations at every layer for LW-AT with the Qwen2.5-0.5B backbone. Visual representations remain separated from the text subspace throughout all layers.

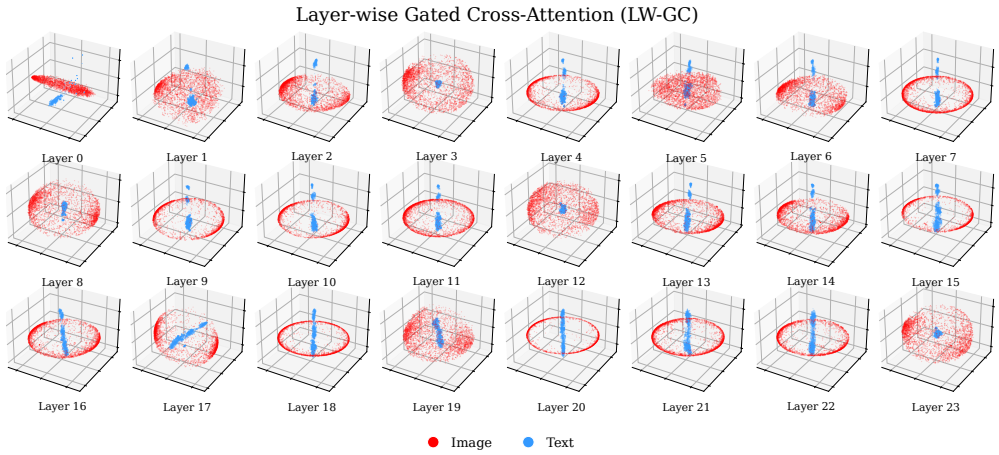


Figure 18: 3D PCA projections of image (red) and text (blue) token representations at every layer for LW-GC with the Qwen2.5-0.5B backbone. As with LW-AT, visual and text representations occupy distinct subspaces throughout the network.

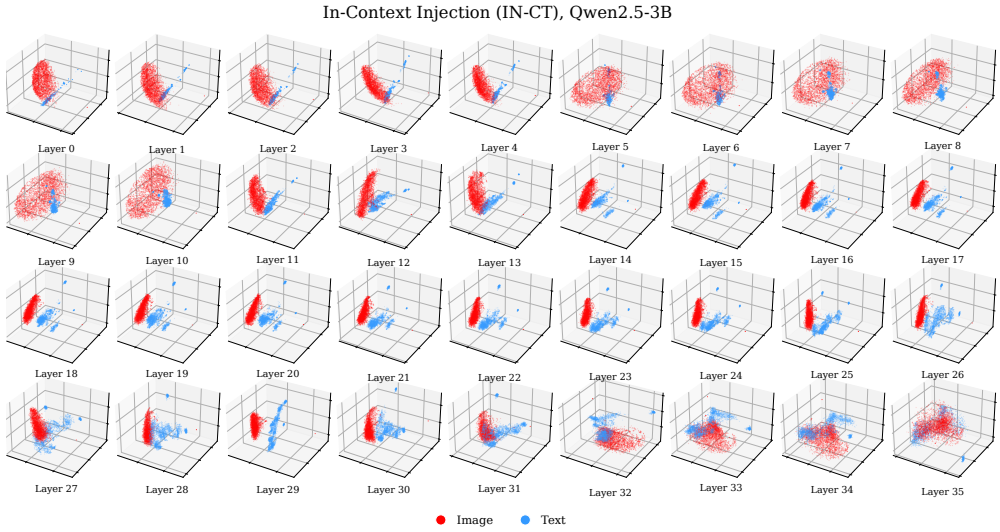


Figure 19: 3D PCA projections of image (red) and text (blue) token representations at every layer for IN-CT with the Qwen2.5-3B backbone. As with the 0.5B backbone, visual tokens progressively converge toward the text token subspace in the final layers.

