STOP DESCRIBING, START ANSWERING: REDUCING DESCRIPTION RELIANCE IN VISION-LANGUAGE MODELS

Anonymous authors
Paper under double-blind review

ABSTRACT

Existing Vision-Language Models (VLMs) have an undesirable habit in visual question answering: models often rely on image descriptions, first describing the visual content and then answering the question. This describe-then-answer pattern can improve accuracy, but it also produces redundant tokens and increases latency. When forced to answer directly, VLMs give shorter but often less accurate answers. We call this gap `description reliance`. To reduce this reliance without generating descriptions at inference, we propose `Internalization`, a two-stage training method that uses caption-derived answer supervision to train a compact image-question prior for direct answering. Stage I aligns the new prior module with the base model distribution through logits-wise warm-up, and Stage II injects prior tokens into hidden states through preference training, using the Stage I model as a teacher to reduce distribution drift. We further introduce `InternVQA`, a benchmark that measures description reliance across image settings, task types, and answer lengths. Experiments on Qwen3-VL and LLaVA-OneVision show that `Internalization` raises the average direct-answer score from 40.21 to 63.11 and shrinks the caption-direct gap from 16.62 to 3.98 judge points while increasing the caption score from 56.83 to 67.09. It also achieves 96.45% aggregate retention across ten public benchmarks and shortens end-to-end inference time by 23.86% by generating fewer tokens, showing that caption-derived supervision can improve direct answering through compact prior tokens instead of generated descriptions.

1 INTRODUCTION

With the rise of Large Language Models (LLMs) (Kumar, 2024), Vision Language Models (VLMs) connect image encoders with text decoders through large scale pre-training, enabling image based reasoning across multiple tasks (Li et al., 2025b). Yet in visual question answering (VQA), a common pattern remains: many existing VLMs do not answer directly. Instead, the models often generate an image description first and then answer the question (Wang et al., 2025c). Though this extra description can improve VQA accuracy, it increases output tokens and latency. When prompts enforce direct answering, VQA performance drops. Fig. 1 illustrates this phenomenon on Qwen3-VL-2B. For the same image and question, caption-then-answering gives the correct answer, while direct answering fails. This contrast motivates learning to answer directly from the image and question using supervision from caption-then-answering, following the context-distillation principle (Snell et al., 2022; Yu et al., 2024). We operationalize **description reliance** as the answer-score gap between caption-then-answering and direct answering under the specified prompts. **We train an image-question prior to transfer this protocol's answer benefit into direct answering.** Here, internalization denotes this functional transfer: higher Direct scores and a smaller caption-direct gap without generating a preliminary description.

Existing methods replace explicit descriptions with other intermediate evidence (Sec. 2). **Caption-based** methods generate textual evidence (Hu et al., 2023; Özdemir & Akagündüz, 2024). **Post-training** methods use preferences or rewards (Zhang et al., 2025e; Liu et al., 2025c). **Decoding interventions** adjust visual attention or decoding (Liu et al., 2024; Huang et al., 2024), while **visual-tool** methods add image operations (Wu et al., 2025a; Fan et al., 2025). **Latent reasoning** methods

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.

move reasoning into hidden states, reducing visible tokens but adding internal computation (Huang et al., 2025b; Hao et al., 2024).

To overcome these limitations, we propose Internalization around a simple idea: using caption-derived answer supervision to train a compact image-question prior for direct answering. We inject newly learned prior tokens into the original token representations before the language model head, which changes the output logits and moves the model away from the description first habit toward direct answers. We train Internalization in two stages:

Stage I (logits-wise warm-up). Directly inserting newly initialized prior tokens into hidden states can disturb the base model distribution and make early training unstable, so Stage I uses a simpler objective: logits correction. We treat image descriptions as question-aware summaries of the image content. To approximate this effect without generating text, we extract image and question tokens from the final decoder hidden states, compress them into a compact prior representation, and let it predict a correction to the original logits, teaching the new module how question-aware visual evidence should shift the answer preference.

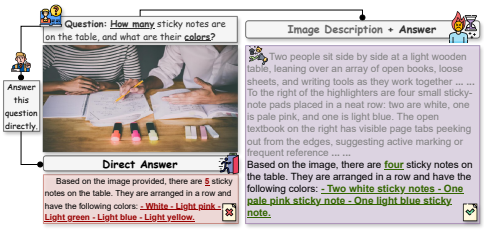


Figure 1: Direct answering gives an incorrect response (red), whereas caption-then-answering recovers the correct answer (green).

Stage II (hidden-space internalization). Stage II turns the Stage I representation into hidden-space priors for direct answering: the Stage I modules generate K prior hidden states, injected into the original hidden states through gated attention before the language model head, so the model answers without describing the image. The priors are trained on preference pairs generated by the Stage I model: the rejected response comes from an unconstrained answering prompt, and the chosen response from a caption-then-answer prompt with the description part removed, so the target keeps the answer benefit of descriptions without the description text. A chosen-only consistency loss anchors Stage II predictions to Stage I logits (Eq. 8), with its contribution assessed by component ablations (Sec. 4.6).

To quantify the benefit of preliminary description generation, we propose the InternVQA benchmark. InternVQA covers three image settings, 10 task types, and diverse answer lengths. For each visual question, our benchmark evaluates the same model under two inference modes: direct answering and caption-then-answering. By comparing the scores, InternVQA quantifies the answer-quality gain of the caption-then-answer protocol, and uses this gain to measure description reliance. Our contributions are as follows:

- We propose Internalization, a two-stage training method that reduces description reliance by learning a compact implicit prior, while achieving higher aggregate public-benchmark retention than the compared methods.
- We propose InternVQA, a benchmark that quantifies description reliance in VQA by comparing answer scores under direct answering and caption-then-answering across 3 image settings, 10 task types, and 3 QA lengths.
- On Qwen3-VL and LLaVA-OneVision, Internalization raises the average direct-answer score from 40.21 to 63.11 and cuts the caption-direct gap from 16.62 to 3.98 points, with consistent gains on all four backbones. It retains 96.45% of aggregate performance on ten public benchmarks and shortens end-to-end inference time by 23.86%.

2 PRELIMINARIES

Description Reliance in Vision Language Models. Mainstream VLMs encode images into visual tokens, project them into the LLM space, and generate responses autoregressively with text tokens. The training data of Qwen-VL (Bai et al., 2025) and LLaVA (Li et al., 2024) contain many caption style samples and long form image descriptions (Kanjula et al., 2025; An et al., 2025), so VLMs often learn a caption-then-answer habit in VQA, and because autoregressive decoding (Younes et al., 2024; Hassan et al., 2025) keeps the generated description in the visible context, the model uses it as self-generated textual evidence. This behavior can improve answer accuracy, but it also increases

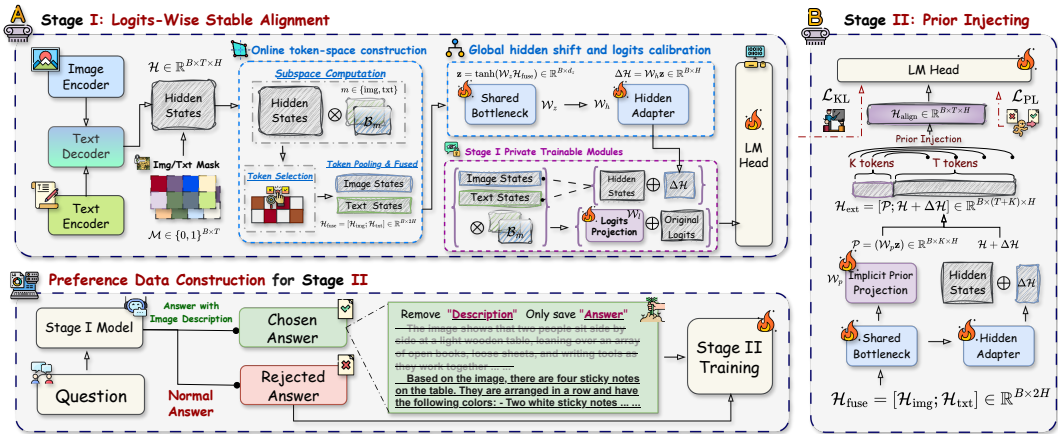


Figure 2: **Overview of `Internalization`.** The stage I/II shared modules are highlighted in **blue**, while the stage-private modules are highlighted in **purple**. **Stage I** builds a compact image question representation from the final decoder hidden states. It selects evidence tokens through online token space construction, fuses image and text states, and learns a logits correction to align the new module with the base model output distribution. **Preference data construction** employs the Stage I model to generate an unconstrained answer as the rejected response and a description-based answer as the chosen response, with the description part removed before training. **Stage II** maps the fused representation into K implicit prior tokens ($K = 16$ by default) and injects them into the original hidden states through gated attention before the LM head. Preference learning encourages concise direct answers, while a chosen only KL loss matches frozen Stage I logits to reduce style drift during preference training. At inference, only the shared modules and the prior tokens remain. The Stage I logits projection and the frozen teacher are training-only.

output tokens and latency. Prior analysis relates this issue to text inertia, where the model over relies on generated text and under uses the image (Liu et al., 2024). **Caption-based** methods further exploit this behavior by generating a short or question-guided caption before answering (Hu et al., 2023; Özdemir & Akagündüz, 2024). They provide useful intermediate evidence but keep explicit text in the inference path, so the cost of description generation remains.

Direct Answering without Explicit Descriptions. Existing methods avoid generated text as intermediate evidence by moving reasoning to other representations. One direction is **Thinking-with-image**, which strengthens the role of visual evidence through attention or logits adjustment (Liu et al., 2024), context pruning, rendered instructions, visual tools, or image operations (Fan et al., 2025; Wu et al., 2025a). They shift evidence construction back to the visual modality but often require inference-time interventions, extra visual transformations, or tool-sensitive designs. Another direction is **Latent reasoning**, which moves intermediate reasoning into hidden or continuous states (Zelikman et al., 2024; Hao et al., 2024). Although these methods remove visible reasoning tokens, latent reasoning still requires additional hidden-state computation and can even be slower than generating text tokens. Appendix H groups the baseline methods in more detail.

Distilling Intermediate Context. Context distillation transfers context-assisted answers into generation without the extra context (Snell et al., 2022). Related approaches internalize explicit CoT (Deng et al., 2023; 2024) or distill System 2 outputs (Yu et al., 2024). We study this objective in VQA through InternVQA, which measures description reliance, and a two-stage architecture that learns an image-question prior at the final decoder layer.

3 METHODS

We split `Internalization` training into two stages for stability (Fig. 2): Stage I learns a logits-wise correction that aligns the new module with the base model, and Stage II turns the aligned representation into a hidden-state prior for direct answering. The method adds four small components: a training-free online PCA token selector, a shared bottleneck $(\mathcal{W}_z, \mathcal{W}_h)$, a Stage I logits head $\mathcal{W}_\ell$, and a Stage II prior-token head $\mathcal{W}_p$ with gated injection. Only the token selector, the bottleneck, and

the prior-token head run at inference. The logits head and the frozen Stage I teacher are training-only. The retained prior module adds no description-generation steps, and its runtime cost is measured in Appendix F.

3.1 Stage I: Logits-Wise Stable Alignment

Motivation: A module that is not pre-trained has a parameter distribution misaligned with the base model, and injecting its outputs directly can act like noise that destabilizes generation. Stage I therefore calibrates the module in the logits space and later reuses the calibrated logits as a lightweight teacher for Stage II. To summarize answer-relevant image and question evidence compactly, we use PCA as an online training-free token selector, since mean pooling may dilute sparse key cues and learnable pooling may overfit the warm-up data.

Stage I employs 280K VQA pairs for instruction tuning (Sec. 4). For efficiency and output stability, we train on the final-layer hidden states of text decoder $\mathcal{H} \in \mathbb{R}^{B\times T\times H}$, where B is the batch size, T is the sequence length, and H is the hidden size. We define two binary masks: $\mathcal{M}_{\text{img}} \in \{0,1\}^{B\times T}$ for image-token positions and $\mathcal{M}_{\text{txt}} \in \{0,1\}^{B\times T}$ for question text positions. Answer tokens, present under teacher forcing, are excluded from both masks in both stages, so the prior depends on the image and question only. In each forward pass, we gather the tokens under each mask to obtain $\mathcal{X}_m \in \mathbb{R}^{B\times N_m\times H}$ for $m \in \{\text{img}, \text{txt}\}$, where $\mathcal{X}_{\text{img}}$ and $\mathcal{X}_{\text{txt}}$ denote the gathered image and question tokens, and N_m counts them.

Online token-space construction: Since the `Implicit Prior` guides answering, it must encode both the image and the question. To keep this prior compact, we select the tokens with the highest projected energy in the image-token space and the text-token space, and aggregate them into a unified prior representation. We first center the tokens: $\bar{\mathcal{X}}_m = \mathcal{X}_m - \mathbf{1}\boldsymbol{\mu}_m^\top$, where $\mathbf{1} \in \mathbb{R}^{N_m}$ is the all-ones vector and $\boldsymbol{\mu}_m = \frac{1}{N_m}\sum_{i=1}^{N_m} \mathcal{X}_m[i,:] \in \mathbb{R}^H$. We compute the covariance matrix $\mathbf{C}_m = \frac{1}{N_m}\bar{\mathcal{X}}_m^\top\bar{\mathcal{X}}_m \in \mathbb{R}^{H\times H}$ and perform eigendecomposition $\mathbf{C}_m\mathcal{B}_{m,j} = \lambda_{m,j}\mathcal{B}_{m,j}$. We take the top r_m eigenvectors by $\lambda_{m,j}$ to form the basis $\mathcal{B}_m = [\mathcal{B}_{m,1}, \ldots, \mathcal{B}_{m,r_m}] \in \mathbb{R}^{H\times r_m}$ with $\mathcal{B}_m^\top\mathcal{B}_m \approx \mathbf{I}$. Here r is the rank budget with $1 \le r \le H$, and we set $r_m = \min(r, N_m, H)$ to avoid unstable directions when N_m is small. Next, we define a projected-energy score for each token $\mathbf{x}_{m,i} = \mathcal{X}_m[i,:]$ as $s_{m,i} = \|(\mathbf{x}_{m,i} - \boldsymbol{\mu}_m)^\top\mathcal{B}_m\|_2$. We select the top-$k_m$ tokens under each mask by $s_{m,i}$ and mean-pool them into $\mathcal{H}_m \in \mathbb{R}^{B\times H}$ for $m \in \{\text{img}, \text{txt}\}$. We then concatenate them into a fused vector $\mathcal{H}_{\text{fuse}} = [\mathcal{H}_{\text{img}}; \mathcal{H}_{\text{txt}}] \in \mathbb{R}^{B\times 2H}$. This procedure is training-free and online: $\mathcal{B}_m$ is computed per sample in each forward pass, adding no trainable parameters or external memory. The hidden states entering the covariance and eigendecomposition are detached and the top-k_m selection is discrete, so no gradient flows through PCA. The loss updates only the downstream module. Because PCA runs on task-conditioned decoder states rather than pixels, a high projected-energy token lies along the dominant directions of the model's own representation of this image and question. We treat this as an effective heuristic rather than a semantic guarantee: the matched controls compare it with bottom-k, random-k, and learnable pooling (Appendices B, C, and G).

Global hidden shift and logits calibration: Stage I and Stage II share a bottleneck that generates an intermediate feature $\mathbf{z}$ and a global hidden shift $\Delta\mathcal{H}$:

$$\mathbf{z} = \tanh(\mathcal{W}_z\mathcal{H}_{\text{fuse}}), \qquad \Delta\mathcal{H} = \mathcal{W}_h\mathbf{z} \in \mathbb{R}^{B\times H}, \tag{1}$$

where $\mathcal{W}_z \in \mathbb{R}^{d_z\times 2H}$ and $\mathcal{W}_h \in \mathbb{R}^{H\times d_z}$ are trainable and d_z is the bottleneck dimension. We apply the shift to all positions, $\mathcal{H}' = \mathcal{H} + \Delta\mathcal{H}$ with $\Delta\mathcal{H}$ broadcast over T, and repeat the pooling on $\mathcal{H}'$ to obtain $(\mathcal{H}'_{\text{img}}, \mathcal{H}'_{\text{txt}})$, which helps the model adapt to hidden-state updates early and reduces the risk of collapse in Stage II. We project the pooled vectors onto the modality bases, $\mathbf{u}_{\text{img}} = \mathcal{H}'_{\text{img}}\mathcal{B}_{\text{img}}$ and $\mathbf{u}_{\text{txt}} = \mathcal{H}'_{\text{txt}}\mathcal{B}_{\text{txt}}$, pad each to a fixed rank, and concatenate them as $\mathbf{u} = [\text{pad}(\mathbf{u}_{\text{img}}); \text{pad}(\mathbf{u}_{\text{txt}})] \in \mathbb{R}^{B\times 2r}$. A linear head produces a vocabulary-level correction:

$$\Delta\boldsymbol{\ell} = \mathcal{W}_\ell\mathbf{u} \in \mathbb{R}^{B\times V}, \quad \boldsymbol{\ell}_T = \mathcal{W}_{\text{lm}}\mathcal{H}_{:,T,:} + \Delta\boldsymbol{\ell}, \tag{2}$$

where V is the vocabulary size, $\mathcal{W}_\ell \in \mathbb{R}^{V\times 2r}$ is trainable, and $\boldsymbol{\ell}_T \in \mathbb{R}^{B\times V}$ denotes the logits at the last position (used to predict the next token). $\Delta\boldsymbol{\ell}$ is one vector per sample, computed from the masked image and question tokens and independent of the answer. Since answers are multi-token, the teacher-forced cross-entropy covers every answer token and the same $\Delta\boldsymbol{\ell}$ is added to the logits

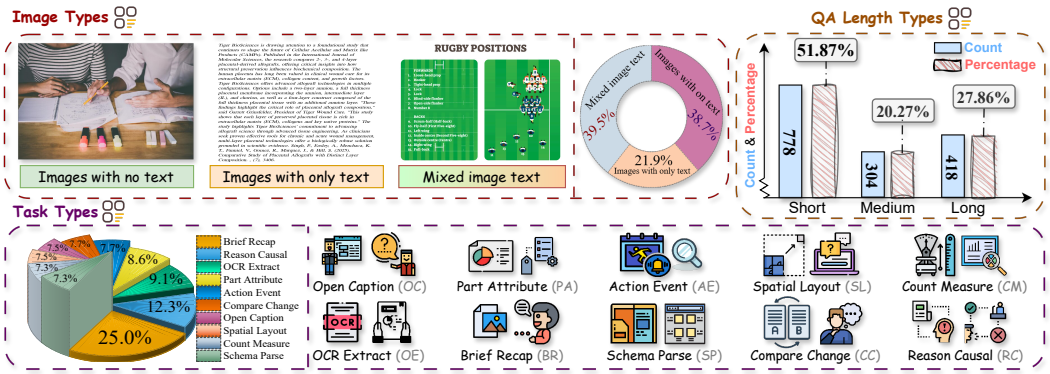


Figure 3: **Composition of `InternVQA`. Image types**: images with no text (38.7%), images with only text (21.9%), and mixed image-text inputs (39.5%). **Task types**: 10 VQA task types (OC, OE, PA, AE, SL, CM, CC, SP, BR, RC) with the largest subset at 25.0% and the rest between 7.3% and 12.3%. **QA length types**: bucketed by the token length of the text completion, short $[1, 256]$, medium $[257, 1024]$, and long > 1024 tokens. Of the 1.5K samples, 778 are short (51.87%), 304 medium (20.27%), and 418 long (27.87%).

of each supervised position, so ℓ_T in Eq. (2) denotes the position predicting the next answer token. The correction acts on logits rather than hidden states, leaving intermediate token representations untouched, and $\mathcal{W}_{\text{lm}}$ stays trainable.

To preserve the base distribution and reduce training cost (Behdin et al., 2025), Stage I only trains the last text decoder layer, the language model head, and the added internalization module, while freezing all other parameters. We employ standard cross-entropy on VQA labels so that $\Delta\ell$ quickly aligns the output distribution with the base model. Appendix B further validates the rank choice, token selection, pooling strategy, and last-layer training design.

Why logits-wise calibration is more stable. Let $\mathcal{L}_{\text{ce}}$ be the token-level cross-entropy, $\mathbf{p}$ the softmax distribution, and $\mathbf{e}$ the one-hot label vector. For the logits-space correction $\Delta\boldsymbol{\ell}(\mathcal{W}_\ell)$ and for a hypothetical hidden-space increment $\Delta\mathbf{h}$ with $\boldsymbol{\ell} = \mathcal{W}_{\text{lm}}(\mathbf{h} + \Delta\mathbf{h})$, the gradients are

$$\frac{\partial \mathcal{L}_{\text{ce}}}{\partial \mathcal{W}_\ell} = (\mathbf{p} - \mathbf{e})^\top \frac{\partial \Delta\boldsymbol{\ell}}{\partial \mathcal{W}_\ell}, \qquad \frac{\partial \mathcal{L}_{\text{ce}}}{\partial \Delta\mathbf{h}} = \mathcal{W}_{\text{lm}}^\top(\mathbf{p} - \mathbf{e}). \tag{3}$$

The logits-space update reaches the new module without passing through $\mathcal{W}_{\text{lm}}$, so its scale is easy to control, whereas the hidden-space update inherits the scale and anisotropy of $\mathcal{W}_{\text{lm}}$ and touches every token position, so early gradients are noisier. Empirically, logits-first training is more stable and performs better (Appendix D, Tab. 7).

3.2 Stage II: Hidden-Space Prior Injection

Motivation: Stage I provides a stable initialization by aligning the new module with the base model in the logits space. Built on this alignment, Stage II writes the description benefit into the hidden space as an `Implicit Prior`, enabling direct answering without generating any description.

Prior token construction: Given the final-layer hidden states $\mathcal{H}$, we reuse the same online subspace pooling in Stage I to obtain $\mathcal{H}_{\text{fuse}}$, then compute $\mathbf{z} = \tanh(\mathcal{W}_z\mathcal{H}_{\text{fuse}})$ and $\Delta\mathcal{H} = \mathcal{W}_h\mathbf{z}$. We generate K prior tokens from $\mathbf{z}$ and apply the global shift to the sequence:

$$\mathcal{P} = \text{reshape}(\mathcal{W}_p\mathbf{z}) \in \mathbb{R}^{B\times K\times H}, \qquad \mathcal{H}' = \mathcal{H} + \Delta\mathcal{H} \in \mathbb{R}^{B\times T\times H}, \tag{4}$$

where $\mathcal{W}_p \in \mathbb{R}^{(KH)\times d_z}$ is trainable and K is a hyper-parameter that controls the number of prior tokens. Fig. 2 draws the pair as the extended sequence $\mathcal{H}_{\text{ext}} = [\mathcal{P}; \mathcal{H}']$, whose K prior tokens are read by the T original positions.

Content-adaptive prior injection: The prior tokens are not decoded. Instead, each original position reads from them: we compute its affinity to the prior tokens and aggregate the prior as

$$\mathcal{A} = \text{softmax}\Big(\frac{\mathcal{H}'\mathcal{P}^\top}{\sqrt{H}}\Big) \in \mathbb{R}^{B\times T\times K}, \quad \tilde{\mathcal{P}} = \mathcal{A}\mathcal{P} \in \mathbb{R}^{B\times T\times H}. \tag{5}$$

We control the injection strength by a position-wise gate $\mathbf{g} = \sigma\big(\mathrm{mean}(\mathcal{H}', \mathrm{dim} = -1)\big) \in \mathbb{R}^{B\times T\times 1}$, where σ is the sigmoid function, and obtain the aligned hidden states $\mathcal{H}_{\mathrm{align}}$. The gate scales the prior contribution using the mean hidden activation at each position, with sigmoid bounding the scale in $(0, 1)$. Removing the gate lowers $S_{\texttt{InternVQA}}$ from 95.51 to 86.26 on Qwen3-VL-8B (Tab. 5). Finally, the logits are computed by the same language head:

$$\mathcal{H}_{\mathrm{align}} = \mathcal{H}' + \mathbf{g} \odot \tilde{\mathcal{P}} \in \mathbb{R}^{B\times T\times H}, \quad \boldsymbol{\ell}_{\mathrm{align}} = \mathcal{W}_{\mathrm{lm}}\mathcal{H}_{\mathrm{align}} \in \mathbb{R}^{B\times T\times V}. \tag{6}$$

This design distributes image-question prior information across positions without an extra generation step, and preference training on $\mathcal{W}_p$ improves prior-conditioned direct answering.

Preference training with Stage I logits. We construct preference pairs $(\mathbf{y}^+, \mathbf{y}^-)$ with the Stage I model. The chosen $\mathbf{y}^+$ is the final-answer segment of a **caption-then-answering** output: the prompt requests a description segment followed by an answer segment, and we keep only the text after this fixed boundary, without rewriting. The rejected $\mathbf{y}^-$ is the output under an **unconstrained answering** prompt (Liu et al., 2025a). This construction transfers a caption-assisted final answer into the direct-answering setting. Since the unconstrained response can include a description, the preference signal combines answer quality and directness. Both responses share the image and question, and the prior is computed from these alone (answer tokens are masked out), so chosen and rejected sequences see identical prior tokens. The objective transfers the answer benefit induced by caption-then-answering into prior-conditioned direct answering, while keeping the prior dependent only on the image and question. We perform preference optimization on π_θ induced by $\boldsymbol{\ell}_{\mathrm{align}}$. To reduce the dependence of sequence scores on completion length, we use the average log-probability over completion tokens. With completion mask $m_t \in \{0, 1\}$ and scale $\beta = 0.1$, we use a length-normalized pairwise logistic preference loss:

$$\bar{\ell}(\mathbf{y} \mid \mathbf{x}) = \frac{\sum_{t=1}^{L} m_t \log \pi_\theta(y_t \mid \mathbf{x}, y_{<t})}{\sum_{t=1}^{L} m_t}, \qquad \mathcal{L}_{\mathrm{PL}} = -\log \sigma\Big(\beta\big(\bar{\ell}(\mathbf{y}^+|\mathbf{x}) - \bar{\ell}(\mathbf{y}^-|\mathbf{x})\big)\Big). \tag{7}$$

The pairwise term uses no reference-policy log-ratio. We regularize it separately with a chosen-only teacher KL loss, reusing the frozen Stage I logits correction to limit distribution drift. We define teacher logits $\boldsymbol{\ell}^{\mathrm{T}}$ and student logits $\boldsymbol{\ell}^{\mathrm{S}}$ as $\boldsymbol{\ell}^{\mathrm{T}} = \mathcal{W}_{\mathrm{lm}}\mathcal{H} + \Delta\boldsymbol{\ell}$ and $\boldsymbol{\ell}^{\mathrm{S}} = \boldsymbol{\ell}_{\mathrm{align}}$. We apply the consistency constraint only on the chosen half of the batch. Let the first B samples be chosen and the next B be rejected. We compute

$$\mathcal{L}_{\mathrm{KL}} = \mathrm{KL}\Big(\mathrm{softmax}(\boldsymbol{\ell}^{\mathrm{T}}_{1:B}/\tau) \,\|\, \mathrm{softmax}(\boldsymbol{\ell}^{\mathrm{S}}_{1:B}/\tau)\Big), \tag{8}$$

where τ is a temperature. This chosen-only design avoids constraining the rejected path, so the consistency term does not weaken the preference signal. Stage II minimizes $\mathcal{L} = \mathcal{L}_{\mathrm{PL}} + \lambda\mathcal{L}_{\mathrm{KL}}$, where λ controls the consistency strength. Stage I serves as a teacher only during training. At inference, the teacher branch and the logits correction $\Delta\boldsymbol{\ell}$ are dropped: the model only uses the prior tokens and one forward pass, without generating any description.

4 Experiments and Analysis

4.1 Experimental Setup

To evaluate our method, we employ two mainstream VLMs, Qwen3-VL-2B/8B and LLaVA-OneVision-0.5B/7B. For baselines, we include representative caption-based, Thinking-with-Image, and latent reasoning methods discussed in Sec. 2. Unless stated otherwise, we use $K = 16$ prior tokens, top-$k_m = 16$ selected tokens per modality, subspace rank $r = 256$, and bottleneck $d_z = 256$. Sec. 4.5 and Appendix B report the sensitivity to each. Evaluation uses our `InternVQA` benchmark, defined below. All experiments run on 64 NVIDIA A800 GPUs, each repeated three times with the mean reported (details in Appendix H).

4.2 Dataset Preparation

Training data construction. As shown in Fig. 2, we collect 280K public images from Internet image sources and use GPT-5 to generate one question for each image. **For Stage I**, we randomly

Table 1: Relative diagnostic $S_{\texttt{InternVQA}}$ breakdown on four backbones, grouped by **Length-Type**, **Image-Type**, and **Task-Type**. Blue shading marks the best result and green shading the second best.

Method	Length-Type			Image-Type			Task-Type										Overall↑
	Short↑	Medium↑	Long↑	Img↑	Txt↑	Img+Txt↑	OC↑	PA↑	AE↑	SL↑	CM↑	OE↑	BR↑	SP↑	CC↑	RC↑	
LLaVA-OneVision-0.5B																	
BaseModel	16.71	47.09	56.38	29.50	16.93	24.20	61.88	22.37	12.42	9.22	34.96	21.16	37.27	-30.08	15.28	56.23	25.66
PromptCap	55.20	55.71	54.70	54.88	55.52	55.24	55.63	55.53	54.36	54.53	55.30	55.16	55.44	55.34	55.19	54.63	55.14
QD-Cap	65.17	65.01	64.17	65.57	64.39	64.44	64.72	64.31	64.99	64.71	64.95	64.72	65.27	65.38	63.99	64.85	64.88
PAI	85.87	86.39	86.39	86.17	86.02	86.12	86.49	86.16	85.82	85.53	85.85	86.04	86.45	86.07	85.72	86.20	86.12
OPERA	84.06	84.40	84.16	83.66	84.56	84.42	84.35	83.84	83.83	83.83	83.95	84.28	84.32	84.26	83.91	84.44	84.15
NaPO	85.66	86.08	85.80	85.83	85.62	85.84	86.31	85.47	85.42	85.86	85.29	85.86	86.12	85.69	85.47	85.64	85.78
Visual-RFT	85.72	86.50	85.88	85.95	85.61	86.08	86.23	86.41	85.88	86.34	85.72	86.09	85.62	85.74	86.23	85.75	85.92
Fast Quiet-STaR	90.88	85.18	82.50	87.95	86.30	87.55	87.30	91.19	87.42	90.32	87.34	89.43	89.77	87.21	82.31	75.16	86.73
COCONUT	88.19	85.94	89.47	87.23	88.67	89.05	89.09	87.70	90.48	91.04	86.57	82.28	87.62	90.55	89.59	77.75	87.39
VTool-R1	89.18	85.55	90.07	86.08	86.91	90.07	85.49	90.44	90.56	87.38	90.97	91.12	81.96	91.20	90.88	85.80	88.09
GRIT	89.16	86.65	90.75	90.52	86.00	89.40	90.10	90.51	91.21	90.37	89.79	85.91	85.96	90.48	89.78	89.25	88.89
Ours	**91.18**	**90.89**	**91.51**	**90.94**	**91.25**	**91.47**	**91.64**	**91.43**	**91.30**	**91.23**	**91.30**	**91.33**	**91.16**	**91.33**	**91.23**	**90.66**	**91.20**
LLaVA-OneVision-7B																	
BaseModel	40.17	62.05	49.48	39.03	48.30	49.84	61.42	35.89	34.20	0.13	37.30	56.34	45.88	57.80	50.12	60.39	45.03
PromptCap	78.33	78.45	78.33	78.72	78.54	77.89	78.23	78.09	78.33	78.74	78.64	78.15	78.50	78.55	77.95	78.21	78.34
QD-Cap	83.31	83.34	83.43	83.07	83.61	83.47	83.65	83.22	82.94	83.62	83.11	83.17	83.32	83.36	83.24	83.67	83.34
PAI	91.47	91.65	91.75	91.52	91.56	91.66	91.48	91.36	91.34	91.26	92.01	91.55	91.45	91.90	91.73	91.92	91.58
OPERA	91.04	91.24	91.45	91.30	90.92	91.25	91.46	90.92	91.08	91.13	91.26	91.52	91.13	91.20	91.27	91.20	91.20
NaPO	84.65	85.52	84.61	85.08	84.01	85.04	85.08	85.23	85.04	86.85	82.82	84.14	85.38	83.79	85.15	83.89	84.82
Visual-RFT	91.80	92.17	91.70	92.05	91.79	91.69	91.93	92.08	92.31	91.96	92.09	91.61	91.53	91.81	91.61	92.11	91.84
Fast Quiet-STaR	93.83	89.68	89.67	93.04	88.67	90.15	90.49	93.57	91.27	93.62	88.84	93.19	92.22	88.20	86.67	82.54	91.07
COCONUT	91.41	89.53	92.92	91.66	92.59	92.24	91.20	92.81	92.95	92.88	92.66	89.27	91.59	93.29	92.21	85.02	91.27
VTool-R1	93.12	87.71	92.54	90.98	91.14	93.18	88.46	92.57	92.86	92.36	93.37	93.87	87.88	93.93	93.54	90.91	92.06
GRIT	91.73	90.42	93.61	93.76	91.62	92.44	92.86	92.92	93.61	92.95	92.35	90.37	88.65	91.25	92.90	93.28	91.17
Ours	**93.94**	**93.71**	**93.76**	**93.86**	**93.84**	**93.84**	**93.69**	**93.94**	**93.75**	**93.83**	**93.59**	**94.01**	**93.77**	**94.07**	**93.68**	**94.10**	**93.85**
Qwen3-VL-2B-Instruct																	
BaseModel	56.65	68.24	70.67	44.11	71.56	71.37	64.07	35.63	42.50	40.40	27.21	84.55	70.84	75.99	59.91	68.05	61.80
PromptCap	80.18	80.46	80.40	79.89	80.48	80.61	79.98	80.06	80.05	80.14	80.03	80.07	80.19	80.89	80.59	80.89	80.28
QD-Cap	88.97	88.57	88.72	88.68	88.80	88.94	88.79	89.09	88.85	88.35	88.72	88.19	88.92	89.04	88.89	89.01	88.82
PAI	88.32	88.05	88.61	88.16	88.43	88.49	88.23	88.39	88.74	88.29	88.82	88.35	88.34	88.55	88.09	87.96	88.35
OPERA	90.71	91.29	90.75	90.37	91.20	91.11	91.31	90.44	90.59	90.24	90.86	90.63	90.98	91.08	90.64	91.16	90.84
NaPO	91.59	91.60	91.51	91.75	91.65	91.37	91.75	91.33	91.61	91.72	91.65	91.31	91.36	91.95	91.77	91.73	91.58
Visual-RFT	91.68	91.67	91.86	91.49	92.14	91.74	92.03	91.40	91.29	91.78	92.16	91.44	91.82	91.91	92.00	91.52	91.74
Fast Quiet-STaR	93.08	90.64	86.85	92.60	88.95	91.69	92.34	94.37	91.18	96.70	89.51	93.59	92.55	92.19	85.52	81.59	90.75
COCONUT	92.00	89.96	90.51	90.48	91.44	92.36	100.18	91.08	96.23	92.30	88.40	88.74	91.88	94.83	93.96	84.86	90.79
VTool-R1	92.20	90.90	91.58	88.53	93.61	94.81	88.53	91.43	93.76	91.68	95.49	94.65	88.50	96.34	91.43	89.57	91.71
GRIT	91.00	92.35	90.64	94.19	91.09	90.80	89.91	95.03	92.72	94.40	90.47	89.28	89.67	91.30	91.47	95.20	90.87
Ours	90.21	**93.81**	**100.03**	89.89	**93.81**	**96.25**	100.15	86.89	86.68	88.57	82.83	**96.49**	**95.59**	96.13	**95.22**	**96.62**	**93.35**
Qwen3-VL-8B-Instruct																	
BaseModel	69.71	77.33	79.73	61.57	83.79	78.75	76.27	49.97	67.66	54.82	41.95	91.52	78.60	91.37	71.29	80.34	73.95
PromptCap	80.53	80.52	80.66	80.46	81.11	80.39	80.65	80.59	81.05	80.61	80.52	80.59	80.27	80.87	80.66	80.53	80.55
QD-Cap	84.56	84.52	84.77	84.60	84.79	84.55	84.85	84.49	84.70	84.65	84.56	84.69	84.63	84.43	84.37	84.70	84.62
PAI	89.67	89.93	89.92	89.51	89.69	90.12	89.53	89.56	89.75	89.77	89.45	89.85	89.80	90.14	89.93	90.00	89.79
OPERA	79.96	79.92	80.10	80.06	80.17	79.83	79.82	80.26	79.88	80.29	79.52	79.79	80.08	80.40	79.97	79.85	79.99
NaPO	88.45	87.32	87.88	87.65	88.93	88.02	88.78	87.83	87.85	89.70	87.40	90.06	88.36	87.83	87.81	85.36	88.07
Visual-RFT	94.67	94.73	94.86	94.35	94.95	95.00	95.00	94.53	94.60	94.45	94.77	94.68	95.00	94.49	95.00	94.46	94.73
Fast Quiet-STaR	98.22	95.13	88.25	94.81	91.66	94.49	93.40	98.73	91.55	95.72	91.34	97.73	98.21	92.99	92.84	85.20	93.11
COCONUT	92.22	95.87	96.71	93.71	94.71	94.29	93.94	95.17	97.53	96.46	92.56	90.93	96.94	93.62	94.80	90.23	93.20
VTool-R1	93.77	93.03	95.22	90.62	96.61	96.34	92.64	91.00	98.17	91.91	95.68	97.05	92.14	101.36	95.68	89.98	94.03
GRIT	92.98	95.05	96.03	95.54	93.04	93.23	94.89	94.51	93.77	97.11	90.54	91.10	93.52	94.96	92.53	97.16	93.07
Ours	92.45	**97.41**	**99.92**	90.28	**99.18**	**98.14**	**100.33**	86.88	92.06	92.54	79.80	**98.03**	97.63	**101.64**	**97.70**	**98.17**	**95.51**

split the 280K image question pairs into two halves: one half uses base model answers to keep the supervision close to the original output style, and the other half uses GPT-5 answers to improve answer diversity and quality. **For Stage II**, we reuse the same questions and let the Stage I model generate the rejected and chosen responses as described in Sec. 3.2, so preference training encourages the prior tokens to guide the model toward correct direct answers without the description. **To avoid incorrect preference labels**, we use GPT-5 to judge whether the chosen final answer is logically correct and better aligned with the question than the unconstrained answer. This consistency filtering removes 3.9% noisy pairs before Stage II training. Details are in Appendices A and B.5.

Benchmark and metrics: Fig. 3 summarizes `InternVQA`. GPT-5-mini scores direct and caption-then-answer outputs in $[0, 1]$ using the same image-conditioned rubric. A single-blind study with 30 volunteers ranks the methods consistently with the judge (Spearman $\rho = 0.944$, Appendix E.2). Let D and A denote the mean Direct and Cap scores. Our primary quantities are $100D$, $100A$, and the signed gap $100(A - D)$. For a bounded summary of direct-answer sufficiency, we additionally define $N_{\texttt{InternVQA}} = 100D/\max(D, A) \in [0, 100]$, with $N = 0$ when $D = A = 0$. Higher is better: $N = 100$ means that direct answering matches or exceeds caption-then-answering. We report N alongside Direct and Cap so that the summary is interpreted together with answer quality (Tab. 2). The relative diagnostic $S_{\texttt{InternVQA}} = 100(1 - (A - D)/D)$ for $D > 0$ is retained in the detailed subset and ablation results; its relation to N is given in Appendix J.1. A uniform GPT-based format checker is applied across all methods. Benchmark composition, evaluation prompts, and the human study are documented in Appendices I, M, and E. The training data and the benchmark do not overlap.

4.3 Benchmark Evaluations

InternVQA performance evaluations: Tab. 2 reports the primary quantities on four backbones and Tab. 1 the $S_{\texttt{InternVQA}}$ summary by subset. The base models show clear description reliance: captioning adds 16.62 judge points over direct answering on average. `Internalization` raises the average direct score from 40.21 to 63.11 and shrinks the gap to 3.98 points, consistently across model families and scales. The gap closes from the direct side: the caption score also rises from 56.83 to 67.09, so the trained models answer directly about as well as the base models answer after a caption (better on both LLaVA-OneVision backbones, within 0.3 and 4.2 points on Qwen3-

VL-2B/8B). Among the methods with reported absolute scores, `Internalization` has the best three-run mean Direct, Gap, and $S_{\texttt{InternVQA}}$ on every backbone, so the top of the ranking does not depend on the normalization (Tab. 25). We also report the description-trigger rate, the fraction of outputs under the unconstrained prompt that contain an explicit image-description segment before the final answer, detected with the GPT-based format check of Appendix M. Averaged over the four backbones, it drops from 76.19% to 4.26%, indicating a clear shift from description-first generation to direct answering (more results in Appendix J and B).

Table 2: **Primary InternVQA quantities**: Direct, Cap, and signed Gap ($\times 100$, three-run means), with bounded $N_{\texttt{InternVQA}}$ computed from the displayed Direct and Cap means. Average is the mean of the four backbone entries. Direct improves while Cap is preserved. Baseline comparisons appear in Tab. 25.

Method	LLaVA-OneVision-0.5B				LLaVA-OneVision-7B				Qwen3-VL-2B				Qwen3-VL-8B				Average			
	Direct↑	Cap	Gap↓	N ↑	Direct↑	Cap	Gap↓	N ↑	Direct↑	Cap	Gap↓	N ↑	Direct↑	Cap	Gap↓	N ↑	Direct↑	Cap	Gap↓	N ↑
Base model	19.45	33.91	14.46	57.36	33.29	51.59	18.30	64.53	45.65	63.09	17.44	72.36	62.45	78.72	16.27	79.33	40.21	56.83	16.62	68.39
Ours	**50.14**	**54.55**	**4.41**	**91.92**	**64.93**	**68.92**	**3.99**	**94.21**	**62.82**	**67.00**	**4.18**	**93.76**	**74.53**	77.88	**3.35**	**95.70**	**63.11**	**67.09**	**3.98**	**93.90**

Table 3: **Average public benchmark performance and retention** over LLaVA-OneVision-0.5B/7B and Qwen3-VL-2B/8B. **RetainScore** sums the ten benchmark scores and **RetainPct** is relative to the corresponding base model (Appendix E.1).

Method	AI2D↑	ChartQA↑	GQA↑	HB↑	MMBench↑	MMStar↑	OCRVQA↑	POPE↑	RWQA↑	SQA↑	RetainScore↑	RetainPct↑
PromptCap	66.88	66.69	**58.50**	34.62	71.36	51.98	53.93	77.84	58.99	80.32	621.12	91.55%
QD-Cap	67.68	67.73	53.76	34.44	67.88	50.41	52.47	79.81	57.81	78.02	610.00	89.92%
PAI	65.79	66.81	52.36	33.21	66.86	49.68	51.85	81.63	58.11	76.43	602.75	88.85%
OPERA	71.03	69.07	56.12	35.43	71.47	52.64	55.88	84.23	59.08	82.98	637.91	94.03%
NaPO	71.14	69.87	55.58	35.19	71.33	**53.87**	53.50	81.30	**62.59**	78.88	633.26	93.34%
Visual-RFT	70.75	**70.08**	55.77	35.77	70.94	52.63	55.74	83.49	60.14	81.38	636.69	93.85%
Fast Quiet-STaR	69.76	63.36	53.97	33.16	69.28	52.13	55.16	81.41	58.81	81.77	618.80	91.21%
COCONUT	66.62	65.27	54.38	33.34	67.81	52.81	54.73	76.61	61.03	81.02	613.63	90.45%
VTool-R1	71.59	66.97	57.50	34.55	71.70	52.31	52.54	85.17	61.81	82.69	636.84	93.87%
GRIT	69.31	64.32	57.80	35.79	70.92	52.86	55.78	84.97	60.41	80.80	632.97	93.30%
Ours	**73.68**	67.95	58.35	**40.98**	**72.56**	53.06	**56.37**	**85.88**	62.33	**83.18**	**654.33**	**96.45%**

We further assess general VQA ability on ten public benchmarks using their task-specific scoring protocols (Appendix B.1), complementing the within-taxonomy InternVQA evaluation.

Among the compared post-training and inference methods, `Internalization` achieves the highest average RetainScore of 654.33 and RetainPct of 96.45% across four backbones, ranking first on six benchmarks and second on three (Tab. 3). This is aggregate retention: task-level changes vary, with a largest decline of 20.48 points on Qwen3-VL-2B ChartQA ($77.40 \rightarrow 56.92$, Appendix E.1). Across Tabs. 1, 3, and 4, `Internalization` gives the best joint trade-off among direct-answer behavior, retention, and inference cost, although individual benchmarks and InternVQA subsets are led by different methods.

Matched controls and evaluation consistency: The heterogeneous baselines establish competitiveness, while matched controls assess the added value of our training design under shared answer supervision. A vanilla SFT baseline uses the same chosen answers with matched data size, backbones, and optimization settings, directly comparing our training design with answer-only supervision: it reaches an $S_{\texttt{InternVQA}}$ of 86.72, a RetainPct of 91.84%, and a trigger rate of 10.63%, against 93.48, 96.45%, and 4.26% for `Internalization`. A matched Stage II ablation without prior injection retains the same preference pairs, optimization budget, losses, and Stage I initialization, but reaches only an average $S_{\texttt{InternVQA}}$ of 81.52 and RetainPct of 91.87% over the four backbones, compared with 93.48 and 96.45% for the full method (Qwen3-VL-8B in Tab. 5, all backbones in Appendix C). This ablation supports the contribution of prior injection within the two-stage training pipeline. Since GPT-5-mini scores InternVQA and GPT-5 labels half of the Stage I answers, we also re-score the judge-dependent part of InternVQA with two judges from other model families: `Internalization` obtains an average $S_{\texttt{InternVQA}}$ of 92.76 and 91.88, against 89.41 and 88.95 for the strongest baseline, reproducing the leading method across judge families. The human study independently hides method names and randomizes answer order, providing a complementary check under a human rubric (Appendix E.2).

Supervision and prior design: The label-source ablation supports mixed supervision: with 280K Stage I pairs, 140K GPT-5 plus 140K base-model labels outperform 280K GPT-5 labels on both $S_{\texttt{InternVQA}}$ and RetainPct across all four backbones (Appendix B.5). Thus, more GPT-5 labels alone do not reproduce the result. Together with SFT and prior-injection comparisons, this supports our training design. InternVQA measures functional transfer within the shared task taxonomy, complemented by public-benchmark retention.

4.4 Inference Efficiency

Table 4: Normalized inference time (%, base model = 100%, lower is better).

Method	Short(%) ↓	Medium(%) ↓	Long(%) ↓
Base Model	100.00	100.00	100.00
PromptCap	142.80	118.45	106.72
QD-Cap	136.25	114.36	104.88
PAI	96.84	94.72	92.65
OPERA	108.36	112.74	119.58
NaPO	87.92	84.65	82.31
Visual-RFT	86.45	83.92	81.74
Fast Quiet-STaR	93.18	89.64	86.37
COCONUT	95.72	91.48	87.95
VTool-R1	178.62	151.34	132.48
GRIT	154.27	133.86	117.52
Ours	**71.23**	**76.82**	**80.36**

We report the average normalized inference cost on the four backbones in Tab. 4, testing each method under the same backbone, prompt, and decoding configuration. Values are ratios of end-to-end wall-clock time to that of the direct-answering base model (100.00%), and lower is better.

Caption-based methods add description generation, and tool-based and latent reasoning methods add visual operations or hidden computation, whereas `Internalization` only adds compact prior tokens before the language model head. It achieves the lowest cost across short, medium, and long answers, 71.23%, 76.82%, and 80.36% of the base model, an average reduction of 23.86%. The saving comes from generating fewer tokens, not from a faster forward pass: at a fixed `max_new_tokens`, the base model and `Internalization` decode at 33.79 and 34.30 tokens/s (Appendix F).

4.5 Hyperparameter Sensitivity

Number of prior tokens: Tab. 9 studies the prior token count K. On Qwen3-VL-8B, as K increases from 1 to 64, the direct score rises from 63.62 to 75.02 and $S_{\texttt{InternVQA}}$ from 92.38 to 96.09. The gain saturates quickly: $K = 256$ slightly reduces $S_{\texttt{InternVQA}}$ (93.66), and RetainPct stays between 94.27% and 96.05% for all settings. $K = 16$ ($S_{\texttt{InternVQA}}$ 95.51, RetainPct 95.60%) gives the most stable trade-off across the four backbones, so we use it by default.

Subspace decomposition: Fig. 7 compares PCA, SVD, ICA, and QR in the complete Stage I pipeline. Centered PCA and SVD give identical token scores for the same subspace, but the logits head also consumes basis-dependent coordinates. These end-to-end results do not isolate token selection. We use PCA by default; its measured overhead is 1.9% on 1K long-context samples averaging 11.3K tokens (Appendix G).

4.6 Ablation Studies

Tab. 5 reports an ablation on Qwen3-VL-8B, with other backbones in Appendix C. Both stages are necessary. Removing Stage I lowers RetainScore and $S_{\texttt{InternVQA}}$, supporting its role in stabilizing the prior. Repeated pooling and z contribute more than token selection in this comparison. Removing Stage II causes a substantial $S_{\texttt{InternVQA}}$ drop, supporting its role in direct answering, with gating and prior injection being critical and $\mathcal{L}_{\text{PL}}/\mathcal{L}_{\text{KL}}$ improving stability. Removing shared modules hurts both metrics the most, highlighting shared priors as key to generalization.

Table 5: Ablation studies on Qwen3-VL-8B.