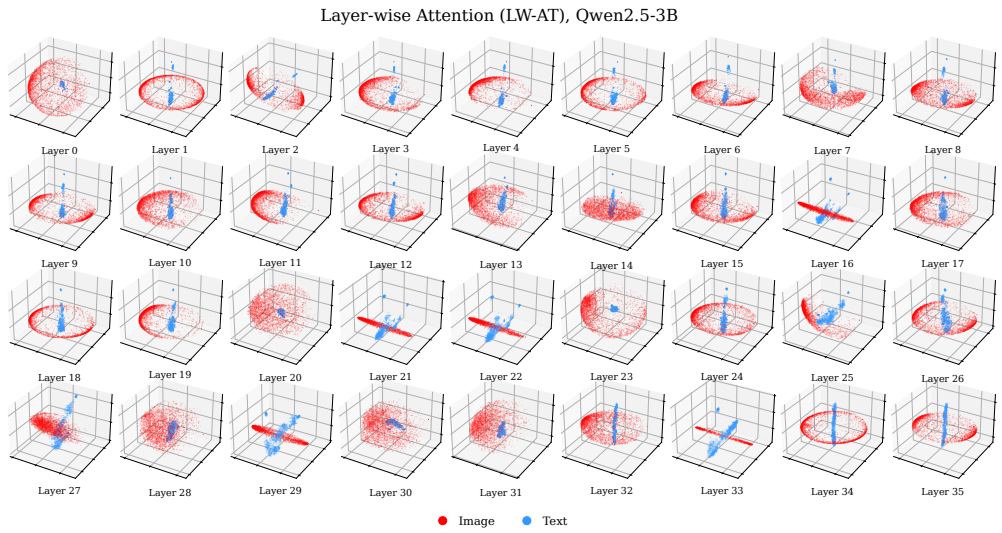


Figure 20: 3D PCA projections of image (red) and text (blue) token representations at every layer for LW-AT with the Qwen2.5-3B backbone. As with the 0.5B backbone, visual representations remain separated from the text subspace throughout all layers.

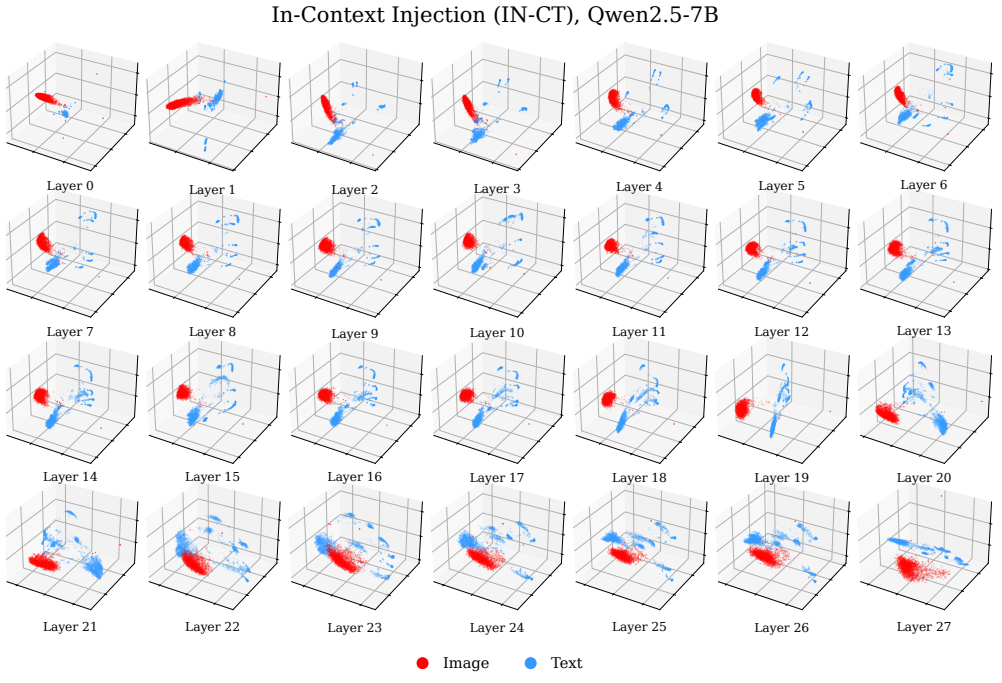


Figure 21: 3D PCA projections of image (red) and text (blue) token representations at every layer for IN-CT with the Qwen2.5-7B backbone. As with the 0.5B backbone, visual tokens progressively converge toward the text token subspace in the final layers, ending much closer to it than in LW-AT.