Method	RetainScore↑	$\mathbf{S}_{\texttt{InternVQA}}$ ↑
Ours	722.04	95.51
w/o Stage I	671.94 ▼50.10	86.34 ▼9.17
↪ w/o repeated pooling	694.08 ▼27.96	90.48 ▼5.03
↪ w/o z	693.07 ▼28.97	89.31 ▼6.20
↪ w/o token selecting	703.26 ▼18.78	91.84 ▼3.67
w/o Stage II	676.34 ▼45.70	73.39 ▼22.12
↪ w/o gating	696.86 ▼25.18	86.26 ▼9.25
↪ w/o prior injecting	688.21 ▼33.83	88.31 ▼7.20
↪ w/o $\mathcal{L}_{\text{PL}}$	695.59 ▼26.45	85.57 ▼9.94
↪ w/o $\mathcal{L}_{\text{KL}}$	683.29 ▼38.75	90.87 ▼4.64
w/o shared modules	661.57 ▼60.47	69.97 ▼25.54

5 Conclusion

We mitigate VLMs' reliance on image descriptions for VQA with `Internalization`, a two-stage training method that learns a compact image-question prior from caption-derived answer supervision: Stage I calibrates logits for stable alignment, and Stage II learns prior tokens via preference training anchored by a Stage I logits teacher. On the proposed `InternVQA` benchmark, it raises the average direct score of four base models from 40.21 to 63.11 and cuts the caption-direct gap from 16.62 to 3.98 points, achieves 96.45% aggregate public-benchmark retention, and shortens inference time by 23.86%. Label-source and prior-injection ablations support the training design, cross-family judges reproduce the leading method, and the human study agrees with the overall automatic ranking ($\rho = 0.944$).

ETHICS STATEMENT

Our goal is to make vision-language models answer more directly by internalizing the utility of intermediate descriptions into a compact implicit prior, reducing unnecessary generation and computation. The approach does not add new sensing capabilities, does not rely on external tools or private data, and operates within the base model's input-output interface. Potential risks mainly relate to broader accessibility of VLM deployment and possible changes in response style after post-training. We therefore report RetainPct on diverse public benchmarks and use logits-wise alignment plus consistency regularization to mitigate distribution shift and style drift. Our human evaluation (Appendix E.2) involves 30 volunteers who rate anonymized model outputs. All volunteers were informed of the purpose of the study and agreed to participate.

REPRODUCIBILITY STATEMENT

Sec. 3 specifies both training stages, and Appendix A gives pseudocode for the full pipeline (Alg. 1, Alg. 2, and Alg. 3). Sec. 4 and Appendix H describe the experimental settings, and Appendix B.5 details the training data construction. Appendix I describes the `InternVQA` composition and evaluation protocol, and Appendix M lists the prompts for Stage I data construction and benchmark evaluation. Our code and `InternVQA` examples are provided in the supplementary material.

AI USE STATEMENT

We used generative AI tools as part of the research methodology: GPT-5 generates the training questions and half of the Stage I answers and filters the Stage II preference pairs, and GPT-5-mini scores responses for `InternVQA` (Sec. 4). Our human evaluation ranks the methods consistently with the automatic judge (Spearman $\rho = 0.944$) under a combined correctness, faithfulness, and directness rubric (Appendix E.2). We also used an AI assistant for LaTeX formatting and language editing of the manuscript. All authors reviewed the AI-assisted content and take full responsibility for this paper, including its text, claims, and results.

REFERENCES

Xiang An, Yin Xie, Kaicheng Yang, Wenkang Zhang, Xiuwei Zhao, Zheng Cheng, Yirui Wang, Songcen Xu, Changrui Chen, Didi Zhu, Chunsheng Wu, Huajie Tan, Chunyuan Li, Jing Yang, Jie Yu, Xiyao Wang, Bin Qin, Yumeng Wang, Zizhen Yan, Ziyong Feng, Ziwei Liu, Bo Li, and Jiankang Deng. Llava-onevision-1.5: Fully open framework for democratized multimodal training, 2025. URL https://arxiv.org/abs/2509.23661.

Daniel Aarao Reis Arturi, Eric Zhang, Andrew Ansah, Kevin Zhu, Ashwinee Panda, and Aishwarya Balwani. Shared parameter subspaces and cross-task linearity in emergently misaligned behavior, 2025. URL https://arxiv.org/abs/2511.02022.

Shuai Bai, Yuxuan Cai, Ruizhe Chen, Keqin Chen, Xionghui Chen, Zesen Cheng, Lianghao Deng, Wei Ding, Chang Gao, Chunjiang Ge, Wenbin Ge, Zhifang Guo, Qidong Huang, Jie Huang, Fei Huang, Binyuan Hui, et al. Qwen3-vl technical report, 2025. URL https://arxiv.org/abs/2511.21631.

Kayhan Behdin, Yun Dai, Ata Fatahibaarzi, Aman Gupta, Qingquan Song, Shao Tang, Hejian Sang, Gregory Dexter, Sirou Zhu, Siyu Zhu, et al. Efficient ai in practice: Training and deployment of efficient llms for industry applications. *arXiv preprint arXiv:2502.14305*, 2025.

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*, 2022.

Yuntian Deng, Kiran Prasad, Roland Fernandez, Paul Smolensky, Vishrav Chaudhary, and Stuart Shieber. Implicit chain of thought reasoning via knowledge distillation. *arXiv preprint arXiv:2311.01460*, 2023. URL https://arxiv.org/abs/2311.01460.

Yuntian Deng, Yejin Choi, and Stuart Shieber. From explicit CoT to implicit CoT: Learning to internalize CoT step by step. *arXiv preprint arXiv:2405.14838*, 2024. URL https://arxiv.org/abs/2405.14838.

Yue Fan, Xuehai He, Diji Yang, Kaizhi Zheng, Ching-Chen Kuo, Yuting Zheng, Sravana Jyothi Narayanaraju, Xinze Guan, and Xin Eric Wang. Grit: Teaching mllms to think with images. *arXiv preprint arXiv:2505.15879*, 2025.

Shibo Hao, Sainbayar Sukhbaatar, DiJia Su, Xian Li, Zhiting Hu, Jason Weston, and Yuandong Tian. Training large language models to reason in a continuous latent space. *arXiv preprint arXiv:2412.06769*, 2024.

Conor Hassan, Nasrulloh Loka, Cen-You Li, Daolang Huang, Paul E. Chang, Yang Yang, Francesco Silvestrin, Samuel Kaski, and Luigi Acerbi. Efficient autoregressive inference for transformer probabilistic models, 2025. URL https://arxiv.org/abs/2510.09477.

Yushi Hu, Hang Hua, Zhengyuan Yang, Weijia Shi, Noah A Smith, and Jiebo Luo. Promptcap: Prompt-guided image captioning for vqa with gpt-3. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 2963–2975, 2023.

Jenny Y. Huang, Mehul Damani, Yousef El-Kurdi, Ramon Astudillo, and Wei Sun. Latency and token-aware test-time compute, 2025a. URL https://arxiv.org/abs/2509.09864.

Qidong Huang, Xiaoyi Dong, Pan Zhang, Bin Wang, Conghui He, Jiaqi Wang, Dahua Lin, Weiming Zhang, and Nenghai Yu. Opera: Alleviating hallucination in multi-modal large language models via over-trust penalty and retrospection-allocation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 13418–13427, 2024.

Wei Huang, Yizhe Xiong, Xin Ye, Zhijie Deng, Hui Chen, Zijia Lin, and Guiguang Ding. Fast quiet-star: Thinking without thought tokens. *CoRR, abs/2505.17746*, 2025b.

Karthik Reddy Kanjula, Surya Guthikonda, Nahid Alam, and Shayekh Bin Islam. Understanding and mitigating toxicity in image-text pretraining datasets: A case study on llava, 2025. URL https://arxiv.org/abs/2505.06356.

Pranjal Kumar. Large language models (llms): survey, technical frameworks, and future challenges. *Artificial Intelligence Review*, 57(10):260, 2024.

Bo Li, Yuanhan Zhang, Dong Guo, Renrui Zhang, Feng Li, Hao Zhang, Kaichen Zhang, Peiyuan Zhang, Yanwei Li, Ziwei Liu, et al. Llava-onevision: Easy visual task transfer. *arXiv preprint arXiv:2408.03326*, 2024.

Hua Li, Tianyang Wang, Feibin Zhang, and Fulei Chu. Svd theory for machine fault detection: A review. *IEEE Sensors Journal*, 2025a.

Zongxia Li, Xiyang Wu, Hongyang Du, Fuxiao Liu, Huy Nghiem, and Guangyao Shi. A survey of state of the art large vision language models: Benchmark evaluations and challenges. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*, pp. 1587–1606, June 2025b.

Shi Liu, Kecheng Zheng, and Wei Chen. Paying more attention to image: A training-free method for alleviating hallucination in lvlms. In *European Conference on Computer Vision*, pp. 125–140. Springer, 2024.

Shunyu Liu, Wenkai Fang, Zetian Hu, Junjie Zhang, Yang Zhou, Kongcheng Zhang, Rongcheng Tu, Ting-En Lin, Fei Huang, Mingli Song, et al. A survey of direct preference optimization. *arXiv preprint arXiv:2503.11701*, 2025a.

Yi Liu, Dianqing Liu, Mingye Zhu, Junbo Guo, Yongdong Zhang, and Zhendong Mao. Leveraging importance sampling to detach alignment modules from large language models, 2025b. URL https://arxiv.org/abs/2505.19700.

Ziyu Liu, Zeyi Sun, Yuhang Zang, Xiaoyi Dong, Yuhang Cao, Haodong Duan, Dahua Lin, and Jiaqi Wang. Visual-rft: Visual reinforcement fine-tuning. *arXiv preprint arXiv:2503.01785*, 2025c.

Humza Naveed, Asad Ullah Khan, Shi Qiu, Muhammad Saqib, Saeed Anwar, Muhammad Usman, Naveed Akhtar, Nick Barnes, and Ajmal Mian. A comprehensive overview of large language models. *ACM Transactions on Intelligent Systems and Technology*, 16(5):1–72, 2025.

Shiwen Ni, Guhong Chen, Shuaimin Li, Xuanang Chen, Siyi Li, Bingli Wang, Qiyao Wang, Xingjian Wang, Yifan Zhang, Liyang Fan, Chengming Li, Ruifeng Xu, Le Sun, and Min Yang. A survey on large language model benchmarks, 2025. URL https://arxiv.org/abs/2508.15361.

Övgü Özdemir and Erdem Akagündüz. Enhancing visual question answering through question-driven image captions as prompts. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 1562–1571, 2024.

Ruizhe Shi, Minhak Song, Runlong Zhou, Zihan Zhang, Maryam Fazel, and Simon S Du. Understanding the performance gap in preference learning: A dichotomy of rlhf and dpo. *arXiv preprint arXiv:2505.19770*, 2025.

Gaurav Shinde, Anuradha Ravi, Emon Dey, Shadman Sakib, Milind Rampure, and Nirmalya Roy. A survey on efficient vision-language models. *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, 15(3):e70036, 2025.

Charlie Snell, Dan Klein, and Ruiqi Zhong. Learning by distilling context. *arXiv preprint arXiv:2209.15189*, 2022. URL https://arxiv.org/abs/2209.15189.

Kun Wang, Guibin Zhang, Zhenhong Zhou, Jiahao Wu, Miao Yu, Shiqian Zhao, Chenlong Yin, Jinhu Fu, Yibo Yan, Hanjun Luo, et al. A comprehensive survey in llm (-agent) full stack safety: Data, training and deployment. *arXiv preprint arXiv:2504.15585*, 2025a.

Xiasi Wang, Tianliang Yao, Simin Chen, Runqi Wang, Lei Ye, Kuofeng Gao, Yi Huang, and Yuan Yao. Vlminferslow: Evaluating the efficiency robustness of large vision-language models as a service. *arXiv preprint arXiv:2506.15755*, 2025b.

Yaoting Wang, Shengqiong Wu, Yuecheng Zhang, Shuicheng Yan, Ziwei Liu, Jiebo Luo, and Hao Fei. Multimodal chain-of-thought reasoning: A comprehensive survey. *arXiv preprint arXiv:2503.12605*, 2025c.

Zichong Wang, Zhibo Chu, Thang Viet Doan, Shiwen Ni, Min Yang, and Wenbin Zhang. History, development, and principles of large language models: an introductory survey. *AI and Ethics*, 5(3): 1955–1971, 2025d.

Mingyuan Wu, Jingcheng Yang, Jize Jiang, Meitang Li, Kaizhuo Yan, Hanchao Yu, Minjia Zhang, Chengxiang Zhai, and Klara Nahrstedt. Vtool-r1: Vlms learn to think with images via reinforcement learning on multimodal tool use. *arXiv preprint arXiv:2505.19255*, 2025a.

Taiqiang Wu, Chaofan Tao, Jiahao Wang, Runming Yang, Zhe Zhao, and Ngai Wong. Rethinking kullback-leibler divergence in knowledge distillation for large language models. In *Proceedings of the 31st International Conference on Computational Linguistics*, pp. 5737–5755, 2025b.

Wenyi Xiao, Zechuan Wang, Leilei Gan, Shuai Zhao, Zongrui Li, Ruirui Lei, Wanggui He, Luu Anh Tuan, Long Chen, Hao Jiang, et al. A comprehensive survey of direct preference optimization: Datasets, theories, variants, and applications. *arXiv preprint arXiv:2410.15595*, 2024.

Yousef Younes, Lukas Galke, and Ansgar Scherp. Radar: A transformer-based autoregressive decoder architecture for hierarchical text classification. In *Proceedings of the 27th European Conference on Artificial Intelligence (ECAI 2024)*, pp. 1559–1566. IOS Press, October 2024. doi: 10.3233/faia240661. URL http://dx.doi.org/10.3233/FAIA240661.

Ping Yu, Jing Xu, Jason Weston, and Ilia Kulikov. Distilling system 2 into system 1. *arXiv preprint arXiv:2407.06023*, 2024. URL https://arxiv.org/abs/2407.06023.

Eric Zelikman, Georges Harik, Yijia Shao, Varuna Jayasiri, Nick Haber, and Noah D Goodman. Quiet-star: Language models can teach themselves to think before speaking. *arXiv preprint arXiv:2403.09629*, 2024.

Dan Zhang, Tao Feng, Lilong Xue, Yuandong Wang, Yuxiao Dong, and Jie Tang. Parameter-efficient fine-tuning for foundation models. *arXiv preprint arXiv:2501.13787*, 2025a.

Jingyi Zhang, Jiaxing Huang, Sheng Jin, and Shijian Lu. Vision-language models for vision tasks: A survey. *IEEE transactions on pattern analysis and machine intelligence*, 46(8):5625–5644, 2024.

Kejia Zhang, Juanjuan Weng, Shaozi Li, and Zhiming Luo. Towards adversarial robustness via debiased high-confidence logit alignment. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 2783–2792, 2025b.

Mingcui Zhang, Ying Li, Tao Wang, and Jianhua Sun. Qr decomposition of dual quaternion matrix and blind watermarking scheme. *Numerical Algorithms*, 99(4):1763–1790, 2025c.

Yuhan Zhang, Jianxiao Zhao, Xi Sun, Yangyang Zheng, Tao Chen, and Zhiwen Wang. Leveraging independent component analysis to unravel transcriptional regulatory networks: A critical review and future directions. *Biotechnology Advances*, 78:108479, 2025d.

Zefeng Zhang, Hengzhu Tang, Jiawei Sheng, Zhenyu Zhang, Yiming Ren, Zhenyang Li, Dawei Yin, Duohe Ma, and Tingwen Liu. Debiasing multimodal large language models via noise-aware preference optimization. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pp. 9423–9433, 2025e.

APPENDIX

Due to space limitations in the main paper, we place additional implementation details, benchmark specifications, and extended experiments in the appendix. Our code and InternVQA examples are available in the supplementary material. The appendix contains the following parts.

A Pseudocode Overview 14
Pseudocode for the full pipeline (Alg. 1, Alg. 2, and Alg. 3). Extends Sec. 3.

B More Stage I/II Experiments 16
Per-backbone public benchmark results, token budget, subspace rank, warm-up space, training data labeling, prior token count, consistency weight, top-k selection, and last-layer training. Extends Secs. 3 and 4.

C More Ablation Studies 22
Component-level ablations on all four backbones, PCA versus learnable pooling, and the stability of logits-wise warm-up. Extends Sec. 4.

D Why logits-wise calibration is more stable 24
Gradient-path analysis of logits-wise versus hidden-wise calibration. Extends Sec. 3.

E More Metrics and Human Evaluation 25
Token F1 and numeric accuracy, the RetainScore and RetainPct definitions, and the single-blind human study. Extends Sec. 4.

F More Efficiency Experiments 28
Inference cost, training resources, and PCA runtime measurements. Extends Sec. 4.

G Matrix Decomposition Methods Evaluations 30
PCA, SVD, ICA, and QR as subspace extractors, and the shared bottleneck dimension. Extends Sec. 3.

H More Experiment Settings 34
Training and hardware settings, the baseline overview, and objective definitions. Extends Sec. 4.

I InternVQA Benchmark Details 35
The composition and evaluation protocol of InternVQA, and its ten task types, three length types, and three image types with examples. Extends Sec. 4.

J More InternVQA Quantitative Experiments 42
Ranking under the absolute direct score and gap, and extended InternVQA results by backbone and subset. Extends Sec. 4.

K InternVQA Qualitative Test 49
Qualitative comparisons of direct answering and caption-then-answering. Extends Sec. 4.

L Failure Cases and Discussion 52
Fine-grained counting under clutter and multilingual robustness. Extends Sec. 5.

M Prompts for Data Construction and Benchmark Evaluation 54
Prompts for Stage I data construction and InternVQA judging. Extends Sec. 4.

N Limitations 56
Caption and judge dependence, failure cases, training cost, and evaluation scope. Extends Sec. 5.

O Future Works 57
Future directions.

P Generation Benchmark Overview 57
The ten public benchmarks used for the retention tests. Extends Sec. 4.

A PSEUDOCODE OVERVIEW

To facilitate reproducibility and implementation, we summarize our training pipeline using three pseudocode blocks. Algorithm 1 presents the overall two-stage procedure, while Algorithm 2 and Algorithm 3 detail the optimization objectives and forward constructions in Stage I and Stage II, respectively.

A.1 Overall Procedure (Algorithm 1).

This pseudocode outlines the end-to-end workflow of INTERNALIZATION, including initialization, Stage I logits-wise calibration, freezing the Stage I branch as a teacher, and Stage II hidden-space prior writing with preference learning and consistency regularization.

Algorithm 1 INTERNALIZATION: From Explicit Description to Implicit Prior (Overall)

Require: Base VLM parameters Θ_0, LM head W_{lm}, internalization module $\Phi = \{W_z, W_h, W_p, W_\ell\}$
Require: Stage I dataset $\mathcal{D}_{\mathrm{S1}}$, Stage II preference dataset $\mathcal{D}_{\mathrm{S2}}$
Require: Hyperparameters: rank r, top-k budgets $k_{\mathrm{img}}, k_{\mathrm{txt}}$, prior count K, β, τ, λ
Ensure: Trained parameters $\Theta^\star$
Initialize $\Theta \leftarrow \Theta_0$, $\Phi \leftarrow$ Xavier
Stage I: Train logits-wise calibration (Alg. 2) to align Φ with base distribution
Teacher setup: Freeze Stage I parameters as teacher branch $\mathcal{T}_{\mathrm{S1}}$
Stage II: Inject hidden-space prior via preference learning and consistency (Alg. 3)
return $\Theta^\star \leftarrow (\Theta, \Phi)$

A.2 Stage I: Logits-wise Stable Distribution Calibration (Algorithm 2).

This pseudocode specifies how we construct online modality subspaces, pool high projected-energy tokens, and produce a lightweight vocabulary-level logit correction. Stage I aligns the newly introduced module to the base model's output distribution to improve stability and avoid early disruption of hidden representations.

Algorithm 2 INTERNALIZATION Stage I: Logits-wise Stable Distribution Calibration

Require: Dataset $\mathcal{D}_{\mathrm{S1}}$, base Θ, internalization Φ, rank r, bottleneck d_z
Require: Trainable set $\Theta^{(1)}_{\mathrm{train}} = \{\text{LLM last layer}, W_{\mathrm{lm}}, \Phi\}$
Ensure: Stage I parameters $(\Theta^{(1)}, \Phi^{(1)})$
for each minibatch $\mathcal{B} \in \mathcal{D}_{\mathrm{S1}}$ **do**
  Forward base VLM to obtain last-layer hidden states $H \in \mathbb{R}^{B\times T\times H}$
  (Online subspace pooling)
  for modality $m \in \{\mathrm{img}, \mathrm{txt}\}$ **do**
    Extract tokens X_m via mask M_m ▷ M_{txt} covers question tokens only; answer tokens are excluded
    Compute PCA basis $B_m \in \mathbb{R}^{H\times r_m}$ from covariance of centered X_m
    Compute projected-energy scores $s_{m,i} = \|(x_{m,i} - \mu_m)^\top B_m\|_2$
    Select top-k_m tokens and mean pool to get $h_m \in \mathbb{R}^{B\times H}$
  end for
  $h_{\mathrm{fuse}} \leftarrow [h_{\mathrm{img}}; h_{\mathrm{txt}}] \in \mathbb{R}^{B\times 2H}$
  (Bottleneck shift)
  $z \leftarrow \tanh(W_z h_{\mathrm{fuse}})$, $\Delta h \leftarrow W_h z$
  $H' \leftarrow H + \Delta h$ ▷ Global hidden shift broadcast over T
  (Logits correction)
  Re-pool H' with modality bases B_m to obtain projections $u \in \mathbb{R}^{B\times 2r}$
  $\Delta\ell \leftarrow W_\ell u \in \mathbb{R}^{B\times V}$
  $\ell_T \leftarrow W_{\mathrm{lm}} H_{:,T,:} + \Delta\ell$ ▷ Correct final-step logits for next-token prediction
  $\mathcal{L}_{\mathrm{S1}} \leftarrow \mathrm{CE}(\ell_T, y)$ ▷ Teacher-forced CE over all answer tokens; $\Delta\ell$ is shared by every supervised position
  Update $\Theta^{(1)}_{\mathrm{train}}$ via backprop
end for
return $(\Theta^{(1)}, \Phi^{(1)})$

A.3 Stage II: Hidden-space Prior Writing with Preference and Consistency (Algorithm 3).

This pseudocode describes how we generate learnable prior tokens and a global hidden shift to induce an "already-captioned" internal context without explicit caption generation. We optimize a

length-normalized preference objective and further apply a chosen-only consistency KL regularizer using the frozen Stage I teacher logits to preserve output style and stabilize training.

Algorithm 3 INTERNALIZATION Stage II: Hidden-space Prior Writing with Preference + Consistency

Require: Preference dataset $\mathcal{D}_{\mathrm{S2}}$, frozen Stage I teacher $\mathcal{T}_{\mathrm{S1}}$, hyperparams K, β, τ, λ
Require: Trainable set $\Theta^{(2)}_{\mathrm{train}}$ (internalization Φ and optionally W_{lm})
Ensure: Final parameters $(\Theta^{(2)}, \Phi^{(2)})$

for each minibatch $\{(I, q, y^+, y^-)\} \in \mathcal{D}_{\mathrm{S2}}$ **do**
  Construct paired batch (Chosen $1:B$, Rejected $B+1:2B$)
  Obtain last-layer hidden states $H \in \mathbb{R}^{2B\times T\times H}$
  (Prior generation)
  Compute h_{fuse} via subspace pooling, $z \leftarrow \tanh(W_z h_{\mathrm{fuse}})$, $\Delta h \leftarrow W_h z$
  $P \leftarrow \mathrm{reshape}(W_p z) \in \mathbb{R}^{2B\times K\times H}$ ▷ Learnable implicit prior tokens
  (Content-adaptive gated injection)
  $\mathcal{H} \leftarrow H + \Delta h$
  $\mathcal{A} \leftarrow \mathrm{softmax}(\mathcal{H}P^\top/\sqrt{H}) \in \mathbb{R}^{2B\times T\times K}$ ▷ Affinity to prior tokens
  $\tilde{\mathcal{P}} \leftarrow \mathcal{A}P$, $\mathbf{g} \leftarrow \sigma(\mathrm{mean}(\mathcal{H}, \mathrm{dim} = -1))$
  $H_{\mathrm{align}} \leftarrow \mathcal{H} + \mathbf{g} \odot \tilde{\mathcal{P}}$ ▷ Gated prior injection
  $\ell_{\mathrm{align}} \leftarrow W_{\mathrm{lm}} H_{\mathrm{align}} \in \mathbb{R}^{2B\times T\times V}$
  (Optimization)
  $\bar{\ell}^+ \leftarrow \textsc{AvgLogProb}(\ell_{\mathrm{align}}[1:B], y^+)$, $\bar{\ell}^- \leftarrow \textsc{AvgLogProb}(\ell_{\mathrm{align}}[B+1:2B], y^-)$
  $\mathcal{L}_{\mathrm{PL}} \leftarrow -\log\sigma(\beta(\bar{\ell}^+ - \bar{\ell}^-))$
  $\ell^T \leftarrow W_{\mathrm{lm}} H[1:B] + \Delta\ell_{\mathrm{S1}}$ ▷ Frozen Stage I teacher logits
  $\mathcal{L}_{\mathrm{KL}} \leftarrow \mathrm{KL}(\mathrm{softmax}(\ell^T/\tau) \| \mathrm{softmax}(\ell_{\mathrm{align}}[1:B]/\tau))$
  $\mathcal{L} \leftarrow \mathcal{L}_{\mathrm{PL}} + \lambda\mathcal{L}_{\mathrm{KL}}$
  Update $\Theta^{(2)}_{\mathrm{train}}$ via backprop
end for
return $(\Theta^{(2)}, \Phi^{(2)})$

B MORE STAGE I/II EXPERIMENTS

Due to space constraints, we place additional experiments in the Appendix, including more ablations on our modules and hyperparameter evaluations.

B.1 PERFORMANCE ON PUBLIC VQA BENCHMARKS

Although `Internalization` mainly targets description reliance, it should not improve InternVQA at the cost of broader VQA ability. This is a common risk in post-training. When the training objective focuses on a specific behavior and the training data is much smaller than the original pre-training and instruction-tuning corpus, the model can easily shift its output style or decision boundary. This often leads to lower performance on standard public benchmarks. In our setting, this risk is natural because the model is explicitly encouraged to answer more directly and rely less on image descriptions.

To examine this effect, we evaluate all methods on ten representative public VQA benchmarks across four backbones. As shown in Tab. 6, `Internalization` achieves the highest Total score and RetainPct on all four backbones. On LLaVA-OneVision-0.5B, our method reaches a Total score of 549.71 and a RetainPct of 96.98%, outperforming the strongest baseline OPERA (542.10, 95.64%). On LLaVA-OneVision-7B, our method achieves 683.44 Total and 95.59% RetainPct, surpassing PAI (681.27, 95.29%). On Qwen3-VL-2B-Instruct, `Internalization` obtains 655.12 Total and 97.63% RetainPct, higher than Visual-RFT (646.05, 96.28%). On Qwen3-VL-8B-Instruct, our method reaches 722.04 Total and 95.60% RetainPct, outperforming PAI (717.75, 95.03%).

These results show that direct-answer improvements coexist with the highest aggregate retention among the compared methods. Aggregate retention combines task-level improvements and regressions. On Qwen3-VL-2B, ChartQA declines from 77.40 to 56.92, while HallusionBench rises from 29.17 to 47.39. Appendix E.1 therefore reports the largest per-task drop on each backbone alongside RetainPct.

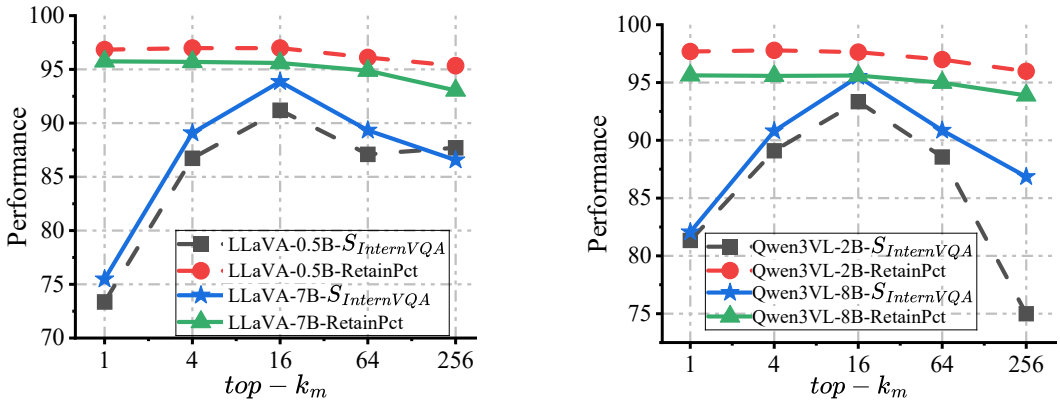
Table 6: **Performance and retention on public VQA benchmarks after post-training.** We report performance on ten public VQA benchmarks across four backbones. **Total** denotes the sum over the ten benchmarks. **RetainPct** denotes the retained percentage relative to the corresponding base model. Higher is better.

Methods	AI2D↑	ChartQA↑	GQA↑	HB↑	MMBench↑	MMStar↑	OCRVQA↑	POPE↑	RWQA↑	SQA↑	Total↑	RetainPct↑
LLaVA-OneVision-0.5B												
Base Model	59.33	60.68	58.32	30.23	59.67	37.13	54.59	87.77	51.37	67.72	566.83	100.00%
PromptCap-2023	51.48	50.92	51.65	27.05	52.97	33.23	47.43	72.95	51.63	60.49	499.79	88.17%
QD-Cap-2024	52.37	55.25	53.74	27.22	52.70	33.71	49.70	77.69	46.42	59.04	507.84	89.59%
PAI-2024	51.37	54.54	49.61	26.28	51.91	32.53	48.51	78.52	44.86	56.75	494.87	87.30%
OPERA-2024	59.64	58.00	55.07	29.30	58.28	36.11	51.87	83.83	48.55	61.45	542.10	95.64%
NaPO-2025	60.78	56.71	54.74	27.71	56.77	37.47	49.33	79.55	45.23	60.54	528.82	93.29%
Visual-RFT-2025	55.47	57.97	54.13	28.88	53.34	34.51	52.86	85.22	48.00	64.17	534.56	94.31%
Fast Quiet-STaR-2025	57.19	55.84	52.37	28.45	54.92	35.76	50.18	81.47	47.66	61.28	525.12	92.64%
COCONUT-2024	54.63	53.71	54.89	27.93	55.41	34.85	51.24	82.19	48.37	60.91	524.13	92.47%
VTool-R1-2025	53.82	56.35	51.76	28.71	53.18	33.94	52.47	80.36	49.85	59.67	520.11	91.76%
GRIT-2025	56.74	54.92	53.45	27.38	56.12	34.29	49.76	84.65	46.91	62.18	526.40	92.87%
Ours	**61.50**	56.83	**55.62**	29.17	56.98	**37.54**	**54.59**	85.00	49.98	62.49	**549.71**	**96.98%**
LLaVA-OneVision-7B												
Base Model	82.58	79.64	62.14	36.15	80.80	62.20	58.20	88.25	69.67	95.29	714.94	100.00%
PromptCap-2023	68.50	67.37	53.29	31.01	66.84	62.77	59.33	71.86	60.92	81.73	623.61	87.23%
QD-Cap-2024	73.14	69.36	53.83	31.05	67.18	51.86	48.43	75.79	63.56	81.24	615.43	86.08%
PAI-2024	75.59	73.49	59.78	35.10	75.73	58.98	57.10	90.22	68.71	86.57	681.27	95.29%
OPERA-2024	76.42	72.16	55.10	32.63	74.22	54.34	52.78	77.57	63.95	84.97	644.14	90.10%
NaPO-2025	75.91	70.98	62.94	32.46	82.72	56.47	52.63	77.16	62.64	84.14	658.06	92.04%
Visual-RFT-2025	76.09	74.63	54.11	32.24	79.11	60.67	53.30	80.07	67.14	87.01	664.37	92.93%
Fast Quiet-STaR-2025	74.85	72.47	56.92	33.67	74.38	57.91	54.76	83.54	65.82	85.29	659.61	92.26%
COCONUT-2024	73.26	71.83	58.41	34.85	76.55	58.73	55.14	84.92	66.37	86.18	666.24	93.19%
VTool-R1-2025	75.12	73.95	57.28	32.79	75.67	57.46	56.81	82.35	64.93	84.76	661.12	92.47%
GRIT-2025	72.68	70.54	55.79	33.21	73.84	56.92	53.67	85.73	67.45	85.64	655.47	91.68%
Ours	**80.90**	**76.79**	57.37	33.12	78.58	58.04	54.05	89.43	68.40	86.76	**683.44**	**95.59%**
Qwen3-VL-2B-Instruct												
Base Model	74.87	77.40	58.55	29.17	75.39	53.03	58.33	88.70	66.27	87.31	670.99	100.00%
PromptCap-2023	68.85	67.22	51.69	26.41	68.47	48.89	51.99	82.05	61.68	88.59	615.84	91.78%
QD-Cap-2024	74.19	75.08	55.44	27.24	69.37	55.15	56.38	81.53	62.99	81.45	638.80	95.20%
PAI-2024	74.01	72.99	53.44	28.12	73.27	52.65	55.51	82.75	63.52	80.45	636.72	94.89%
OPERA-2024	69.38	70.70	55.68	24.87	68.49	54.75	55.75	84.36	64.15	86.18	634.32	94.53%
NaPO-2025	72.51	71.99	54.81	26.52	72.13	51.57	54.80	80.53	68.58	88.39	641.83	95.65%
Visual-RFT-2025	71.90	74.64	56.91	28.74	73.96	51.79	55.21	86.86	61.56	84.48	646.05	96.28%
Fast Quiet-STaR-2025	70.46	72.83	55.27	27.85	71.29	52.64	55.93	84.17	63.74	85.12	639.30	95.28%
COCONUT-2024	71.95	73.41	56.38	26.97	70.85	53.72	57.18	83.64	62.81	84.73	641.64	95.63%
VTool-R1-2025	69.73	71.28	54.62	28.13	69.84	52.38	54.67	85.29	64.36	83.95	634.25	94.52%
GRIT-2025	72.84	70.95	55.13	25.64	71.67	53.19	55.46	82.91	63.28	85.87	636.94	94.93%
Ours	**75.45**	56.92	**57.47**	**47.39**	71.44	54.00	56.25	**88.68**	63.92	83.59	**655.12**	**97.63%**
Qwen3-VL-8B-Instruct												
Base Model	81.70	82.92	61.86	56.41	85.37	69.33	63.83	88.44	70.98	94.40	755.26	100.00%
PromptCap-2023	70.44	70.76	56.17	51.38	77.93	61.22	53.95	79.59	64.24	82.77	677.14	89.66%
QD-Cap-2024	76.89	80.52	58.69	54.10	81.64	63.68	58.33	80.42	64.91	90.35	709.52	93.94%
PAI-2024	72.96	77.89	59.27	56.02	80.23	67.59	62.36	84.86	64.53	92.03	717.75	95.03%
OPERA-2024	75.33	76.42	55.77	49.41	76.84	62.38	59.16	81.83	61.40	91.23	689.75	91.33%
NaPO-2025	74.29	73.42	55.61	50.14	75.89	64.68	57.62	79.80	63.47	90.02	684.93	90.69%
Visual-RFT-2025	73.66	74.02	56.75	50.18	79.85	61.82	63.95	81.47	61.90	87.26	690.86	91.47%
Fast Quiet-STaR-2025	74.57	75.81	57.64	52.73	78.96	63.45	60.72	83.28	67.85	89.14	704.15	93.23%
COCONUT-2024	76.12	74.68	58.37	53.19	79.74	64.21	61.48	82.95	68.73	90.81	710.28	94.04%
VTool-R1-2025	73.84	76.95	56.82	51.87	78.41	62.93	59.67	81.76	66.49	88.92	697.66	92.37%
GRIT-2025	75.46	73.27	57.91	52.45	80.18	63.82	60.94	83.51	67.12	89.67	704.33	93.26%
Ours	**80.38**	76.24	**59.65**	51.46	80.42	64.73	61.30	**88.28**	69.15	90.43	**722.04**	**95.60%**

Compared with caption-based methods, visual evidence intervention methods, Thinking-with-Image methods, and latent reasoning methods, `Internalization` gives a better balance between direct-answer improvement and public benchmark retention. Caption-based methods still rely on explicit descriptions, while visual intervention and tool-based methods can introduce extra processing or distribution changes. Latent reasoning methods move reasoning into hidden states, but their hidden trajectories are difficult to constrain. In contrast, our two-stage design first aligns the new module in the logits space and then injects compact prior tokens with teacher consistency. This allows the model to benefit from description-like evidence without drifting too far from the base model behavior.

B.2 Stage I High Projected-Energy Tokens

We study how the token budget top-k_m in Stage I affects both internalization quality and generalization. Across all four backbones, increasing top-k_m from 1 to 16 consistently improves $S_{\texttt{InternVQA}}$, indicating that too few tokens underfit the semantic evidence needed by the online subspace pooling and logits calibration. However, further enlarging top-k_m to 64 or 256 leads to a clear drop in $S_{\texttt{InternVQA}}$, suggesting that an overly large token set introduces redundant or weakly relevant tokens that dilute the estimated subspace and reduce the effectiveness of alignment. RetainPct remains high and varies more smoothly, but it tends to slightly decrease when top-k_m becomes large, which



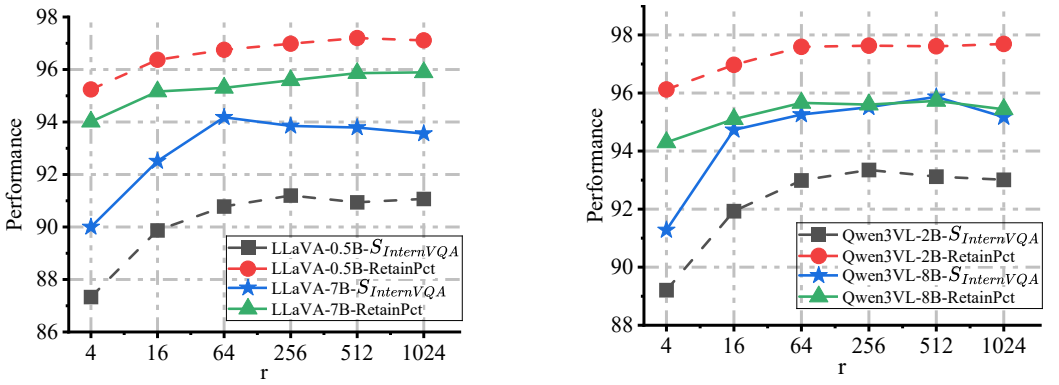
(a) LLaVA-OneVision (0.5B, 7B). (b) Qwen3-VL (2B, 8B).

Figure 4: **Sensitivity to the number of high projected-energy tokens in Stage I.** We vary top-$k_m \in \{1, 4, 16, 64, 256\}$ when selecting high projected-energy tokens for online subspace pooling. The solid curves report $S_{\texttt{InternVQA}}$ (higher is better), and the dashed curves report RetainPct (higher is better).

matches the intuition that noisier calibration signals can cause mild style drift and harm generalization. We also observe a scale effect: larger models (LLaVA-7B and Qwen3-VL-8B) are more robust to top-k_m, while smaller ones (LLaVA-0.5B and Qwen3-VL-2B) exhibit stronger sensitivity under large top-k_m. Overall, top-$k_m = 16$ provides the most stable and strong trade-off across backbones, and we adopt it as default.

B.3 Stage I Subspace Rank r Experiments

We study how the subspace rank r used by the online token-space construction affects both $S_{\texttt{InternVQA}}$ and **RetainPct** across four backbones, as shown in Fig.5.



(a) LLaVA-OneVision (0.5B, 7B). (b) Qwen3-VL (2B, 8B).

Figure 5: **Stage I subspace rank sensitivity.** We vary the subspace rank $r \in \{4, 16, 64, 256, 512, 1024\}$ used by the online token-space construction. The bar curves report $S_{\texttt{InternVQA}}$ (higher is better), and the line curves report **RetainPct** (higher means better preservation of the base model generalization).

Across all four backbones, increasing r from very small values brings consistent gains, indicating that a too small subspace cannot capture enough semantic variance from image and question tokens to form a useful Stage I signal. In particular, both $S_{\texttt{InternVQA}}$ and RetainPct improve rapidly when moving from $r = 4$ to $r = 64$.

When r becomes moderate to large, the trend saturates. RetainPct keeps improving or stays stable as r increases, suggesting that a richer subspace helps Stage I alignment preserve the base behavior more reliably. In contrast, $S_{\texttt{InternVQA}}$ often peaks around a mid-range rank (typically $r = 256$ or $r = 512$) and can slightly drop at $r = 1024$ on some backbones, which matches the intuition that an

overly large subspace may include less stable or noisy directions, yielding diminishing returns for InternVQA.

We also observe model-scale differences. Smaller backbones tend to be more sensitive to the choice of r, while larger backbones show smoother curves and earlier saturation, indicating better robustness to the subspace capacity. Overall, a mid-range rank provides the best trade-off, and we use $r = 256$ as the default setting in the main experiments.

B.4 Stage I Logits-Wise versus Hidden-Wise Warm-Up

Stage I aims to prevent early distribution shift introduced by newly added, randomly initialized parameters. We therefore warm up the internalization module in the logits space, aligning the output distribution before training hidden-state prior injection with caption-derived answers in Stage II. Tab. 7 validates this design choice. Across all four backbones, **logits-wise warm-up** yields higher $S_{\texttt{InternVQA}}$ and higher **RetainPct** than hidden-wise warm-up. This trend matches our analysis: logits-wise calibration provides a shallow and better-conditioned update path to the new parameters, which stabilizes optimization and reduces early style drift.

Table 7: Logits-wise warm-up improves $S_{\texttt{InternVQA}}$ and RetainPct.

Backbone	**Warm-up**	$S_{\texttt{InternVQA}}$ ↑	**RetainPct**↑
Qwen3-VL-2B	logits-wise	**93.35**	**97.63**
	hidden-wise	91.71	92.82
Qwen3-VL-8B	logits-wise	**95.51**	**95.60**
	hidden-wise	92.02	90.62
LLaVA-0.5B	logits-wise	**91.20**	**96.98**
	hidden-wise	86.73	92.74
LLaVA-7B	logits-wise	**93.85**	**95.59**
	hidden-wise	88.93	89.78

B.5 Training Data Construction and Labeling Strategy

Stage I aims to stabilize the newly added module by aligning it with the base model before we inject any hidden-space prior in Stage II. A key design choice is how to label the 280K VQA pairs. Using only base-model answers yields a conservative supervision signal that can inherit base-model uncertainty, while using only GPT-5 answers may deviate from the base model distribution and cause larger style drift. We therefore adopt a mixed labeling strategy, where half of the samples use base-model answers and the other half use GPT-5 answers. For Stage II, we further filter the preference pairs with a GPT-5 consistency check: we keep a pair only when the description-based final answer is judged logically correct and better aligned with the question than the unconstrained answer. This filtering removes 3.9% noisy pairs before preference training and reduces label conflicts in the chosen/rejected supervision. Tab.8 reports the ablation results across four backbones.

Tab. 8 shows that the mixed strategy gives the highest $S_{\texttt{InternVQA}}$ on every backbone and the highest RetainPct on three of four. Compared with 280K GPT-5 labels, the 140K/140K mixture improves both metrics on all four backbones. On Qwen3-VL-2B, for example, $S_{\texttt{InternVQA}}$ rises from 91.71 to 93.35 and RetainPct from 94.62% to 97.63%. Using 280K base-model labels yields 87.62 and 96.79%, respectively. These results motivate combining the two label sources to balance task adaptation and retention.

This comparison varies the Stage I label source within the full training pipeline. It supports the mixed-label design, while the SFT and prior-injection comparisons separately assess answer supervision and the prior module.

B.6 Stage II More Implicit Prior Experiments

We study the effect of Stage II implicit prior token count K in Tab.9. We observe that increasing K consistently improves w/o desc and $S_{\texttt{InternVQA}}$ across backbones, but the gains are front loaded: most

Table 8: **Stage I labeling strategy ablation.** We compare three ways to label the 280K Stage I VQA pairs. The mixed strategy uses 140K GPT-5 answers and 140K base-model answers. We report $S_{\texttt{InternVQA}}$ and RetainPct, where higher is better.

Backbone	Stage I labels	$S_{\texttt{InternVQA}}\uparrow$	RetainPct↑
Qwen3-VL-2B	140K GPT-5 + 140K BaseModel	**93.35**	**97.63**
	280K GPT-5	91.71	94.62
	280K BaseModel	87.62	96.79
Qwen3-VL-8B	140K GPT-5 + 140K BaseModel	**95.51**	95.60
	280K GPT-5	92.64	92.89
	280K BaseModel	92.88	**95.94**
LLaVA-OneVision-0.5B	140K GPT-5 + 140K BaseModel	**91.20**	**96.98**
	280K GPT-5	90.76	93.87
	280K BaseModel	88.82	95.02
LLaVA-OneVision-7B	140K GPT-5 + 140K BaseModel	**93.85**	**95.59**
	280K GPT-5	91.06	91.58
	280K BaseModel	90.45	93.47

Table 9: Experiments on the number of Stage II implicit prior tokens. We vary the prior token count K and report performance under direct answering (w/o desc), caption-then-answering (w/ desc), the proposed $S_{\text{InternVQA}}$ and RetainPct across four backbones. Gray cells indicate the best result within each backbone.

#Implicit Prior	Qwen3-VL-2B				Qwen3-VL-8B			
	w/o desc↑	w/ desc↑	$S_{\texttt{InternVQA}}\uparrow$	RetainPct↑	w/o desc↑	w/ desc↑	$S_{\texttt{InternVQA}}\uparrow$	RetainPct↑
K=1	52.02	59.32	85.97	97.52	63.62	68.47	92.38	96.05
K=4	58.02	62.81	91.74	97.30	70.87	74.77	94.5	94.26
K=16	62.82	67.00	93.35	97.63	74.53	77.88	95.51	95.60
K=64	62.9	66.78	93.83	96.52	75.02	77.95	96.09	94.27
K=256	63.47	67.41	93.79	96.64	73.06	77.69	93.66	95.09
#Implicit Prior	**LLaVA-OneVision-0.5B**				**LLaVA-OneVision-7B**			
	w/o desc↑	w/ desc↑	$S_{\texttt{InternVQA}}\uparrow$	RetainPct↑	w/o desc↑	w/ desc↑	$S_{\texttt{InternVQA}}\uparrow$	RetainPct↑
K=1	41.81	49.36	81.94	96.94	56.71	61.89	90.87	95.74
K=4	44.23	51.71	83.09	93.22	61.56	66.82	91.46	95.36
K=16	50.14	54.55	91.20	96.98	64.93	68.92	93.85	95.59
K=64	50.26	54.67	91.23	92.52	65.12	69.08	93.92	94.77
K=256	50.43	54.88	91.18	92.21	65.26	69.38	93.69	93.22

of the improvement happens from $K = 1$ to $K = 16$, and then saturates. The observed gains saturate at moderate K, while larger K offers limited additional improvement and can reduce RetainPct. Overall, $K = 16$ gives the best trade-off between internalization and generalization.

B.7 Stage II Consistency Weight

We study the weight λ of the chosen-only KL consistency loss in Stage II, as shown in Fig. 6. Overall, λ controls a clear trade-off between preserving the base model behavior and strengthening direct answering without captions.

B.8 Top-k Evaluations

Tab. 10 validates the token selection strategy in Stage I. Selecting top-k high projected-energy tokens consistently yields the best $S_{\texttt{InternVQA}}$ and RetainPct across all four backbones. In contrast, Bottom-k selection causes a large drop on both metrics, showing that low projected-energy selection is less effective in this setting. Random-k performs between top-k and bottom-k, supporting projected-energy selection under the tested token budget.

B.9 Training only Last Layer

Tab. 11 validates the layer selection strategy. Training all layers yields slightly higher $S_{\texttt{InternVQA}}$ on some backbones, but consistently reduces RetainPct, suggesting larger distribution drift and weaker generalization. In contrast, training only the last layer achieves the best RetainPct on all four

Table 10: **Verifying top-k semantic token selection in Stage I.** We compare selecting top-k high projected-energy tokens, bottom-k tokens, and random-k tokens for prior construction. Top-k achieves the best $S_{\texttt{InternVQA}}$ and RetainPct across all backbones.

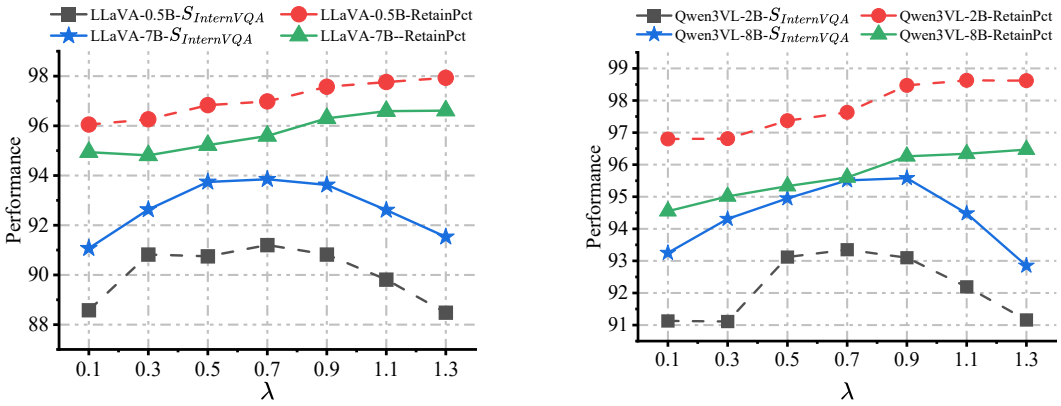
Backbone	Selection	$S_{\texttt{InternVQA}}\uparrow$	RetainPct↑
Qwen3VL-2B	Top-k	**93.35**	**97.63**
	Bottom-k	84.61	83.71
	Random-k	91.87	95.73
Qwen3VL-8B	Top-k	**95.51**	**95.60**
	Bottom-k	89.62	89.71
	Random-k	93.12	92.17
LLaVA-OneVision-0.5B	Top-k	**91.20**	**96.98**
	Bottom-k	86.52	87.66
	Random-k	88.92	93.72
LLaVA-OneVision-7B	Top-k	**93.85**	**95.59**
	Bottom-k	86.02	88.29
	Random-k	91.16	92.34

Table 11: **Verifying last-layer training.** We compare training the last layer, all layers, or a random layer. Last-layer training gives the highest RetainPct across all backbones, with competitive $S_{\texttt{InternVQA}}$.