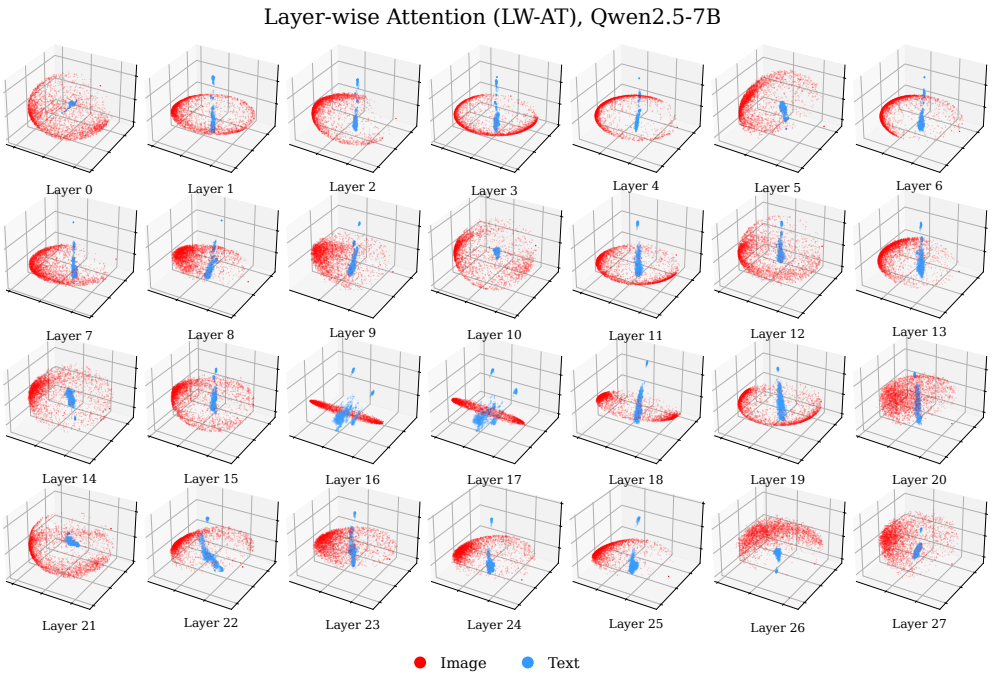


Figure 22: 3D PCA projections of image (red) and text (blue) token representations at every layer for LW-AT with the Qwen2.5-7B backbone. As with the 0.5B backbone, visual representations remain separated from the text subspace throughout all layers.

F Detailed Frequency Intervention Results

Section 4.5 of the main paper suppresses the high-frequency content of visual representations by applying a low-pass filter to one decoder depth group at a time, and Figure 6 summarizes the resulting mean performance change per task category. Tables 6, 7, and 8 report the exact per-benchmark scores for OCR & Chart, General Knowledge, and Vision-Centric benchmarks, respectively.

Table 6: Effect of low-pass filtering the visual token hidden states on OCR & Chart benchmarks, corresponding to Figure 6(a). For each model, the shaded *No filter* row reports the unfiltered score and each subsequent row applies the filter (cutoff $\rho = 0.9$) to one decoder depth group. Parentheses show the change relative to no filtering (green for improvement, red for decline).