Backbone	Method	$S_{\texttt{InternVQA}}\uparrow$	RetainPct↑
Qwen3VL-2B	Last Layer	93.35	**97.63**
	All Layer	**94.02**	91.81
	Random Layer	91.66	95.12
Qwen3VL-8B	Last Layer	95.51	**95.60**
	All Layer	**95.82**	88.61
	Random Layer	92.87	94.61
LLaVA-OneVision-0.5B	Last Layer	91.20	**96.98**
	All Layer	**91.26**	88.56
	Random Layer	89.15	93.25
LLaVA-OneVision-7B	Last Layer	**93.85**	**95.59**
	All Layer	93.12	91.61
	Random Layer	91.01	93.82
Average Performance (Avg)	Last Layer	93.48	**96.45**
Average Performance (Avg)	All Layer	**93.56**	90.15
Average Performance (Avg)	Random Layer	91.17	94.20

backbones, while keeping $S_{\texttt{InternVQA}}$ competitive. Therefore, we adopt last-layer training as the default setting for a better stability and generalization tradeoff.

As λ increases, RetainPct consistently improves or saturates across all backbones, indicating that a stronger teacher constraint better preserves the base model distribution and reduces style drift during preference training. Meanwhile, $S_{\texttt{InternVQA}}$ follows a non monotonic trend, typically peaking at a moderate λ and then dropping when λ becomes too large. This behavior matches the intuition that a small λ is insufficient to regularize preference optimization, while an overly large λ over constrains the student logits and suppresses the preference signal that improves direct answering. Across model sizes, smaller models are more sensitive to large λ, showing a clearer degradation of $S_{\texttt{InternVQA}}$ in the high λ regime, whereas larger models retain higher RetainPct and show a milder decline. We therefore use $\lambda = 0.7$ as the default, which provides a strong and stable balance between improving $S_{\texttt{InternVQA}}$ and maintaining RetainPct.



(a) LLaVA OneVision backbones.

(b) Qwen3 VL backbones.

Figure 6: **Stage II consistency weight sensitivity.** We vary the KL consistency weight λ and report $S_{\texttt{InternVQA}}$ and RetainPct on four backbones. RetainPct is shown by the dashed curves, while $S_{\texttt{InternVQA}}$ is shown by the solid curves.

C More Ablation Studies

C.1 Component Level Ablation Analysis

To further understand which components matter in our post-training, we report additional ablations on two LLaVA-OneVision backbones and two Qwen3-VL backbones. We evaluate (1) **generalization retention** on ten public VQA benchmarks and report **Total** and **RetainPct**, and (2) **InternVQA** under **direct answering** (w/o Desc), **caption-then-answering** (WithDesc), and the aggregated score $S_{\texttt{InternVQA}}$. In all tables, ⟶ denotes an intra-stage ablation: we keep the corresponding stage enabled and remove only the specified sub-module (all other components and training settings remain unchanged).

Table 12: LLaVA-OneVision ablation experiments. ← indicates an *intra-stage ablation*: we keep the corresponding stage enabled and only remove the specified sub-module within that stage (all other components and training settings remain unchanged).

Method	AI2D	ChartQA	GQA	HB	MMBench	MMStar	OCRVQA	POPE	RWQA	SQA	Total	RetainPct
LLaVA-OneVision-0.5B												
Ours	61.50	56.83	55.62	29.17	56.98	37.54	54.59	85.00	49.98	62.49	549.71	96.98%
w/o stageI	54.95	51.64	48.77	25.89	49.59	33.54	51.81	79.45	45.60	56.58	497.82	87.83%
△ vs Ours (w/o stageI)	-6.55	-5.19	-6.85	-3.28	-7.39	-4.00	-2.78	-5.55	-4.38	-5.91	-51.89	-9.15%
⟶w/o stageI repeated pooling	59.02	53.77	52.44	24.75	54.61	33.29	50.10	82.84	46.95	58.34	516.11	91.05%
△ vs Ours (⟶w/o stageI repeated pooling)	-2.48	-3.06	-3.18	-4.42	-2.37	-4.25	-4.49	-2.16	-3.03	-4.15	-33.60	-5.93%
⟶w/o stageI z	59.24	54.22	53.38	25.91	52.50	35.64	51.35	81.05	46.33	59.64	519.26	91.61%
△ vs Ours (⟶w/o stageI z)	-2.26	-2.61	-2.24	-3.26	-4.48	-1.90	-3.24	-3.95	-3.65	-2.85	-30.45	-5.37%
⟶w/o token selecting	59.91	54.40	53.55	26.86	54.23	35.02	52.15	82.93	46.93	61.11	527.09	92.99%
△ vs Ours (⟶w/o token selecting)	-1.59	-2.43	-2.07	-2.31	-2.75	-2.52	-2.44	-2.07	-3.05	-1.38	-22.62	-3.99%
w/o stageII	57.52	53.19	49.32	21.32	54.31	31.41	50.48	80.95	44.66	59.11	502.27	88.61%
△ vs Ours (w/o stageII)	-3.98	-3.64	-6.30	-7.85	-2.67	-6.13	-4.11	-4.05	-5.32	-3.38	-47.44	-8.37%
⟶w/o stageII gating	58.96	53.78	52.34	26.05	55.02	33.90	50.78	83.12	46.67	58.89	519.51	91.65%
△ vs Ours (⟶w/o stageII gating)	-2.54	-3.05	-3.28	-3.12	-1.96	-3.64	-3.81	-1.88	-3.31	-3.60	-30.20	-5.33%
⟶w/o stageII prior injecting	58.33	53.28	52.43	25.79	54.72	33.89	52.59	80.78	46.82	60.33	518.96	91.55%
△ vs Ours (⟶w/o stageII prior injecting)	-3.17	-3.55	-3.19	-3.38	-2.26	-3.65	-2.00	-4.22	-3.16	-2.16	-30.75	-5.43%
⟶w/o stageII $\mathcal{L}_{\text{PL}}$	58.37	53.61	53.13	27.39	54.36	34.17	52.87	81.90	46.69	60.23	522.72	92.22%
△ vs Ours (⟶w/o stageII $\mathcal{L}_{\text{PL}}$)	-3.13	-3.22	-2.49	-1.78	-2.62	-3.37	-1.72	-3.10	-3.29	-2.26	-26.99	-4.76%
⟶w/o stageII $\mathcal{L}_{\text{KL}}$	57.26	52.00	52.52	23.53	52.18	32.59	51.74	80.66	45.86	57.82	506.16	89.30%
△ vs Ours (⟶w/o stageII $\mathcal{L}_{\text{KL}}$)	-4.24	-4.83	-3.10	-5.64	-4.80	-4.95	-2.85	-4.34	-4.12	-4.67	-43.55	-7.68%
w/o shared modules	52.47	50.55	47.59	21.01	48.30	27.86	46.75	77.17	44.94	53.24	469.88	82.90%
△ vs Ours (w/o shared modules)	-9.03	-6.28	-8.03	-8.16	-8.68	-9.68	-7.84	-7.83	-5.04	-9.25	-79.83	-14.08%
LLaVA-OneVision-7B												
Ours	80.90	76.79	57.37	33.12	78.58	58.04	54.05	89.43	68.40	86.76	683.44	95.59%
w/o stageI	76.16	72.40	51.31	29.37	75.33	54.42	50.44	83.32	65.27	83.44	641.46	89.72%
△ vs Ours (w/o stageI)	-4.74	-4.39	-6.06	-3.75	-3.25	-3.62	-3.61	-6.11	-3.13	-3.32	-41.98	-5.87%
⟶w/o stageI repeated pooling	77.80	73.80	54.63	28.67	75.01	55.22	50.37	88.09	65.65	84.76	654.00	91.47%
△ vs Ours (⟶w/o stageI repeated pooling)	-3.10	-2.99	-2.74	-4.45	-3.57	-2.82	-3.68	-1.34	-2.75	-2.00	-29.44	-4.12%
⟶w/o stageI z	77.24	73.46	54.45	28.90	75.18	55.94	50.84	86.74	65.31	83.75	651.81	91.17%
△ vs Ours (⟶w/o stageI z)	-3.66	-3.33	-2.92	-4.22	-3.40	-2.10	-3.21	-2.69	-3.09	-3.01	-31.63	-4.42%
⟶w/o token selecting	77.63	74.79	54.68	31.57	77.17	55.47	51.64	87.47	67.00	84.92	662.34	92.64%
△ vs Ours (⟶w/o token selecting)	-3.27	-2.00	-2.69	-1.55	-1.41	-2.57	-2.41	-1.96	-1.40	-1.84	-21.10	-2.95%
w/o stageII	78.19	74.05	51.70	24.31	73.83	54.09	49.07	86.84	62.80	84.04	638.92	89.36%
△ vs Ours (w/o stageII)	-2.71	-2.74	-5.67	-8.81	-4.75	-3.95	-4.98	-2.59	-5.60	-2.72	-44.52	-6.23%
⟶w/o stageII gating	77.39	73.81	53.21	29.27	75.11	55.50	52.91	85.86	64.91	84.57	652.54	91.27%
△ vs Ours (⟶w/o stageII gating)	-3.51	-2.98	-4.16	-3.85	-3.47	-2.54	-1.14	-3.57	-3.49	-2.19	-30.90	-4.32%
⟶w/o stageII prior injecting	78.76	74.21	54.06	29.98	74.59	55.56	50.43	85.90	66.35	84.22	654.06	91.48%
△ vs Ours (⟶w/o stageII prior injecting)	-2.14	-2.58	-3.31	-3.14	-3.99	-2.48	-3.62	-3.53	-2.05	-2.54	-29.38	-4.11%
⟶w/o stageII $\mathcal{L}_{\text{PL}}$	77.86	74.51	56.23	31.29	75.38	56.80	51.29	87.02	66.57	85.45	662.40	92.65%
△ vs Ours (⟶w/o stageII $\mathcal{L}_{\text{PL}}$)	-3.04	-2.28	-1.14	-1.83	-3.20	-1.24	-2.76	-2.41	-1.83	-1.31	-21.04	-2.94%
⟶w/o stageII $\mathcal{L}_{\text{KL}}$	77.14	72.18	54.92	29.45	74.91	54.80	49.09	86.74	64.30	82.86	646.39	90.41%
△ vs Ours (⟶w/o stageII $\mathcal{L}_{\text{KL}}$)	-3.76	-4.61	-2.45	-3.67	-3.67	-3.24	-4.96	-2.69	-4.10	-3.90	-37.05	-5.18%
w/o shared modules	72.15	70.88	50.63	25.27	72.90	50.25	49.35	80.81	60.61	79.88	612.73	85.70%
△ vs Ours (w/o shared modules)	-8.75	-5.91	-6.74	-7.85	-5.68	-7.79	-4.70	-8.62	-7.79	-6.88	-70.71	-9.89%

Table 13: Qwen3-VL ablation experiments. ← indicates an *intra-stage ablation*: we keep the corresponding stage enabled and only remove the specified sub-module within that stage (all other components and training settings remain unchanged).

Method	AI2D	ChartQA	GQA	HB	MMBench	MMStar	OCRVQA	POPE	RWQA	SQA	Total	RetainPct
Qwen3-VL-2B-Instruct												
Ours	75.45	56.92	57.47	47.39	71.44	54.00	56.25	88.68	63.92	83.59	655.12	97.63%
w/o stageI	69.08	52.24	53.08	42.20	66.36	48.57	51.34	84.84	57.92	79.76	605.39	90.22%
△ **vs Ours (w/o stageI)**	-6.37	-4.68	-4.39	-5.19	-5.08	-5.43	-4.91	-3.84	-6.00	-3.83	-49.73	-7.41%
↳w/o stageI repeated pooling	72.22	55.34	53.75	44.93	67.70	52.58	54.55	85.01	61.15	80.94	628.17	93.61%
△ **vs Ours (↳w/o stageI repeated pooling)**	-3.23	-1.58	-3.72	-2.46	-3.74	-1.42	-1.70	-3.67	-2.77	-2.65	-26.95	-4.02%
↳w/o stageI z	72.63	52.40	54.46	44.27	67.61	51.34	52.31	86.01	61.23	80.22	622.48	92.77%
△ **vs Ours (↳w/o stageI z)**	-2.82	-4.52	-3.01	-3.12	-3.83	-2.66	-3.94	-2.67	-2.69	-3.37	-32.64	-4.86%
↳w/o token selecting	73.23	55.44	54.41	46.13	70.33	50.50	54.31	86.93	63.32	81.99	636.59	94.87%
△ **vs Ours (↳w/o token selecting)**	-2.22	-1.48	-3.06	-1.26	-1.11	-3.50	-1.94	-1.75	-0.60	-1.60	-18.53	-2.76%
w/o stageII	71.09	51.95	54.00	43.11	65.45	50.86	52.52	85.38	59.21	80.70	614.27	91.54%
△ **vs Ours (w/o stageII)**	-4.36	-4.97	-3.47	-4.28	-5.99	-3.14	-3.73	-3.30	-4.71	-2.89	-40.85	-6.09%
↳w/o stageII gating	73.48	53.71	54.22	45.96	68.65	51.55	53.95	86.87	60.16	81.41	629.96	93.88%
△ **vs Ours (↳w/o stageII gating)**	-1.97	-3.21	-3.25	-1.43	-2.79	-2.45	-2.30	-1.81	-3.76	-2.18	-25.16	-3.75%
↳w/o stageII prior injecting	73.17	54.18	53.58	43.40	69.80	50.71	53.93	86.74	60.77	79.98	626.26	93.33%
△ **vs Ours (↳w/o stageII prior injecting)**	-2.28	-2.74	-3.89	-3.99	-1.64	-3.29	-2.32	-1.94	-3.15	-3.61	-28.86	-4.30%
↳w/o stageII $\mathcal{L}_{\mathrm{PL}}$	73.64	55.45	55.05	44.16	69.59	50.48	54.17	87.12	61.76	81.88	633.30	94.38%
△ **vs Ours (↳w/o stageII $\mathcal{L}_{\mathrm{PL}}$)**	-1.81	-1.47	-2.42	-3.23	-1.85	-3.52	-2.08	-1.56	-2.16	-1.71	-21.82	-3.25%
↳w/o stageII $\mathcal{L}_{\mathrm{KL}}$	70.99	53.38	52.71	43.21	67.59	47.98	51.32	85.30	60.55	80.77	613.80	91.47%
△ **vs Ours (↳w/o stageII $\mathcal{L}_{\mathrm{KL}}$)**	-4.46	-3.54	-4.76	-4.18	-3.85	-6.02	-4.93	-3.38	-3.37	-2.82	-41.32	-6.16%
w/o shared modules	70.08	49.44	48.21	42.20	65.88	48.48	50.69	83.78	54.68	76.74	590.18	87.95%
△ **vs Ours (w/o shared modules)**	-5.37	-7.48	-9.26	-5.19	-5.56	-5.52	-5.56	-4.90	-9.24	-6.85	-64.94	-9.68%
Qwen3-VL-8B-Instruct												
Ours	80.38	76.24	59.65	51.46	80.42	64.73	61.30	88.28	69.15	90.43	722.04	95.60%
w/o stageI	75.93	70.67	53.60	43.76	75.47	58.33	56.83	85.03	65.81	86.51	671.94	88.97%
△ **vs Ours (w/o stageI)**	-4.45	-5.57	-6.05	-7.70	-4.95	-6.40	-4.47	-3.25	-3.34	-3.92	-50.10	-6.63%
↳w/o stageI repeated pooling	76.22	74.69	56.54	48.39	78.39	62.79	59.03	85.60	65.25	87.18	694.08	91.90%
△ **vs Ours (↳w/o stageI repeated pooling)**	-4.16	-1.55	-3.11	-3.07	-2.03	-1.94	-2.27	-2.68	-3.90	-3.25	-27.96	-3.70%
↳w/o stageI z	77.45	72.93	56.24	49.55	77.54	60.87	58.38	85.98	66.29	87.84	693.07	91.76%
△ **vs Ours (↳w/o stageI z)**	-2.93	-3.31	-3.41	-1.91	-2.88	-3.86	-2.92	-2.30	-2.86	-2.59	-28.97	-3.84%
↳w/o token selecting	77.64	74.33	58.34	50.18	77.67	63.66	59.31	86.78	67.17	88.18	703.26	93.11%
△ **vs Ours (↳w/o token selecting)**	-2.74	-1.91	-1.31	-1.28	-2.75	-1.07	-1.99	-1.50	-1.98	-2.25	-18.78	-2.49%
w/o stageII	75.20	72.23	55.98	48.99	76.16	59.35	56.73	82.80	64.47	84.43	676.34	89.55%
△ **vs Ours (w/o stageII)**	-5.18	-4.01	-3.67	-2.47	-4.26	-5.38	-4.57	-5.48	-4.68	-6.00	-45.70	-6.05%
↳w/o stageII gating	78.39	74.04	55.96	48.87	78.81	61.85	57.51	85.80	66.66	88.97	696.86	92.27%
△ **vs Ours (↳w/o stageII gating)**	-1.99	-2.20	-3.69	-2.59	-1.61	-2.88	-3.79	-2.48	-2.49	-1.46	-25.18	-3.33%
↳w/o stageII prior injecting	77.06	74.04	55.30	47.66	77.20	60.79	57.80	84.88	65.79	87.69	688.21	91.12%
△ **vs Ours (↳w/o stageII prior injecting)**	-3.32	-2.20	-4.35	-3.80	-3.22	-3.94	-3.50	-3.40	-3.36	-2.74	-33.83	-4.48%
↳w/o stageII $\mathcal{L}_{\mathrm{PL}}$	77.28	72.96	57.62	47.82	79.08	62.02	59.08	84.94	67.16	87.63	695.59	92.10%
△ **vs Ours (↳w/o stageII $\mathcal{L}_{\mathrm{PL}}$)**	-3.10	-3.28	-2.03	-3.64	-1.34	-2.71	-2.22	-3.34	-1.99	-2.80	-26.45	-3.50%
↳w/o stageII $\mathcal{L}_{\mathrm{KL}}$	77.23	74.13	57.12	46.13	75.15	61.50	56.04	84.71	64.77	86.51	683.29	90.47%
△ **vs Ours (↳w/o stageII $\mathcal{L}_{\mathrm{KL}}$)**	-3.15	-2.11	-2.53	-5.33	-5.27	-3.23	-5.26	-3.57	-4.38	-3.92	-38.75	-5.13%
w/o shared modules	72.87	71.62	53.56	46.61	73.50	59.25	56.64	80.73	64.58	82.21	661.57	87.59%
△ **vs Ours (w/o shared modules)**	-7.51	-4.62	-6.09	-4.85	-6.92	-5.48	-4.66	-7.55	-4.57	-8.22	-60.47	-8.01%

Tab.12 shows that both stages matter for retaining broad VQA ability. On LLaVA-OneVision-0.5B, **Ours** reaches **Total** 549.71 and **RetainPct** 96.98%. Removing Stage I drops Total by 51.89 to 497.82 and reduces RetainPct by 9.15%. Removing Stage II yields a comparable degradation, with Total 502.27 and RetainPct 88.61%. On LLaVA-OneVision-7B, **Ours** reaches Total 683.44 and RetainPct 95.59%. Removing Stage I and Stage II drops Total by 41.98 and 44.52, and reduces RetainPct by 5.87% and 6.23%, respectively.

Intra-stage ablations confirm the key stabilizers. Within Stage I, removing repeated pooling or z produces larger drops than removing token selecting. For example, on 0.5B, ↳ w/o token selecting reduces Total by 22.62 and RetainPct by 3.99%, while ↳ w/o repeated pooling reduces Total by 33.60 and RetainPct by 5.93%. Within Stage II, $\mathcal{L}_{\mathrm{KL}}$ plays a stronger role than $\mathcal{L}_{\mathrm{PL}}$ for retention. On 0.5B, ↳ w/o $\mathcal{L}_{\mathrm{KL}}$ drops Total by 43.55 and RetainPct by 7.68%, while ↳ w/o $\mathcal{L}_{\mathrm{PL}}$ drops Total by 26.99 and RetainPct by 4.76%. Finally, removing **shared modules** causes the largest damage on both backbones, with Total drops of 79.83 (0.5B) and 70.71 (7B), and RetainPct drops of 14.08% (0.5B) and 9.89% (7B), indicating that shared modules anchor generalization across diverse benchmarks.

Tab.13 exhibits the same pattern on Qwen3-VL backbones. On Qwen3-VL-2B-Instruct, **Ours** reaches Total 655.12 and RetainPct 97.63%. Removing Stage I drops Total by 49.73 and RetainPct by 7.41%. Removing Stage II drops Total by 40.85 and RetainPct by 6.09%. On Qwen3-VL-8B-Instruct, **Ours** reaches Total 722.04 and RetainPct 95.60%. Removing Stage I drops Total by 50.10 and RetainPct by 6.63%. Removing Stage II drops Total by 45.70 and RetainPct by 6.05%.

Intra-stage ablations follow consistent importance ordering. Within Stage I, token selecting is the least sensitive component, while repeated pooling and z contribute more. For example, on 2B, ↳ w/o token selecting drops Total by 18.53 (RetainPct −2.76%), while ↳ w/o z drops Total by 32.64 (RetainPct −4.86%). Within Stage II, $\mathcal{L}_{\mathrm{KL}}$ again matters more than $\mathcal{L}_{\mathrm{PL}}$ for retention. On 2B, ↳ w/o $\mathcal{L}_{\mathrm{KL}}$ drops Total by 41.32 and RetainPct by 6.16%, while ↳ w/o $\mathcal{L}_{\mathrm{PL}}$ drops Total by 21.82 and RetainPct by 3.25%. Removing **shared modules** produces the largest retention collapse on both Qwen backbones, with Total drops of 64.94 (2B) and 60.47 (8B), confirming that shared modules provide a critical generalization anchor.

The prior-injection ablation as a same-pairs preference control. The ↳ w/o stageII prior injecting row is the closest control in this paper to preference optimization on the same Stage II pairs without the prior-token architecture: it keeps the preference pairs, the trainable parameter set, $\mathcal{L}_{\mathrm{PL}}$, $\mathcal{L}_{\mathrm{KL}}$, the Stage I initialization, and the global shift $\Delta\mathcal{H}$, and only removes the gated attention through which the K prior tokens reach the answer positions. Its $S_{\texttt{InternVQA}}$ is 75.40, 81.98, 80.38, and 88.31 on LLaVA-OneVision-0.5B, LLaVA-OneVision-7B, Qwen3-VL-2B, and Qwen3-VL-8B (average 81.52), against 91.20, 93.85, 93.35, and 95.51 (average 93.48) for the full method and 25.66, 45.03, 61.80, and 73.95 (average 51.61) for the base models; its RetainPct is 91.55%, 91.48%, 93.33%, and 91.12% (average 91.87%), against 96.45% on average for the full method. This ablation supports the contribution of prior injection within the two-stage training pipeline.

C.2 PCA-BASED POOLING VS. LEARNABLE POOLING

A natural alternative for Stage I is to replace the online PCA-based pooling with a lightweight learnable pooling block. Tab. 14 shows that the PCA-based design consistently yields higher $S_{\texttt{InternVQA}}$ and higher RetainPct across all four backbones. This result supports our design motivation: Stage I is meant to be a *stable alignment module*, not a second strong learner. A learnable pooling block can absorb task-specific fitting pressure too early, which weakens the warm-up role and increases distribution drift. In contrast, the online PCA-based pooling is sample-adaptive and low-parameter, so it keeps Stage I closer to distribution alignment while still extracting useful image-question cues.

Table 14: PCA-based pooling vs. learnable pooling in Stage I.