LLM	Integration Method	Filtered Depth	ChartQA	TextVQA	OCRBench	DocVQA
Qwen2.5-0.5B	IN-CT	No filter	48.2	47.3	39.3	38.1
		Early	46.8 (-1.4)	47.8 (+0.5)	38.4 (-0.9)	37.6 (-0.5)
		Early-Mid	45.2 (-3.0)	46.1 (-1.2)	37.2 (-2.1)	36.6 (-1.5)
		Late-Mid	46.6 (-1.6)	46.0 (-1.3)	37.1 (-2.2)	36.5 (-1.6)
		Late	48.0 (-0.2)	48.4 (+1.1)	38.9 (-0.4)	37.9 (-0.2)
	LW-AT	No filter	42.6	42.8	36.2	31.5
		Early	42.3 (-0.3)	42.3 (-0.5)	36.2 (0.0)	31.5 (0.0)
		Early-Mid	42.2 (-0.4)	42.4 (-0.4)	36.6 (+0.4)	31.3 (-0.2)
		Late-Mid	42.8 (+0.2)	42.9 (+0.1)	36.1 (-0.1)	31.1 (-0.4)
		Late	42.4 (-0.2)	43.1 (+0.3)	36.1 (-0.1)	31.6 (+0.1)
	LW-GC(all)	No filter	11.2	8.5	2.8	8.3
		Early	11.2 (0.0)	8.7 (+0.2)	2.8 (0.0)	8.3 (0.0)
		Early-Mid	11.3 (+0.1)	8.2 (-0.3)	2.9 (+0.1)	8.3 (0.0)
		Late-Mid	11.4 (+0.2)	8.2 (-0.3)	2.9 (+0.1)	8.3 (0.0)
		Late	11.2 (0.0)	8.8 (+0.3)	2.9 (+0.1)	8.3 (0.0)
Qwen2.5-3B	IN-CT	No filter	59.6	55.8	47.8	49.3
		Early	58.6 (-1.0)	55.9 (+0.1)	47.8 (0.0)	48.7 (-0.6)
		Early-Mid	57.6 (-2.0)	55.6 (-0.2)	46.7 (-1.1)	47.3 (-2.0)
		Late-Mid	56.3 (-3.3)	55.4 (-0.4)	47.2 (-0.6)	47.2 (-2.1)
		Late	59.2 (-0.4)	56.0 (+0.2)	47.9 (+0.1)	48.6 (-0.7)
	LW-AT	No filter	47.0	47.2	37.8	35.4
		Early	47.0 (0.0)	46.8 (-0.4)	37.8 (0.0)	35.2 (-0.2)
		Early-Mid	47.0 (0.0)	46.9 (-0.3)	37.4 (-0.4)	35.2 (-0.2)
		Late-Mid	46.2 (-0.8)	46.9 (-0.3)	37.5 (-0.3)	35.3 (-0.1)
		Late	45.5 (-1.5)	47.1 (-0.1)	37.1 (-0.7)	34.1 (-1.3)
Qwen2.5-7B	IN-CT	No filter	61.0	60.1	51.1	52.9
		Early	59.7 (-1.3)	59.2 (-0.9)	51.2 (+0.1)	51.5 (-1.4)
		Early-Mid	57.3 (-3.7)	58.7 (-1.4)	48.6 (-2.5)	50.2 (-2.7)
		Late-Mid	57.5 (-3.5)	58.1 (-2.0)	49.1 (-2.0)	50.0 (-2.9)
		Late	60.5 (-0.5)	59.9 (-0.2)	50.2 (-0.9)	52.1 (-0.8)
	LW-AT	No filter	45.5	52.6	41.9	40.0
		Early	45.5 (0.0)	52.2 (-0.4)	41.9 (0.0)	39.9 (-0.1)
		Early-Mid	44.8 (-0.7)	51.8 (-0.8)	41.4 (-0.5)	39.2 (-0.8)
		Late-Mid	45.2 (-0.3)	52.6 (0.0)	41.6 (-0.3)	39.8 (-0.2)
		Late	44.4 (-1.1)	51.4 (-1.2)	41.1 (-0.8)	38.9 (-1.1)

Table 7: Effect of low-pass filtering the visual token hidden states on General Knowledge benchmarks, corresponding to Figure 6(b). For each model, the shaded *No filter* row reports the unfiltered score and each subsequent row applies the filter (cutoff $\rho = 0.9$) to one decoder depth group. Parentheses show the change relative to no filtering (green for improvement, red for decline).