Backbone	**Pooling**	$S_{\texttt{InternVQA}}\uparrow$	**RetainPct**↑
Qwen3-VL-2B	PCA-based	**93.35**	**97.63**
	learnable	91.22	96.02
Qwen3-VL-8B	PCA-based	**95.51**	**95.60**
	learnable	93.72	94.24
LLaVA-0.5B	PCA-based	**91.20**	**96.98**
	learnable	88.15	93.16
LLaVA-7B	PCA-based	**93.85**	**95.59**
	learnable	91.27	92.06

C.3 WHY LOGITS WISE WARM UP IS MORE STABLE

In contrast, directly training hidden space edits from scratch tends to perturb token representations and attention interactions earlier, which hurts both direct answering behavior on InternVQA and generalization retention on public benchmarks (Li et al., 2025b; Ni et al., 2025). To further verify this point, we compare early training stability between hidden wise and logits wise warm up. Tab. 15 reports the average gradient L2 norm, gradient standard deviation, and loss variance over the first 1000 training steps. Logits wise warm up produces smaller gradients and lower variance, supporting that logits space optimization is more stable at the beginning of training.

Table 15: Early stage stability of Stage I warm up.

Warm up	**Grad L2**	**Grad std**	**Loss var.**
hidden wise	1.64	0.71	0.226
logits wise	**1.02**	**0.49**	**0.171**

D WHY LOGITS-WISE CALIBRATION IS MORE STABLE

In Stage I, the new module produces a vocabulary-level correction $\Delta\boldsymbol{\ell}(\mathcal{W}_\ell)$, shared across the supervised answer positions. We derive the gradient for one such position, suppressing its index. During autoregressive decoding, this is the last position predicting the next token; under teacher forcing, the loss sums or averages these contributions over supervised answer positions. For sample b, denote the logits at this position as

$$\boldsymbol{\ell}^{(b)} \;=\; \boldsymbol{\ell}_0^{(b)} \;+\; \Delta\boldsymbol{\ell}^{(b)}, \qquad \Delta\boldsymbol{\ell}^{(b)} = \mathcal{W}_\ell \mathbf{u}^{(b)}, \tag{9}$$

where $\mathbf{u}^{(b)} \in \mathbb{R}^{2r}$ is the pooled feature $\mathbf{u}^{(b)} = [\mathrm{pad}(\mathbf{u}^{(b)}_{\mathrm{img}}); \mathrm{pad}(\mathbf{u}^{(b)}_{\mathrm{txt}})]$ constructed by the online subspace pooling described in Stage I, and $\mathcal{W}_\ell \in \mathbb{R}^{V\times 2r}$.

Let $\mathbf{p}^{(b)} = \mathrm{softmax}(\boldsymbol{\ell}^{(b)}) \in \mathbb{R}^V$ and $\mathbf{e}^{(b)} \in \{0,1\}^V$ be the one-hot label vector. The token-level cross-entropy at this position is

$$\mathcal{L}^{(b)}_{\mathrm{ce}} = -\sum_{v=1}^{V} \mathbf{e}^{(b)}_v \log \mathbf{p}^{(b)}_v, \qquad \mathcal{L}_{\mathrm{ce}} = \sum_{b=1}^{B} \mathcal{L}^{(b)}_{\mathrm{ce}}. \tag{10}$$

Step 1: derivative w.r.t. logits. Using the standard softmax-cross-entropy derivative,

$$\frac{\partial \mathcal{L}^{(b)}_{\mathrm{ce}}}{\partial \boldsymbol{\ell}^{(b)}} = \mathbf{p}^{(b)} - \mathbf{e}^{(b)} \;\triangleq\; \mathbf{g}^{(b)}_\ell \in \mathbb{R}^V. \tag{11}$$

Step 2: derivative of $\Delta\boldsymbol{\ell}$ w.r.t. $\mathcal{W}_\ell$. From Eq. (9), the v-th component is $\Delta\ell^{(b)}_v = \sum_{j=1}^{2r} (\mathcal{W}_\ell)_{v,j}\, u^{(b)}_j$. Thus, element-wise we have

$$\frac{\partial \Delta\ell^{(b)}_v}{\partial (\mathcal{W}_\ell)_{a,c}} = \mathbb{I}[v=a]\; u^{(b)}_c, \tag{12}$$

where $\mathbb{I}[\cdot]$ is the indicator function. Equivalently, the Jacobian satisfies

$$\frac{\partial \Delta\boldsymbol{\ell}^{(b)}}{\partial \mathcal{W}_\ell} \;\Longrightarrow\; \nabla_{\mathcal{W}_\ell} \Delta\boldsymbol{\ell}^{(b)} = \Big[\frac{\partial \Delta\ell^{(b)}_v}{\partial (\mathcal{W}_\ell)_{a,c}}\Big]_{(a,c)} = \text{(broadcast of } \mathbf{u}^{(b)} \text{ over the row index)}. \tag{13}$$

Step 3: chain rule to obtain $\nabla_{\mathcal{W}_\ell}\mathcal{L}_{\mathrm{ce}}$. Because $\boldsymbol{\ell}^{(b)}$ depends on $\mathcal{W}_\ell$ only through $\Delta\boldsymbol{\ell}^{(b)}$, we have

$$\frac{\partial \mathcal{L}^{(b)}_{\mathrm{ce}}}{\partial \mathcal{W}_\ell} = \Big(\frac{\partial \mathcal{L}^{(b)}_{\mathrm{ce}}}{\partial \boldsymbol{\ell}^{(b)}}\Big)^\top \frac{\partial \boldsymbol{\ell}^{(b)}}{\partial \mathcal{W}_\ell} = \big(\mathbf{g}^{(b)}_\ell\big)^\top \frac{\partial \Delta\boldsymbol{\ell}^{(b)}}{\partial \mathcal{W}_\ell}. \tag{14}$$

Plugging Eq. (12) into Eq. (14), the gradient admits a simple outer-product form:

$$\frac{\partial \mathcal{L}^{(b)}_{\mathrm{ce}}}{\partial \mathcal{W}_\ell} = \big(\mathbf{p}^{(b)} - \mathbf{e}^{(b)}\big)\big(\mathbf{u}^{(b)}\big)^\top \in \mathbb{R}^{V\times 2r}. \tag{15}$$

Summing over the batch yields

$$\frac{\partial \mathcal{L}_{\mathrm{ce}}}{\partial \mathcal{W}_\ell} = \sum_{b=1}^{B} \big(\mathbf{p}^{(b)} - \mathbf{e}^{(b)}\big)\big(\mathbf{u}^{(b)}\big)^\top. \tag{16}$$

Matrix form (compact). Let $\mathbf{p}, \mathbf{e} \in \mathbb{R}^{B\times V}$ stack the batch distributions, and $\mathbf{U} \in \mathbb{R}^{B\times 2r}$ stack the pooled features. Then Eq. (16) can be written as

$$\frac{\partial \mathcal{L}_{\mathrm{ce}}}{\partial \mathcal{W}_\ell} = (\mathbf{p} - \mathbf{e})^\top \mathbf{U}, \tag{17}$$

which is the expanded form of Eq. (3) under $\Delta\boldsymbol{\ell} = \mathcal{W}_\ell \mathbf{u}$.

E More Metrics and Human Evaluation

E.1 More Metrics Calculation

Computation of Public Benchmark Retention: We evaluate generalization retention on ten public VQA benchmarks across four base models: LLaVA-OneVision-0.5B, LLaVA-OneVision-7B, Qwen3-VL-2B, and Qwen3-VL-8B. For a method m on backbone b, we first compute the summed public benchmark score as

$$\mathrm{RetainScore}(m,b) = \sum_{j=1}^{10} s_j(m,b), \tag{18}$$

Table 16: Token F1 and num_acc metrics experiments.

Method		LLaVA-OneVision-0.5B		LLaVA-OneVision-7B		Qwen3-VL-2B-Instruct		Qwen3-VL-8B-Instruct	
		Token F1	num_acc	Token F1	num_acc	Token F1	num_acc	Token F1	num_acc
PromptCap	w/o Desc	35.80	34.00	46.42	45.54	45.97	44.45	52.50	52.34
	WithDesc	46.57	44.76	54.53	53.76	54.30	52.83	62.32	61.40
	$S_{\texttt{InternVQA}}$	69.92	68.35	82.53	81.95	81.88	81.15	81.30	82.69
QD-Cap	w/o Desc	36.82	35.00	47.83	47.30	47.46	46.53	54.58	53.94
	WithDesc	45.99	44.17	55.44	54.23	52.93	52.09	62.76	62.06
	$S_{\texttt{InternVQA}}$	75.10	73.80	84.09	85.35	88.47	88.05	85.01	84.95
PAI	w/o Desc	43.41	41.59	50.81	50.68	46.42	46.02	56.97	56.32
	WithDesc	49.03	47.29	56.26	54.70	52.75	51.36	63.29	62.69
	$S_{\texttt{InternVQA}}$	87.05	86.29	89.27	92.07	86.36	88.40	88.91	88.69
OPERA	w/o Desc	42.47	40.64	49.29	48.77	47.77	47.02	54.18	54.18
	WithDesc	48.49	46.73	54.91	53.88	52.69	51.83	64.74	63.66
	$S_{\texttt{InternVQA}}$	85.83	85.01	88.60	89.52	89.70	89.77	80.51	82.50
NaPO	w/o Desc	44.40	42.61	49.31	48.59	50.53	50.13	54.91	54.15
	WithDesc	50.27	48.57	56.90	56.00	56.10	55.19	61.52	60.85
	$S_{\texttt{InternVQA}}$	86.78	86.01	84.61	84.75	88.98	89.91	87.96	87.63
Visual-RFT	w/o Desc	42.85	41.03	51.79	51.73	51.72	50.96	58.14	58.52
	WithDesc	48.45	46.69	57.36	56.11	56.91	55.58	63.04	62.57
	$S_{\texttt{InternVQA}}$	86.93	86.21	89.25	91.53	89.97	90.93	91.57	93.08
Fast Quiet-STaR	w/o Desc	43.58	41.53	51.85	51.56	36.64	62.28	40.45	63.71
	WithDesc	48.36	46.29	56.66	55.74	37.93	64.77	38.99	62.67
	$S_{\texttt{InternVQA}}$	89.03	88.55	90.73	91.89	96.49	96.00	103.61	101.64
COCONUT	w/o Desc	43.49	41.45	51.90	51.32	36.38	62.21	41.40	64.04
	WithDesc	48.26	46.05	56.71	55.48	37.70	64.68	39.03	62.61
	$S_{\texttt{InternVQA}}$	89.02	88.91	90.73	91.88	96.38	96.03	105.71	102.23
VTool-R1	w/o Desc	43.67	41.41	52.05	51.54	36.68	62.12	41.23	64.34
	WithDesc	48.34	46.02	56.89	55.76	38.00	64.53	39.16	62.69
	$S_{\texttt{InternVQA}}$	89.30	88.86	90.70	91.80	96.38	96.11	105.03	102.57
Grit	w/o Desc	43.61	41.46	51.94	51.45	36.76	62.13	40.33	64.27
	WithDesc	48.32	46.30	56.87	55.69	37.96	64.52	39.25	62.83
	$S_{\texttt{InternVQA}}$	89.22	88.31	90.51	91.77	96.74	96.16	102.69	102.25
Ours	w/o Desc	45.87	44.10	54.51	54.35	39.34	65.46	47.00	68.35
	WithDesc	50.28	48.58	59.01	58.15	40.22	67.30	42.38	66.01
	$S_{\texttt{InternVQA}}$	90.39	89.84	91.74	93.01	97.76	97.19	109.83	103.42

where $s_j(m, b)$ denotes the score of method m on the j-th public benchmark under backbone b. We then compute the retained percentage relative to the corresponding base model:

$$\mathrm{RetainPct}(m, b) = \frac{\mathrm{RetainScore}(m, b)}{\mathrm{RetainScore}(\mathrm{Base}, b)} \times 100\%. \tag{19}$$

Finally, we average these values over the four backbones:

$$\overline{\mathrm{RetainScore}}(m) = \frac{1}{4} \sum_{b=1}^{4} \mathrm{RetainScore}(m, b), \tag{20}$$

$$\overline{\mathrm{RetainPct}}(m) = \frac{1}{4} \sum_{b=1}^{4} \mathrm{RetainPct}(m, b). \tag{21}$$

Thus, **RetainScore** measures the average summed performance on public VQA benchmarks, while **RetainPct** measures how much performance is retained relative to each corresponding base model.

Task-level retention. RetainPct summarizes a sum of task scores, so improvements on one task can offset regressions on another. We additionally compute the worst per-task drop, $\max_j [s_j(\mathrm{Base}, b) - s_j(m, b)]_+$, in score points. From Tab. 6, the largest drops for `Internalization` are 5.23 on ScienceQA (LLaVA-OneVision-0.5B), 8.53 on ScienceQA (LLaVA-OneVision-7B), 20.48 on ChartQA (Qwen3-VL-2B), and 6.68 on ChartQA (Qwen3-VL-8B). The 96.45% average RetainPct describes aggregate performance, while these task-level differences characterize where capabilities change.

Besides the judge-based InternVQA score in the main paper, we additionally report two lightweight token-level metrics: Token F1 and num_acc. These metrics are computed from the predicted answer $\hat{y}$ and the reference answer y.

Token F1. We tokenize y and $\hat{y}$ into token multisets $T(y)$ and $T(\hat{y})$. Let $c(t, S)$ be the multiplicity of token t in multiset S. The number of matched tokens is

$$M = \sum_{t \in \mathcal{V}} \min\big(c(t, T(y)),\ c(t, T(\hat{y}))\big), \tag{22}$$

where $\mathcal{V}$ is the tokenizer vocabulary. Token precision and recall are

$$P = \frac{M}{|T(\hat{y})| + \epsilon}, \quad R = \frac{M}{|T(y)| + \epsilon}, \tag{23}$$

and Token F1 is

$$\text{TokenF1} = \frac{2PR}{P + R + \epsilon}. \tag{24}$$

We set $\epsilon = 10^{-8}$ to avoid division by zero.

num_acc. We extract all numeric entities from y and $\hat{y}$, denoted as $N(y)$ and $N(\hat{y})$. We define an exact-match accuracy on the numeric set:

$$\text{num_acc} = \mathbb{I}\big(N(\hat{y}) = N(y)\big), \tag{25}$$

where $\mathbb{I}(\cdot)$ is the indicator function. We average num_acc over the evaluation set. Under this definition, two empty numeric sets also match, so the metric measures numeric-set agreement across all questions rather than accuracy restricted to numerical questions.

InternVQA score. The InternVQA score $S_{\texttt{InternVQA}}$ is computed in the same way as in the main paper. In short, each sample is evaluated under two inference settings: *direct answering* (`w/o Desc`) and *caption-then-answering* (`WithDesc`). A judge model returns a score in $[0, 1]$ for each setting, and $S_{\texttt{InternVQA}}$ aggregates the relative gain from captions versus direct answering as defined in the main paper.

Tab. 16 shows that captioning generally improves Token F1 and num_acc for most baselines, which is consistent with the observation that explicit descriptions provide extra cues. These reference-based metrics complement the judge scores: Token F1 measures lexical agreement, while num_acc checks numeric agreement. They target different aspects of answer quality. On Qwen3-VL-2B, our Direct Token F1 is 39.34, below PromptCap's 45.97 and Visual-RFT's 51.72, while our num_acc is 65.46, above 44.45 and 50.96, respectively. The judge-based improvement therefore does not imply an improvement in every reference-based metric.

First, Token F1 mainly measures lexical overlap between the prediction and the reference. This makes it sensitive to wording rather than semantic correctness. For long-form answers, a verbose response can repeat many reference tokens and obtain a high Token F1, even if the key conclusion is wrong. For example, on a question that asks *which wrap has the highest calories and its macros*, a model may mention many food names, calorie values, and nutrient terms from the infographic. This can increase token overlap, while the final selected item or the matched macro values are still incorrect. In fine-grained VQA, this issue becomes more severe because the answer often hinges on a small visual detail, such as a color, count, position, or attribute binding.

Second, num_acc only checks exact matching of extracted numeric entities. This metric is useful for simple counting or OCR-style questions, but it is brittle when the answer involves rounding, unit conversion, partial numeric evidence, or different but equivalent formats. For example, an answer may round a value, convert a unit, or report the requested subset of numbers correctly, but still be penalized by set-level exact matching. Conversely, a response may include the correct number somewhere in a long explanation while using it for the wrong object, which num_acc cannot reliably detect.

These limitations explain why lightweight token-level metrics do not fully capture VQA quality in InternVQA. Our benchmark includes short answers, long descriptions, OCR, counting, spatial reasoning, schema parsing, and fine-grained attribute questions, where correctness depends on visual grounding and answer intent. The LLM judge evaluates the final answer using the question and image, providing a semantic assessment alongside these reference-based checks. Therefore, we report Token F1 and num_acc only as auxiliary signals, while using judge-based InternVQA and human evaluation as the primary evidence for our conclusions.

E.2 Single-Blind Human Evaluation

To validate that the judge-based ranking aligns with human preferences, we conduct a single-blind human study with 30 volunteers. Each volunteer randomly evaluates 100 questions (3000 question-level ratings in total).

Protocol. For each question, we show the question and the image, along with the outputs from all methods. The method names are removed and the method order is randomly permuted per question. Volunteers do not know which method produced which answer.

Scoring rubric. Each answer receives a score in $\{0, \ldots, 5\}$ by summing three components:

- **Correctness (0–3):** 3 for fully correct, 2 for mostly correct with minor omissions, 1 for partially correct, 0 for wrong or irrelevant.
- **Faithfulness (0–1):** 1 if the answer is grounded in visible evidence and does not hallucinate, otherwise 0.
- **Directness (0–1):** 1 if it answers the question directly without unnecessary description, otherwise 0.

If the question asks "How many sticky notes are on the table and what are their colors?" an answer that directly states "four sticky notes: two white, one pale pink, one light blue" receives high directness and correctness. A long paragraph that describes people studying, mentions highlighters, and then guesses the colors may lose directness and correctness even if some tokens overlap.

Tab. 17 reports the total score over 100 questions (maximum 500) as mean $\pm$ std across 30 volunteers, along with the variance across volunteers. We rank methods by their four-backbone mean $S_{\text{InternVQA}}$ in Tab. 1 and independently by their mean human total score in Tab. 17. The rank correlation describes agreement between these two mean-score orderings. For the 12 method ranks in Tab. 17, Spearman's rank correlation is $\rho = 1 - 6\sum_{j=1}^{12}(r_j - h_j)^2/[12(12^2-1)] = 0.944$, where r_j and h_j are the ranks of automatic and human mean scores, respectively. This high correlation supports the automatic evaluator at the level assessed by the study: overall method ordering. The leading methods form a closely scored group. Our method achieves the highest mean total score ($442.3/500$), with GRIT (441.8) and VTool-R1 (440.6) showing comparable top-tier quality relative to the reported across-rater variation. Correctness and Faithfulness contribute four of the five available points per question, while Directness contributes one. The study therefore assesses answer quality together with directness; the table reports the full across-rater variation alongside the mean ranking.

Table 17: Single-blind human evaluation with 30 volunteers (100 questions each). Total score is in $[0, 500]$. Higher is better.

Method	Rank of mean S	Total score (mean $\pm$ std)	Var. across volunteers	Rank of mean score
BaseModel	12	312.4 ± 48.2	2321.7	12
PromptCap	11	386.6 ± 35.8	1281.6	11
QD-Cap	10	411.0 ± 31.5	992.3	10
PAI	7	419.8 ± 28.9	835.4	9
OPERA	9	425.3 ± 26.7	713.9	8
NaPO	8	430.1 ± 24.5	600.8	7
Fast Quiet-STaR	6	435.7 ± 22.1	488.6	6
Visual-RFT	3	437.4 ± 20.6	425.3	5
COCONUT	5	438.9 ± 19.7	398.7	4
VTool-R1	2	440.6 ± 19.1	367.4	3
GRIT	4	441.8 ± 18.8	352.1	2
Ours	**1**	$\mathbf{442.3 \pm 18.5}$	**342.3**	**1**

F More Efficiency Experiments

F.1 New Modules Efficiency

The 23.86% end-to-end saving in Tab. 4 mixes two effects that we separate here: the cost of the new modules in the forward pass, and the tokens no longer generated because the description is gone. Replacing explicit description generation with compact prior tokens creates a natural tradeoff: we save the cost of generating long intermediate descriptions, but introduce a lightweight prior module before answering. The key question is therefore whether the module itself slows decoding.

Throughput under a shared token budget. To isolate the runtime cost of the new modules, we compare the base direct-answering model and our trained model under the same hardware, image prompt, and decoding configuration, with the same `max_new_tokens` setting. We gradually increase `max_new_tokens` to 500 and measure end-to-end throughput (Tab. 18). The base model reaches 33.79 tokens/s and `Internalization` reaches 34.30 tokens/s, a ratio of $34.30/33.79 =$

1.015, so the prior module causes no systematic decoding slowdown; the PCA selector adds 1.9% on long contexts (Tab. 21). The 23.86% end-to-end reduction in Tab. 4 is therefore a token-count effect: the trained model emits fewer tokens for the same question, not a faster forward pass.

Table 18: **Decoding throughput under a shared token budget.** Base direct answering and `Internalization` under the same hardware, prompt, and decoding configuration, with `max_new_tokens` raised to 500 (higher is faster). The token-normalized throughput complements the end-to-end timings in Tab. 4.

Model	Throughput (tokens/s)	Ratio to base
Base model (direct answering)	33.79	1.000
`Internalization`	34.30	1.015

Baselines relative to caption-then-answer. Tab. 19 reports the end-to-end speedup of each baseline over a caption-then-answer pipeline on the same backbone. Caption-based methods still generate intermediate descriptions, so their speedups remain limited. Tool-based and latent reasoning methods avoid some description tokens, but usually introduce extra visual operations, tool calls, or latent computation. In contrast, `Internalization` keeps the inference path close to direct answering while removing explicit descriptions, which is why its end-to-end time in Tab. 4 falls below the direct-answering base model rather than only below caption-then-answer. This makes compact prior tokens a practical replacement for description generation, especially when combined with stronger generalization retention.

Table 19: **End-to-end inference speedup of the baselines over caption-then-answer.** We report the speedup of each baseline relative to a caption-then-answer inference pipeline under the same backbone and decoding configuration (higher is faster). Our method is compared against the direct-answering base model in Tab. 4 and Tab. 18.

Method	Speedup
PromptCap-2023	1.062
QD-Cap-2024	1.088
PAI-2024	1.071
OPERA-2024	1.049
GRIT	1.112
VTool-R1	1.135
Fast Quiet-STaR	1.167
COCONUT	1.154
NaPO-2025	1.214
Visual-RFT-2025	1.183

F.2 Training Memory and Cost

Table 20: **Relative training VRAM usage and optimization time.** Both columns use Internalization Stage I as the 100% reference under the same hardware setup. Optimization time adds across the two sequential stages; VRAM is reported per stage. Supervision generation and filtering are outside the reported optimization time.

Method	Avg. VRAM (%)	ZeRO-3 Time (%)
NaPO	124.8	165.3
Visual-RFT	127.6	172.1
Fast Quiet-STaR	108.9	119.5
COCONUT	107.3	115.8
VTool-R1	118.7	141.2
GRIT	114.5	133.9
Ours (Stage I)	100.0	100.0
Ours (Stage II)	102.42	108.56
Ours (Stage I + II)	—	**208.56**

Tab. 20 separates training memory from cumulative optimization time. Stage II uses 102.42% of Stage I's average VRAM, and each stage uses less memory than the listed training baselines. The two sequential stages take $100.00 + 108.56 = 208.56\%$ of the Stage I reference time; their VRAM footprints are not added. Prompt-only methods require no post-training optimization. Our two-stage

Table 21: **Runtime efficiency of PCA-based pooling on long-context samples.** We evaluate 1K long-context samples with an average context length of 11.3K tokens across four backbones. The overhead is measured relative to standard decoding without PCA-based pooling.

Setting	Avg. context length	#Samples	Extra time
Standard decoding	11.3K	1K	0.0%
PCA-based pooling	11.3K	1K	1.9%

investment targets deployment efficiency: at inference the Stage I teacher and logits-correction branch are removed, and shorter outputs reduce end-to-end latency by 23.86% on average (Tab. 4).

F.3 PCA Efficiency

A natural concern is whether the online PCA-based pooling in Stage I introduces noticeable overhead, especially for long-text inputs or high-resolution images with many visual tokens. To examine this, we conduct a runtime experiment on 1K long-context samples across the four main backbones. The average context length is 11.3K tokens, covering both long textual contexts and large visual-token inputs.

Following our Stage I design, we collect masked image and question tokens from the final decoder hidden states, compute a compact PCA basis online, and use the projected token energy for projected-energy token selection and pooling. This procedure introduces no extra trainable parameters or external memory, and only operates on the selected image and question token spaces rather than the full generation trajectory. As reported in Tab. 21, the PCA-based pooling introduces only 1.9% additional inference time on average over standard decoding. This result shows that the PCA module remains lightweight even when the context becomes long or the image contains many visual tokens.

The small overhead comes from three factors. First, PCA is used only for online token-space construction, where the masked image and question tokens are compressed into a compact basis for token scoring and pooling. Second, we use a truncated rank budget, so the practical computation scales with the selected token count and the rank rather than requiring a full decomposition of all hidden states. Third, PCA is applied before answer generation and does not introduce any extra autoregressive decoding step. Therefore, even in long-text or large-image settings, the cost of PCA is quickly dominated by the normal VLM forward pass and decoding cost. This makes PCA-based pooling an efficient choice for constructing the implicit prior.