LLM	Integration Method	Filtered Depth	MMB	SEED-I	GQA	SQA	MMMU	AI2D
Qwen2.5-0.5B	IN-CT	No filter	49.1	59.6	57.5	60.4	32.8	53.3
		Early	49.1 (0.0)	59.4 (-0.2)	57.6 (+0.1)	60.1 (-0.3)	31.1 (-1.7)	53.0 (-0.3)
		Early-Mid	47.8 (-1.3)	58.8 (-0.8)	57.2 (-0.3)	61.4 (+1.0)	30.9 (-1.9)	52.8 (-0.5)
		Late-Mid	49.5 (+0.4)	59.5 (-0.1)	57.4 (-0.1)	60.5 (+0.1)	31.7 (-1.1)	53.3 (0.0)
		Late	49.1 (0.0)	59.2 (-0.4)	57.5 (0.0)	60.3 (-0.1)	31.1 (-1.7)	53.3 (0.0)
	LW-AT	No filter	43.8	50.6	54.8	58.9	30.0	52.5
		Early	43.9 (+0.1)	50.7 (+0.1)	54.6 (-0.2)	58.5 (-0.4)	31.6 (+1.6)	52.1 (-0.4)
		Early-Mid	43.6 (-0.2)	50.8 (+0.2)	54.5 (-0.3)	59.0 (+0.1)	28.8 (-1.2)	51.5 (-1.0)
		Late-Mid	43.7 (-0.1)	51.1 (+0.5)	54.6 (-0.2)	58.9 (0.0)	30.4 (+0.4)	52.2 (-0.3)
		Late	43.9 (+0.1)	50.8 (+0.2)	54.7 (-0.1)	58.9 (0.0)	30.1 (+0.1)	52.3 (-0.2)
	LW-GC(all)	No filter	28.7	40.1	45.7	58.1	27.0	51.0
		Early	28.0 (-0.7)	39.3 (-0.8)	45.6 (-0.1)	58.1 (0.0)	29.0 (+2.0)	50.7 (-0.3)
		Early-Mid	28.5 (-0.2)	39.4 (-0.7)	45.6 (-0.1)	58.2 (+0.1)	28.9 (+1.9)	50.9 (-0.1)
		Late-Mid	28.5 (-0.2)	39.6 (-0.5)	45.6 (-0.1)	58.2 (+0.1)	30.1 (+3.1)	50.8 (-0.2)
		Late	28.4 (-0.3)	39.6 (-0.5)	45.6 (-0.1)	58.1 (0.0)	30.1 (+3.1)	50.8 (-0.2)
Qwen2.5-3B	IN-CT	No filter	72.3	71.5	62.7	76.2	39.2	72.4
		Early	72.3 (0.0)	71.6 (+0.1)	62.8 (+0.1)	76.4 (+0.2)	38.9 (-0.3)	72.7 (+0.3)
		Early-Mid	72.2 (-0.1)	70.7 (-0.8)	62.7 (0.0)	75.5 (-0.7)	40.2 (+1.0)	71.3 (-1.1)
		Late-Mid	72.3 (0.0)	71.0 (-0.5)	62.4 (-0.3)	75.9 (-0.3)	38.2 (-1.0)	71.2 (-1.2)
		Late	72.3 (0.0)	71.3 (-0.2)	62.6 (-0.1)	76.2 (0.0)	38.6 (-0.6)	72.3 (-0.1)
	LW-AT	No filter	58.6	59.7	58.0	71.1	38.1	65.5
		Early	58.6 (0.0)	59.9 (+0.2)	58.0 (0.0)	71.1 (0.0)	37.6 (-0.5)	65.5 (0.0)
		Early-Mid	58.2 (-0.4)	59.1 (-0.6)	57.8 (-0.2)	70.0 (-1.1)	38.4 (+0.3)	65.0 (-0.5)
		Late-Mid	58.8 (+0.2)	59.7 (0.0)	58.0 (0.0)	70.8 (-0.3)	39.8 (+1.7)	65.3 (-0.2)
		Late	58.6 (0.0)	59.7 (0.0)	58.0 (0.0)	71.3 (+0.2)	37.1 (-1.0)	65.4 (-0.1)
Qwen2.5-7B	IN-CT	No filter	78.2	73.0	63.8	80.7	44.3	77.3
		Early	78.5 (+0.3)	72.5 (-0.5)	63.9 (+0.1)	80.7 (0.0)	44.4 (+0.1)	76.7 (-0.6)
		Early-Mid	76.4 (-1.8)	71.8 (-1.2)	63.3 (-0.5)	79.1 (-1.6)	43.6 (-0.7)	76.1 (-1.2)
		Late-Mid	76.7 (-1.5)	71.8 (-1.2)	63.3 (-0.5)	79.2 (-1.5)	47.4 (+3.1)	75.7 (-1.6)
		Late	78.3 (+0.1)	72.7 (-0.3)	63.8 (0.0)	80.8 (+0.1)	44.9 (+0.6)	77.3 (0.0)
	LW-AT	No filter	68.5	65.1	61.3	75.4	44.2	69.3
		Early	68.5 (0.0)	65.1 (0.0)	61.2 (-0.1)	75.7 (+0.3)	41.2 (-3.0)	69.3 (0.0)
		Early-Mid	68.8 (+0.3)	65.0 (-0.1)	61.2 (-0.1)	74.7 (-0.7)	43.6 (-0.6)	69.1 (-0.2)
		Late-Mid	69.0 (+0.5)	65.5 (+0.4)	61.0 (-0.3)	75.3 (-0.1)	42.4 (-1.8)	69.5 (+0.2)
		Late	68.8 (+0.3)	65.3 (+0.2)	61.2 (-0.1)	75.1 (-0.3)	44.2 (0.0)	69.2 (-0.1)