In Stage II, we continue preference learning on top of the Stage I model, but freeze the Stage I-specific logits projection branch to preserve the generalization learned in Stage I. This branch serves purely as a training-time teacher that provides stable logit supervision, and we remove it at inference, introducing no extra runtime overhead. By reusing the frozen Stage I teacher instead of a separate reference model, we avoid the substantial memory cost typically required by preference optimization. In practice, this design replaces a full reference model with only a lightweight Stage I-only module (a $V \times 2r$ logits head, with $2rV$ weights; storage depends on numerical precision), while still effectively constraining style drift and maintaining base-model generalization during preference learning.

Optimization design is assessed through the matched logits-wise versus hidden-wise warm-up comparison (Tab. 7) and the component ablations (Appendix C), using the same downstream evaluation quantities.

G Matrix Decomposition Methods Evaluations

This appendix details the *subspace extractor* used in Stage I token-space construction. Specifically, it corresponds to the computation of the modality basis $\mathcal{B}_m$ in the *Online token-space construction* paragraph in Sec. 3, where we form a compact orthonormal basis from masked token hidden states and use it for invariant token scoring, and pass projected pooled coordinates to the trainable logits head in Eq. (2). We compare four basis constructions (PCA, SVD, ICA, QR) within this complete Stage I pipeline.

G.1 Setup and Notation

For each modality $m \in \{\text{img}, \text{txt}\}$ in a single forward pass, we collect the masked token hidden states into a matrix

$$X \in \mathbb{R}^{N \times H}, \tag{26}$$

where N is the number of tokens selected by the modality mask and H is the hidden dimension. We center the tokens to remove the mean direction:

$$\mu = \frac{1}{N} \sum_{i=1}^{N} X_{i,:} \in \mathbb{R}^H, \qquad \bar{X} = X - \mathbf{1}\mu^\top \in \mathbb{R}^{N \times H}. \tag{27}$$

Each method outputs a basis $B \in \mathbb{R}^{H \times r}$ under the same rank budget, used for token scoring and the coordinate-dependent logits correction.

Computational scope. PCA via covariance eigendecomposition and SVD of centered tokens recover the same principal subspace in exact arithmetic. Their numerical cost depends on matrix dimensions and solver choice. Our measured PCA overhead is reported in Tab. 21; ICA and QR serve as alternative basis constructions.

G.2 PCA: Principal Component Analysis

Role and characteristics. PCA is a second-order method that identifies orthogonal directions capturing the largest variance of the centered token distribution. It provides a ranked set of directions (principal components) and is widely used for efficient low-rank compression. In our context, PCA serves as the default subspace extractor and preserves maximal energy under a linear subspace.

Decomposition procedure. PCA is commonly computed from the sample covariance matrix of centered tokens:

$$C = \frac{1}{N} \bar{X}^\top \bar{X} \in \mathbb{R}^{H \times H}. \tag{28}$$

Then we perform eigendecomposition:

$$C = V \Lambda V^\top, \tag{29}$$

where $V = [v_1, \ldots, v_H]$ contains orthonormal eigenvectors and $\Lambda = \text{diag}(\lambda_1, \ldots, \lambda_H)$ with

$$\lambda_1 \geq \lambda_2 \geq \cdots \geq \lambda_H \geq 0. \tag{30}$$

The top-r principal directions define the rank-r basis:

$$B_{\text{PCA}} = V_{:,1:r} = [v_1, \ldots, v_r]. \tag{31}$$

Practical notes. When N is small or H is large, C is low-rank. We set $r \leq \min(N, H)$ and use truncated eigensolvers in practice.

G.3 SVD: Singular Value Decomposition

Role and characteristics. SVD decomposes $\bar{X}$ directly and is often numerically stable in finite precision. For centered data, SVD yields the same principal subspace as PCA, but avoids explicitly forming $\bar{X}^\top \bar{X}$.

Decomposition procedure. Compute the thin SVD of the centered matrix:

$$\bar{X} = U \Sigma V^\top, \tag{32}$$

where $U \in \mathbb{R}^{N \times p}$ and $V \in \mathbb{R}^{H \times p}$ have orthonormal columns, $\Sigma = \text{diag}(\sigma_1, \ldots, \sigma_p)$, $\sigma_1 \geq \cdots \geq \sigma_p \geq 0$, and $p = \min(N, H)$; zero singular values are included when $\bar{X}$ is rank-deficient. The top-r right singular vectors define a basis:

$$B_{\text{SVD}} = V_{:,1:r}. \tag{33}$$

Connection to PCA.

$$\bar{X}^\top \bar{X} = V \Sigma^2 V^\top, \tag{34}$$

so the eigenvectors of $C = \frac{1}{N} \bar{X}^\top \bar{X}$ are V and $\lambda_j = \sigma_j^2 / N$. With identical preprocessing and the same retained principal subspace, the bases agree up to an orthogonal transformation: $B_{\mathrm{SVD}} = B_{\mathrm{PCA}} R$, $R^\top R = I$. A spectral tie at the truncation boundary requires a consistent subspace choice. For any centered token $\bar{x}_i$, $\|\bar{x}_i^\top B_{\mathrm{PCA}} R\|_2 = \|\bar{x}_i^\top B_{\mathrm{PCA}}\|_2$, so exact token scores and top-k selections agree under the same tie-breaking rule. Eigenvector signs cannot explain different token selections.

The logits head uses coordinates, not only projection norms. Stage I also computes $u_m = B_m^\top h'_m$ and applies a learned head to the concatenated modality coordinates (Eq. (2)). For one modality block, replacing B_m by $B_m R_m$ gives $u'_m = R_m^\top u_m$, so a fixed head W_m produces $W_m R_m^\top u_m$, which need not equal $W_m u_m$. Equality is restored by the matching transformation $W'_m = W_m R_m$. A fixed basis rotation can therefore be absorbed into the head, but a sample-dependent rotation generally cannot be absorbed by one shared linear map. The complete pipeline is thus coordinate-dependent even though its token selection is rotation-invariant.

Fig. 7 reports final trained-model scores. It does not isolate token selection or identify the source of the PCA–SVD difference. The coordinate path above shows why equality of token scores alone is insufficient to establish equivalence of the complete trained models; it is not evidence that basis conventions caused the measured gap.

G.4 ICA: Independent Component Analysis

Role and characteristics. ICA seeks statistically independent components rather than merely uncorrelated directions. It serves as a control to test whether higher-order structure benefits our token-space construction.

Decomposition procedure (standard pipeline).

1. **Centering.** Compute $\bar{X}$ as above.
2. **Whitening.** Compute $C = \frac{1}{N} \bar{X}^\top \bar{X}$ and $C = V \Lambda V^\top$. Then
$$Z = \bar{X}\, V \Lambda^{-1/2}, \qquad \frac{1}{N} Z^\top Z = I. \tag{35}$$
3. **Iterative rotation.** Solve for W such that
$$Y = Z W^\top \tag{36}$$
has maximally independent components (e.g., FastICA).
4. **Directions in the original space.**
$$Y = \bar{X} \underbrace{\left(V \Lambda^{-1/2} W^\top \right)}_{\triangleq\, B_{\mathrm{ICA}}}, \qquad B_{\mathrm{ICA}} \in \mathbb{R}^{H \times r}. \tag{37}$$

Practical notes. ICA is more expensive due to iterative optimization and is sensitive to solver settings. We treat it as an analysis control rather than an efficiency baseline.

G.5 QR Decomposition

Role and characteristics. QR provides an orthonormal basis via orthogonalization without explicitly ranking directions by variance. It is a control to test whether variance ordering (PCA or SVD) is necessary.

Decomposition procedure. A common choice is to factorize $\bar{X}^\top$:

$$\bar{X}^\top = QR, \tag{38}$$

where Q is orthonormal and R is upper triangular. A rank-r basis is

$$B_{\mathrm{QR}} = Q_{:,1:r}. \tag{39}$$

Pivoted QR.

$$\bar{X}^\top \Pi = QR, \tag{40}$$

where Π is a permutation matrix. We still take $B_{\mathrm{QR}} = Q_{:,1:r}$.

G.6 Shared Bottleneck Dimension Experiments

Table 22: Stage I/II shared bottleneck dimension experiment. We vary the dimension d_z across four backbones.

d_z	Qwen3-VL-2B				Qwen3-VL-8B			
	w/o desc↑	w/ desc↑	$S_{\texttt{InternVQA}}$↑	RetainPct↑	w/o desc↑	w/ desc↑	$S_{\texttt{InternVQA}}$↑	RetainPct↑
$d_z = 32$	54.65	61.25	87.92	98.16	66.74	70.21	94.8	96.28
$d_z = 64$	57.62	63.09	90.51	98.03	68.23	71.82	94.74	96.24
$d_z = 128$	59.07	64.73	90.42	97.92	70.87	75.26	93.81	95.69
$d_z = 256$	62.82	67.00	93.35	97.63	74.53	77.88	95.51	95.60
$d_z = 512$	63.27	67.66	93.06	94.53	74.80	77.69	96.14	92.52
d_z	LLaVA-OneVision-0.5B				LLaVA-OneVision-7B			
	w/o desc↑	w/ desc↑	$S_{\texttt{InternVQA}}$↑	RetainPct↑	w/o desc↑	w/ desc↑	$S_{\texttt{InternVQA}}$↑	RetainPct↑
$d_z = 32$	43.51	50.72	83.43	97.35	59.86	64.84	91.68	96.17
$d_z = 64$	44.82	51.62	84.83	97.13	61.85	66.92	91.8	96.09
$d_z = 128$	46.61	51.70	89.08	96.98	62.36	66.62	93.17	95.72
$d_z = 256$	50.14	54.55	91.20	96.98	64.93	68.92	93.85	95.59
$d_z = 512$	50.03	54.81	90.45	92.87	65.24	69.03	94.19	93.71

We study how the shared bottleneck dimension d_z affects internalization and generalization in Tab.22. We observe a clear overall trend that increasing d_z steadily improves w/o desc and often $S_{\texttt{InternVQA}}$, showing that a wider bottleneck provides more capacity to encode the fused semantic signal into the implicit prior. However, the marginal gains shrink after a moderate range, and overly large d_z can noticeably reduce RetainPct, suggesting that excess capacity gives preference learning more freedom and makes style drift harder to control. Overall, $d_z = 256$ offers the best trade-off, achieving near-peak internalization while keeping RetainPct high and stable across backbones.

G.7 Stage I Subspace Decomposition

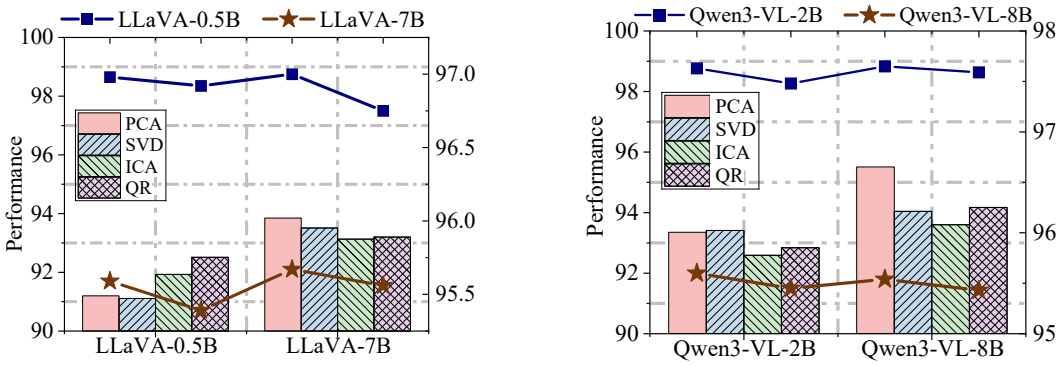


Figure 7: **Stage I sensitivity to subspace decomposition.** We vary the basis construction (PCA, SVD, ICA, QR) used for both token scoring and the Stage I logits head. Bars report $S_{\texttt{InternVQA}}$ (higher is better), while lines report RetainPct on public benchmarks (higher is better).

Fig. 7 reports final `InternVQA` scores and RetainPct for the four basis constructions across two LLaVA-OneVision and two Qwen3-VL backbones. The PCA-labelled runs have the highest reported $S_{\texttt{InternVQA}}$ on three backbones, while QR and ICA are higher on LLaVA-OneVision-0.5B. RetainPct varies less across these runs. We retain PCA as the default configuration.

These results compare the full pipeline, including the coordinate-dependent logits head. They establish neither a token-selection advantage of PCA over centered SVD nor a unique explanation for the observed gap. With identical retained subspaces and tie-breaking, their projected-energy selectors are mathematically equivalent, as shown above. The bottom-k, random-k, and learnable-pooling controls separately evaluate the default selection and pooling design.

H More Experiment Settings

Tab. 23 summarizes the training and hardware settings of Stage I and Stage II to make the training recipe transparent and reproducible.

Table 23: Training and hardware settings.

Field	Value	Field	Value
CPU cores used	4	Framework	PyTorch
GPU platform	NVIDIA A800	Distributed training	Yes
CPU record	Intel Xeon Platinum 8358P @ 2.60 GHz	Distributed method	Accelerate + Transformers; ZeRO-3 in Stage II
Number of GPUs	64	Gradient checkpointing	Enabled in both stages
GPU memory per unit	80 GB	Experimental protocol	Results averaged over 3 independent runs
System RAM	512 GB	Training set size	280,000
Storage type	SATA-SSD	Stage I epochs	2
Infrastructure	On-premises	Effective batch size	16
Precision	BF16	Stage II learning rate	5×10^{-6}
Stage I setup	280K VQA pairs; 2 epochs; effective batch size 16; LR 3×10^{-5}	Test instances	4,941

Stage I performs supervised training on GPUs with bf16 and gradient checkpointing, using an effective batch size of 16 and a learning rate of 3×10^{-5}. Stage II adopts preference training with ZeRO-3 on 64 GPUs while keeping the same effective batch size, and uses a smaller learning rate of 5×10^{-6} to stabilize optimization. The pairwise logistic term uses $\beta = 0.1$.

Baseline methods Overview Tab. 24 summarizes the baseline methods used in our experiments. We distinguish caption generation, preference or reinforcement post-training, decoding interventions, visual-tool reasoning, and latent reasoning. Caption methods supply textual evidence, post-training changes model parameters, and decoding interventions adjust generation without extra training. Visual-tool methods introduce additional image operations. Latent reasoning methods move reasoning into hidden or continuous states, reducing visible reasoning tokens but still requiring additional internal computation. In contrast, `Internalization` targets description reliance directly: it trains compact image-question prior tokens using caption-derived answer supervision and performs direct answering without intermediate descriptions or visual tools.

Table 24: Baseline overview.

Direction	Methods	Main Mechanism
Caption-based	PromptCap (Hu et al., 2023), QD-Cap (Özdemir & Akagündüz, 2024)	Generate question-conditioned textual evidence
Post-training	NaPO (Zhang et al., 2025e), Visual-RFT (Liu et al., 2025c)	Preference optimization or reinforcement fine-tuning
Decoding intervention	PAI (Liu et al., 2024), OPERA (Huang et al., 2024)	Adjust visual attention or decoding decisions
Visual-tool reasoning	VTool-R1 (Wu et al., 2025a), GRIT (Fan et al., 2025)	Use tools or image operations
Latent reasoning	Fast Quiet-STaR (Huang et al., 2025b), COCONUT (Hao et al., 2024)	Move reasoning into hidden or continuous states
Ours	`Internalization`	Inject compact prior tokens before the LM head

Cross-Entropy and KL Divergence We briefly define two standard objectives used in this paper. Let p denote a target distribution over the vocabulary and q_θ denote the model distribution. The cross-entropy is

$$\mathcal{H}(p, q_\theta) \;=\; -\sum_{v \in \mathcal{V}} p(v) \log q_\theta(v). \tag{41}$$

For one-hot supervision with ground-truth token y, i.e., $p(v) = \mathbf{1}[v = y]$, cross-entropy reduces to the negative log-likelihood

$$\mathcal{L}_{\mathrm{CE}} \;=\; -\log q_\theta(y). \tag{42}$$

The Kullback–Leibler (KL) divergence from p to q_θ is

$$\mathrm{KL}(p\|q_\theta) \;=\; \sum_{v\in\mathcal{V}} p(v)\log\frac{p(v)}{q_\theta(v)}. \tag{43}$$

When using teacher logits $\boldsymbol{\ell}^{\mathrm{T}}$ and student logits $\boldsymbol{\ell}^{\mathrm{S}}$, we form distributions with temperature τ as $p = \mathrm{softmax}(\boldsymbol{\ell}^{\mathrm{T}}/\tau)$ and $q_\theta = \mathrm{softmax}(\boldsymbol{\ell}^{\mathrm{S}}/\tau)$, and apply $\mathrm{KL}(p\|q_\theta)$ accordingly.

Tab. 24 summarizes the baseline categories used in our experiments. Caption-based methods reduce the direct answer gap by generating descriptions, but introduce extra text generation. Visual evidence intervention and Thinking-with-Image methods strengthen visual grounding through inference-time modification or extra visual processing. Latent reasoning methods move reasoning into hidden states, but their hidden trajectories are difficult to supervise. Post-training direct answering methods improve direct answers through behavior alignment. In contrast, `Internalization` injects compact prior tokens before the language model head, avoiding explicit descriptions, extra visual tools, and hidden reasoning chains.

I INTERNVQA BENCHMARK DETAILS

To comprehensively evaluate the capabilities of VLMs, our benchmark is structured around **16 representative categories** spanning three key dimensions: (1) **10 Task Types** ranging from perception to reasoning; (2) **3 Length Type constraints** to test generation adaptability; and (3) **3 Image Types** based on textual density. This section provides detailed definitions and qualitative examples for each category. For ease of understanding, we will take the original Qwen3-VL-2B as an example.

I.1 BENCHMARK COMPOSITION AND EVALUATION PROTOCOL

This subsection summarizes the composition of the 1.5K samples in `InternVQA` and their evaluation protocol. Each sample consists of one image, one question, and three labels (image type, task type, length type) that the judge receives as metadata (Appendix M); the judge receives no reference answer: it first writes a checklist of what a correct answer must contain from the image and the question, then scores both settings against that checklist. The composition is 778 short, 304 medium, and 418 long samples; 38.7% images without text, 21.9% text-only images, and 39.5% mixed image-text inputs; and ten task types, the largest at 25.0% and the rest between 7.3% and 12.3% (Fig. 3).

Task scope. For question-answering tasks, Direct produces the requested answer without a preliminary image description. For Open Caption and Brief Recap, the requested description or summary is itself the answer; the comparison measures the benefit of an additional preliminary description. These categories are reported separately in the task-type columns and should be interpreted as generation tasks rather than factual VQA.

Length type. Samples are bucketed by the token length of the text completion into short $[1, 256]$, medium $[257, 1024]$, and long > 1024 tokens (Fig. 3).

Training and evaluation scope. The 280K training questions and the benchmark do not overlap (Sec. 4). Question generation and InternVQA use the same ten task categories: InternVQA evaluates new questions within this shared task scope, while the public benchmarks provide complementary task-specific evaluations. The category breakdown separates factual QA from requested description and summary generation.

Quality control and scoring. Both settings are scored by GPT-5-mini with the metadata-conditioned rubric of Appendix M. A uniform GPT-based format checker is applied across all methods to ensure that evaluated responses satisfy the required output schema, and the description-trigger rate characterizes models' tendency to produce explicit descriptions. A single-blind human study with 30 volunteers ranks the methods consistently with the automatic judge (Spearman $\rho = 0.944$) under a combined correctness, faithfulness, and directness rubric (Appendix E.2). Two judges from other model families also reproduce the leading method (Sec. 4). All InternVQA scores are means over three runs.

I.2 Task Types (10 Classes)

I.2.1 Open Caption (OC): General Scene Understanding

This category evaluates the model's capability to generate holistic and accurate descriptions of a visual scene without specific constraints. Unlike closed-ended QA, Open Captioning requires the model to self-determine the most salient features of an image and translate them into coherent natural language. The core challenge here is "visual grounding"—ensuring that every generated descriptive phrase corresponds to actual pixel data, thereby minimizing the common issue of "hallucination" where models generate plausible but non-existent details based on language priors rather than visual evidence. **A representative example is shown in Fig.8.**

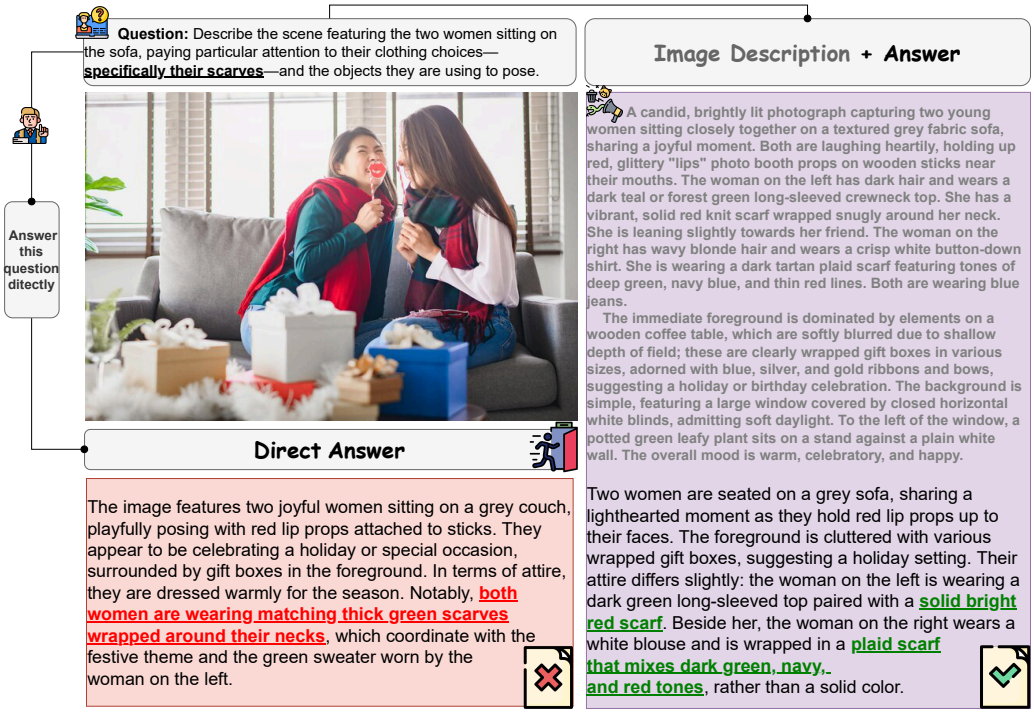


Figure 8: **Sample of Open Caption (OC) Task** The comparison highlights the challenge of fine-grained attribute recognition. The **Direct Answer** (left) fails due to visual hallucination, incorrectly stating that "both women are wearing matching thick green scarves." The **Correct Answer** (right) accurately distinguishes the visual details, identifying the "solid bright red scarf" versus the "plaid scarf."

I.2.2 Part Attribute (PA): Fine-Grained Object Details

Part Attribute tasks assess fine-grained visual perception by focusing on specific properties (such as color, shape, material, texture, or state) of sub-objects within a larger scene. While current models excel at global object recognition, they often struggle with "attribute binding" (e.g., correctly associating a "red" attribute with a "scarf" rather than a "shirt"). This category specifically tests the model's attention mechanism, requiring it to verify minute details—such as tattoos on a specific limb—rather than relying on generalized scene context. **A representative example is shown in Fig.9.**

I.2.3 Action Event (AE): Dynamic Interaction Recognition

This task moves beyond static object detection to require the correct identification of dynamic actions, human-object interactions, and ongoing events. The primary objective is to evaluate whether the model can infer temporal dynamics from a static image. It challenges the model to distinguish between visually similar but functionally distinct activities (e.g., distinguishing "adjusting a mixing



Figure 9: **Sample of Part Attribute (PA) Task** The example tests the detection of tattoos on the player's arms. The **Direct Answer** (left) fails to detect the attribute on one side, incorrectly claiming the left arm is "clean." The **Correct Answer** (right) successfully verifies the presence of tattoos on *both* arms, demonstrating accurate attribute grounding.

console" from "playing a musical keyboard") by analyzing posture, hand placement, and contextual tools, thus testing the depth of semantic visual understanding. **A representative example is shown in Fig.10.**

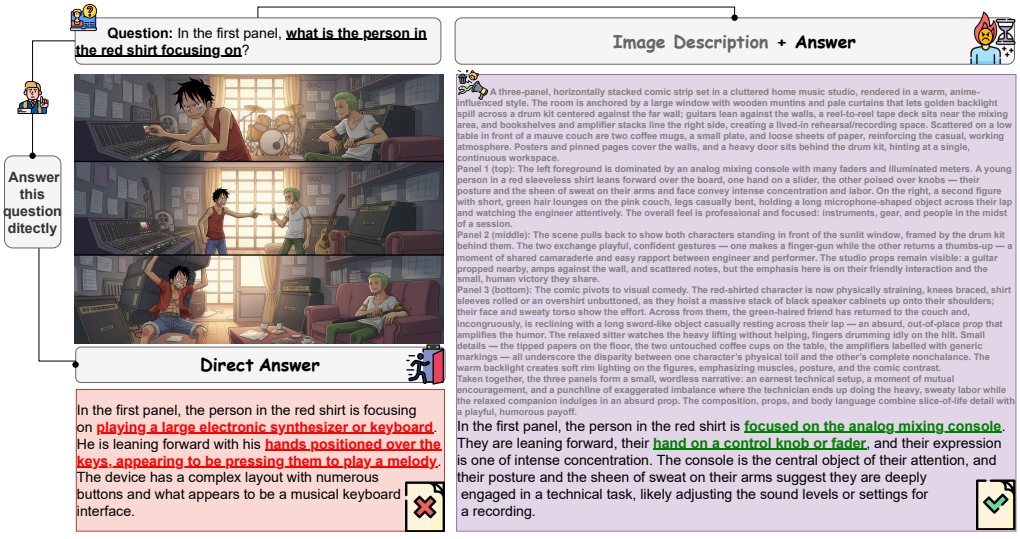


Figure 10: **Sample of Action Event (AE) Task** The model attempts to interpret the character's activity in the comic panel. The **Direct Answer** (left) misinterprets the scene, hallucinating that the user is "playing a melody" on a synthesizer. The **Correct Answer** (right) correctly identifies the action as operating an "analog mixing console" and adjusting "control knobs," showing precise event understanding.