Table 8: Effect of low-pass filtering the visual token hidden states on Vision-Centric benchmarks, corresponding to Figure 6(b). For each model, the shaded *No filter* row reports the unfiltered score and each subsequent row applies the filter (cutoff $\rho = 0.9$) to one decoder depth group. Parentheses show the change relative to no filtering (green for improvement, red for decline).

LLM	Integration Method	Filtered Depth	RW-QA	CV-2D	CV-3D
Qwen2.5-0.5B	IN-CT	No filter	51.6	45.1	54.3
		Early	50.9 (-0.7)	45.3 (+0.2)	52.4 (-1.9)
		Early-Mid	50.5 (-1.1)	46.7 (+1.6)	52.0 (-2.3)
		Late-Mid	51.0 (-0.6)	45.1 (0.0)	54.5 (+0.2)
		Late	51.8 (+0.2)	44.7 (-0.4)	53.5 (-0.8)
	LW-AT	No filter	46.9	43.3	51.5
		Early	46.8 (-0.1)	40.9 (-2.4)	52.5 (+1.0)
		Early-Mid	47.5 (+0.6)	42.4 (-0.9)	52.5 (+1.0)
		Late-Mid	47.6 (+0.7)	42.9 (-0.4)	53.1 (+1.6)
		Late	47.5 (+0.6)	43.5 (+0.2)	53.8 (+2.3)
	LW-GC(all)	No filter	43.8	38.9	50.3
		Early	44.3 (+0.5)	40.1 (+1.2)	50.4 (+0.1)
		Early-Mid	44.1 (+0.3)	37.5 (-1.4)	51.3 (+1.0)
		Late-Mid	44.2 (+0.4)	38.3 (-0.6)	50.9 (+0.6)
		Late	44.3 (+0.5)	38.5 (-0.4)	50.1 (-0.2)
Qwen2.5-3B	IN-CT	No filter	55.8	59.2	57.4
		Early	56.5 (+0.7)	56.5 (-2.7)	60.6 (+3.2)
		Early-Mid	56.0 (+0.2)	56.3 (-2.9)	56.4 (-1.0)
		Late-Mid	55.8 (0.0)	56.4 (-2.8)	55.8 (-1.6)
		Late	56.3 (+0.5)	57.0 (-2.2)	59.3 (+1.9)
	LW-AT	No filter	48.5	52.5	58.6
		Early	49.3 (+0.8)	53.2 (+0.7)	56.0 (-2.6)
		Early-Mid	49.4 (+0.9)	52.9 (+0.4)	56.2 (-2.4)
		Late-Mid	49.3 (+0.8)	52.4 (-0.1)	53.8 (-4.8)
		Late	49.3 (+0.8)	52.5 (0.0)	55.1 (-3.5)
Qwen2.5-7B	IN-CT	No filter	62.1	67.3	69.8
		Early	62.5 (+0.4)	67.3 (0.0)	67.5 (-2.3)
		Early-Mid	61.6 (-0.5)	66.4 (-0.9)	69.0 (-0.8)
		Late-Mid	61.8 (-0.3)	65.5 (-1.8)	68.2 (-1.6)
		Late	62.6 (+0.5)	68.4 (+1.1)	70.6 (+0.8)
	LW-AT	No filter	51.8	61.8	65.7
		Early	52.2 (+0.4)	61.1 (-0.7)	67.7 (+2.0)
		Early-Mid	54.3 (+2.5)	62.5 (+0.7)	66.7 (+1.0)
		Late-Mid	52.6 (+0.8)	60.5 (-1.3)	65.2 (-0.5)
		Late	51.8 (0.0)	61.0 (-0.8)	66.3 (+0.6)