I.2.4 Spatial Layout (SL): Geometric and Positional Reasoning

Spatial Layout tasks test the model's geometric reasoning capabilities and its ability to parse positional relationships between objects in 2D or 3D space. This includes understanding relative directions (left/right, front/back), occlusion relationships, and analyzing character placement across sequential

panels in comic strips. A frequent failure mode in MLLMs is "spatial confusion," where the model correctly identifies objects but misinterprets their spatial configuration. This category serves as a rigorous benchmark for a model's "spatial sense." **A representative example is shown in Fig.11.**

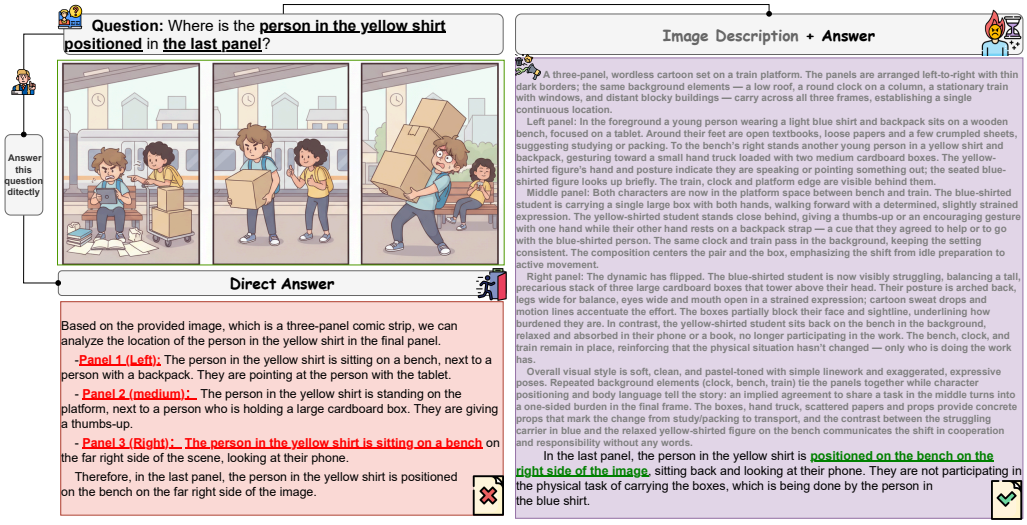


Figure 11: **Sample of Spatial Layout (SL) Task** The question requires identifying the character's position specifically in the *last panel*. The **Direct Answer** (left) fails to adhere to the scope constraint, redundantly describing the entire comic strip sequence (Panels 1-3). The **Correct Answer** (right) demonstrates precise grounding by exclusively focusing on the spatial details of the requested final panel.

I.2.5 Count Measure (CM): Quantitative Analysis and Estimation

This category focuses on quantitative perception, requiring the model to detect and enumerate specific repeated patterns or objects within a scene. Unlike general captioning, counting imposes a strict binary success criterion (correct vs. incorrect number). The challenge lies in maintaining accuracy for small, deformable, or closely spaced features—such as stripes on clothing or items on a cluttered desk—where attention maps often overlap. This tests the model's ability to perform discrete visual reasoning rather than approximate estimation. **A representative example is shown in Fig.12.**

I.2.6 OCR Extract (OE): Text Recognition and Transcription

OCR Extract tasks evaluate the model's Optical Character Recognition (OCR) capabilities alongside semantic text comprehension. Beyond merely transcribing characters from an image (scene text recognition), high-performing models must understand the semantic role of the text within the document structure. This involves extracting specific values (such as dates, times, or prices) and correctly mapping them to their corresponding entities (e.g., distinguishing "departure time" from "arrival time"), bridging the gap between computer vision and natural language understanding. **A representative example is shown in Fig.13.**

I.2.7 Brief Recap (BR): Visual Content Summarization

This task tests the ability to synthesize, filter, and summarize information from lengthy visual text or complex visual scenes. In real-world applications, users often require concise summaries rather than exhaustive descriptions. Models must demonstrate strong "information bottleneck" capabilities: prioritizing key strategic points, main events, or central arguments while effectively filtering out irrelevant background details, decorative elements, or distracting facts that do not contribute to the core narrative. **A representative example is shown in Fig.14.**

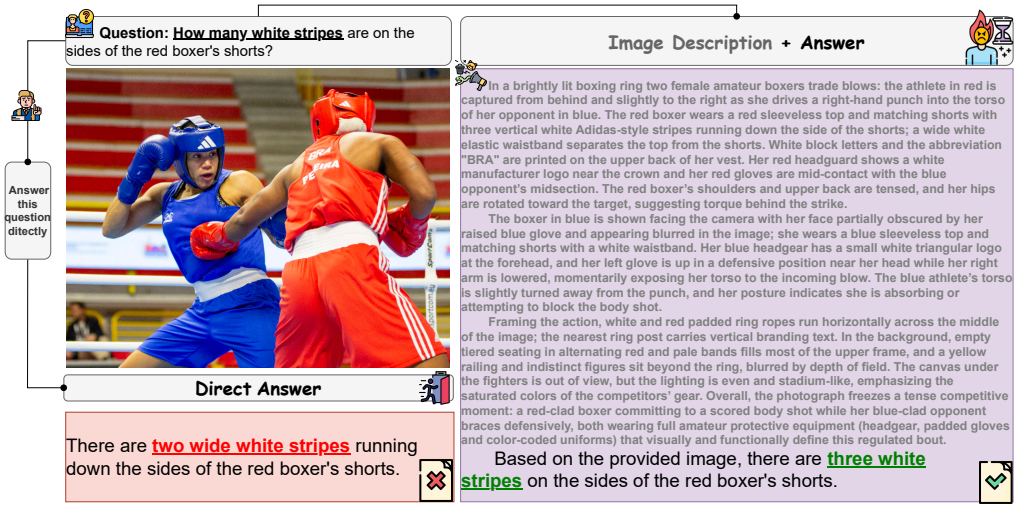


Figure 12: **Sample of Count Measure (CM) Task** The model is asked to count the stripes on the boxer's shorts. The **Direct Answer** (left) suffers from a counting error, incorrectly identifying only "two wide white stripes." The **Correct Answer** (right) demonstrates accurate quantitative perception by correctly counting "three white stripes."

I.2.8 Schema Parse (SP): Structured Data Extraction

Schema Parse involves understanding structured visual data representation, such as charts, infographics, tables, and dashboards. Unlike natural scenes, these images contain implicit syntax and logical rules (e.g., x-axis represents time, grid alignment implies grouping). The model is required to navigate these complex layouts to retrieve specific data points and often perform logical operations—such as comparison, sorting, or finding the maximum value—to answer the query correctly, testing its diagrammatic reasoning skills. **A representative example is shown in Fig.15.**

I.2.9 Compare Change (CC): Difference Detection and Evolution

This category evaluates temporal or comparative reasoning by presenting multi-panel images or sequential states. Models must analyze the differences between two specific visual states—such as the first and last panels of a comic strip—to deduce changes in emotion, environment, character status, or narrative progression. Success relies on detecting subtle visual cues (like a change in facial expression or a movement of an object) and interpreting them within a coherent causal framework. **A representative example is shown in Fig.16.**

I.2.10 Reason Causal (RC): Logical and Causal Inference

Reason Causal tasks demand the highest level of cognitive processing: logical inference. Rather than simply describing what is seen, the model must explain why it is happening. This involves identifying cause-and-effect relationships within the provided context (often a mix of text and image), distinguishing between the actual cause of an event (e.g., a specific market behavior mentioned in a report) and unrelated commentary or future predictions. It tests the model's ability to align visual evidence with textual logic to derive a grounded conclusion. **A representative example is shown in Fig.17.**

I.3 Length Types (3 Classes)

I.3.1 Length Type (Short): Concise Output Generation

This category restricts the output to concise phrases or single sentences, typically for identification or classification tasks. While seemingly simple, it is highly unforgiving: models cannot mask uncertainty with verbose explanations. It tests the model's "precision" and factual accuracy, where even a single word error (e.g., misidentifying a specific machine type) results in a complete failure. It serves as a



Figure 13: **Sample of OCR Extract (OE) Task** The **Direct Answer** (top panel) fails due to an attribute swapping error, reversing the Local and Beijing times. The **Correct Answer** (bottom panel) accurately retrieves and aligns the time values with their respective time zones.

metric for the model's zero-shot classification capability. **A representative example is shown in Fig.18.**

I.3.2 Length Type (Medium): Balanced Explanation

Medium-length responses require a balance between providing a direct answer and offering supporting evidence. This format is ideal for assessing the model's Chain-of-Thought (CoT) reasoning. The model must demonstrate that its conclusion (e.g., identifying an academic subject) is logically derived from observed visual clues (e.g., distinguishing active chalkboard notes from static wall maps). It evaluates the coherence between the reasoning process and the final verdict. **A representative example is shown in Fig.19.**

I.3.3 Length Type (Long): Comprehensive Detailed Description

Long responses challenge the model to generate comprehensive, multi-paragraph descriptions or analyses. The primary technical challenge here is maintaining "long-context consistency" and avoiding "generation drift." As the text length increases, models are prone to hallucinating details or contradicting themselves. This category evaluates the model's capacity to maintain focus, factual accuracy, and narrative structure throughout an extended output covering multiple visual features. **A representative example is shown in Fig.20.**

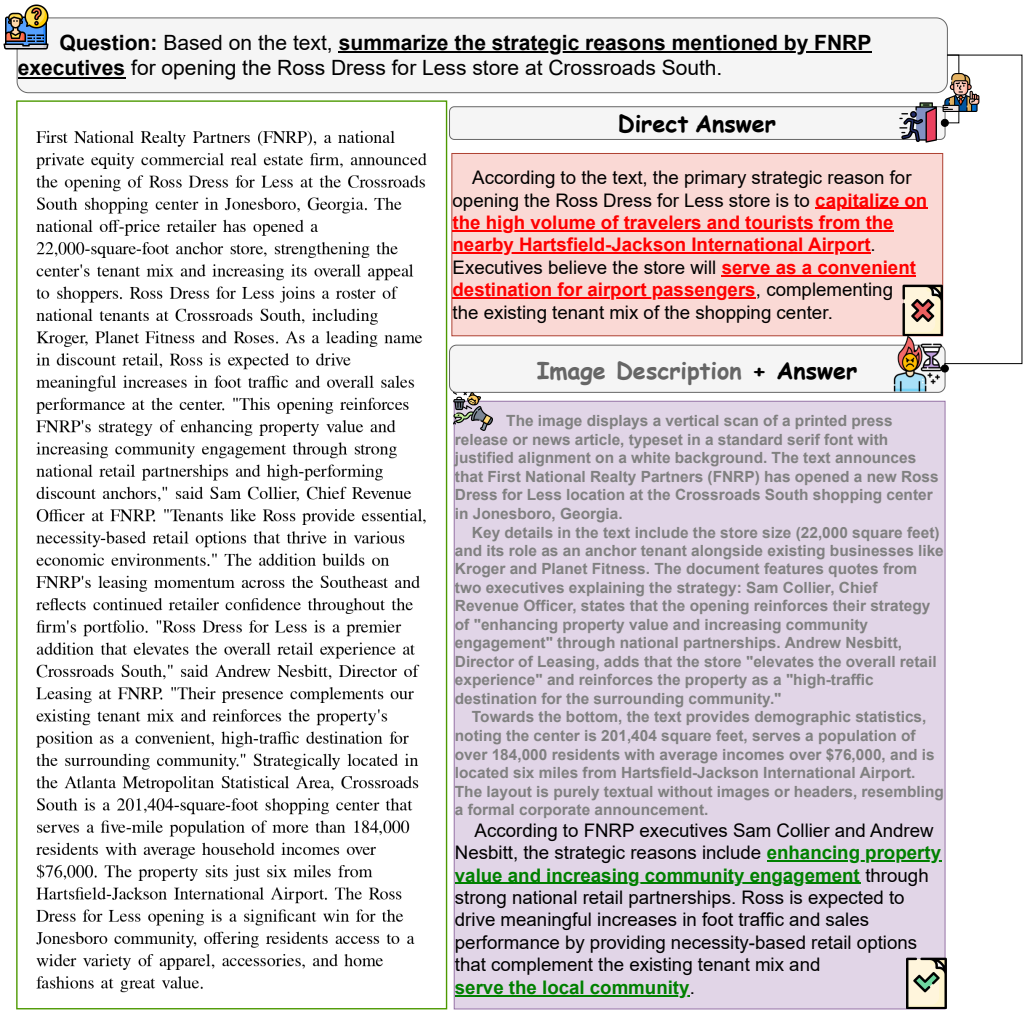


Figure 14: **Sample of Brief Recap (BR) Task** The **Direct Answer** (top panel) is distracted by a location detail, incorrectly inferring a strategy about "airport passengers." The **Correct Answer** (bottom panel) correctly summarizes the actual strategic goals cited by executives: "enhancing property value" and "increasing community engagement."

I.4 Image Types (3 Classes)

I.4.1 Image Type (Images with No Text): Natural Visual Scenes

This category consists of natural photography, artistic renderings, or scenes devoid of textual cues. It isolates the model's pure "visual perception" capabilities, stripping away any semantic hints that text might provide. The model must rely solely on visual pattern recognition to perform tasks like object counting, color detection, and spatial analysis in realistic, unstructured environments. This serves as a baseline for the vision encoder's performance. **A representative example is shown in Fig.21.**

I.4.2 Image Type (Mixed Image-Text): Visual Context with Embedded Text

These samples represent the most complex and common real-world multimodal scenarios, where visual imagery and text are inextricably intertwined (e.g., posters, diagrams with labels, street scenes with signage). Success requires "multimodal alignment"—the ability to perform joint reasoning over both modalities. Models must resolve ambiguities by combining visual features (e.g., color, shape) with textual information (e.g., labels, captions) to extract precise meaning. **A representative example is shown in Fig.22.**

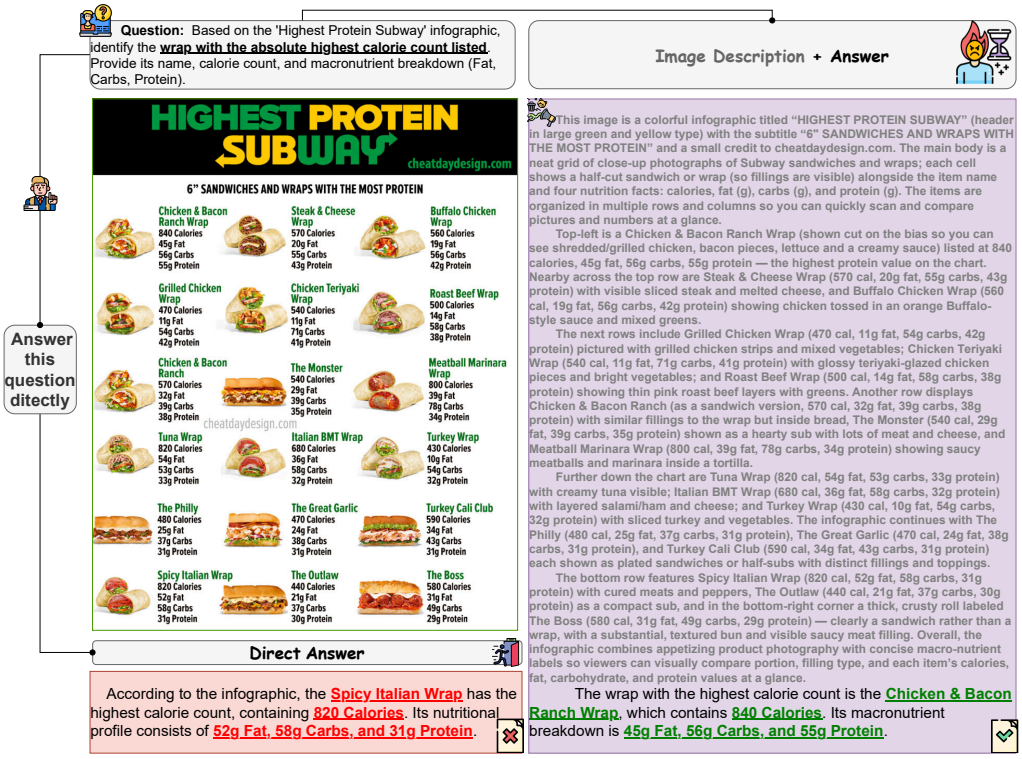


Figure 15: **Sample of Schema Parse (SP) Task** The task requires identifying the item with the highest calorie count from the infographic. The **Direct Answer** (left) fails to find the global maximum, incorrectly selecting an item with 820 Calories. The **Correct Answer** (right) successfully parses the entire grid and identifies the true maximum: the "Chicken & Bacon Ranch Wrap" with 840 Calories.

I.4.3 Image Type (Images with Only Text): Document-Centric Visuals

This category focuses on "Document Intelligence," featuring scans, screenshots, or slides where text is the dominant signal and visual elements are minimal. It pushes the limits of OCR density and tests the model's ability to parse layout semantics (e.g., headers, paragraphs, lists) and extract specific entities from heavy textual noise. This evaluates the model's utility as a reading assistant for dense information retrieval tasks. **A representative example is shown in Fig.23.**

J More InternVQA Quantitative Experiments

Tab. 25 answers whether the ranking depends on the normalization in $S_{\texttt{InternVQA}}$. It collects, for every method whose direct and caption scores are reported in Tabs. 26 and 27, the four-backbone averages of the absolute direct score, the caption score, their gap, and $S_{\texttt{InternVQA}}$, together with the method's rank on each backbone under the direct score, the absolute gap, and $S_{\texttt{InternVQA}}$. `Internalization` has the best reported mean under all three quantities on all four backbones, and Visual-RFT is second on average under all three, so the top of the ranking does not depend on the normalization. On the four-backbone averages, positions in the middle of the table move by at most one place between quantities (for example, NaPO is third under the direct score but fourth under the gap and under $S_{\texttt{InternVQA}}$, behind PAI); on individual backbones the mid-table order varies more, which is the expected behavior of a ratio whose denominator differs across methods. The methods absent from this table (Fast Quiet-STaR, COCONUT, VTool-R1, and GRIT) are reported with $S_{\texttt{InternVQA}}$ only in Tab. 1.

Due to space constraints in the main paper, we provide more fine-grained quantitative results on `InternVQA` in this appendix. We report $S_{\texttt{InternVQA}}$ decomposed by **Length-Type** (Short/Medium/Long), **Image-Type** (Img/Txt/Img+Txt), and **Question Type** (10 categories), so that

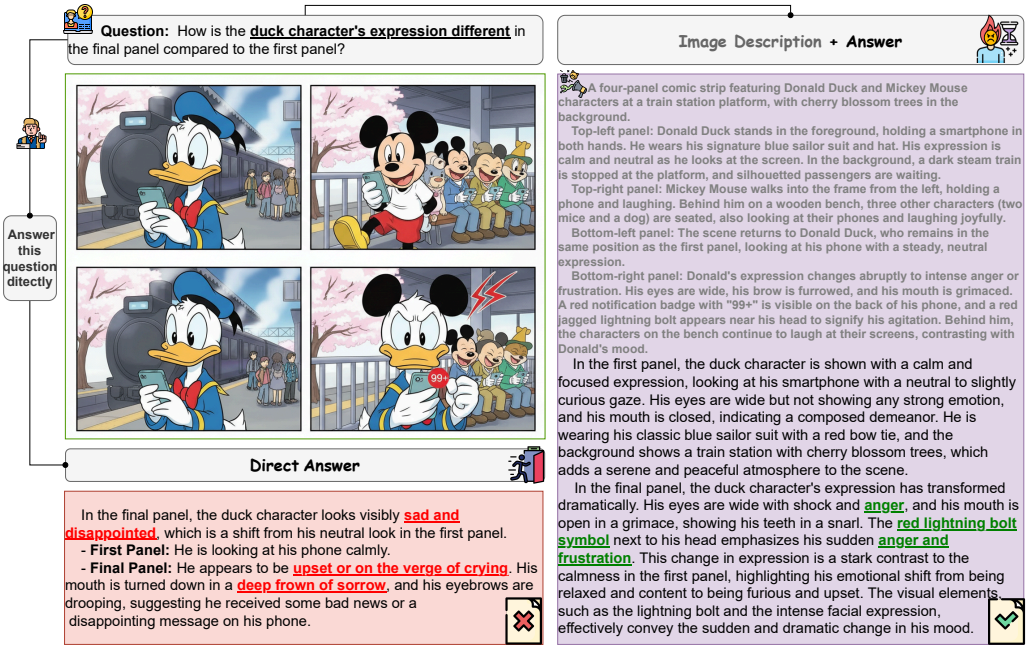


Figure 16: **Sample of Compare Change (CC) Task** The task is to describe the change in the character's expression from the first to the last panel. The **Direct Answer** (left) misinterprets the visual cues, incorrectly describing the angry character as "sad" and "on the verge of crying." The **Correct Answer** (right) correctly identifies the shift to "anger," citing the "red lightning bolt symbol" as evidence of the character's frustration.

Table 25: **Ranking under absolute quantities.** Four-backbone averages of the direct score (Direct), the caption-then-answering score (Cap), their gap, and $S_{\texttt{InternVQA}}$ for every method with reported absolute scores, and each method's rank on every backbone under Direct / Gap / $S_{\texttt{InternVQA}}$ (1 is best). Averages are computed from Tabs. 26 and 27.

Method	Average over four backbones				Rank under Direct / Gap / $S_{\texttt{InternVQA}}$			
	Direct↑	Cap	Gap↓	$S_{\texttt{InternVQA}}$ ↑	LLaVA-OV-0.5B	LLaVA-OV-7B	Qwen3-VL-2B	Qwen3-VL-8B
Base model	40.21	56.83	16.62	51.61	8/7/8	8/8/8	8/8/8	7/8/8
PromptCap	49.27	61.29	12.02	73.58	7/8/7	7/7/7	7/7/7	8/6/6
QD-Cap	51.72	61.02	9.30	80.41	6/6/6	6/6/6	5/5/5	5/5/5
PAI	56.44	62.55	6.11	88.96	3/3/2	3/2/3	6/6/6	3/3/3
OPERA	55.07	62.56	7.50	86.55	5/5/5	4/4/4	4/2/4	6/7/7
NaPO	57.09	64.12	7.03	87.56	2/4/4	5/5/5	3/4/3	4/4/4
Visual-RFT	59.35	64.36	5.01	91.06	4/2/3	2/3/2	2/3/2	2/2/2
Ours	63.11	67.09	3.98	93.48	1/1/1	1/1/1	1/1/1	1/1/1

readers can inspect where each method gains or fails beyond the overall score. We report Direct, Cap, and Gap for both inference settings, with $S_{\texttt{InternVQA}}$ as a relative diagnostic.

Across Qwen3-VL (Tab. 26) and LLaVA-OneVision (Tab. 27), the base models benefit from caption-then-answering. We interpret each method through its paired Direct and Cap scores: a smaller gap is useful when accompanied by stronger direct answers. The subset tables describe performance across the benchmark categories, while the four-backbone absolute comparison in Tab. 25 establishes the overall result.

J.1 Bounded Normalization and Relative Diagnostic

Let D and A be the mean direct and caption scores, respectively. We define the bounded summary

$$N_{\texttt{InternVQA}} = \begin{cases} 100\, D / \max(D, A), & \max(D, A) > 0, \\ 0, & D = A = 0. \end{cases} \tag{44}$$

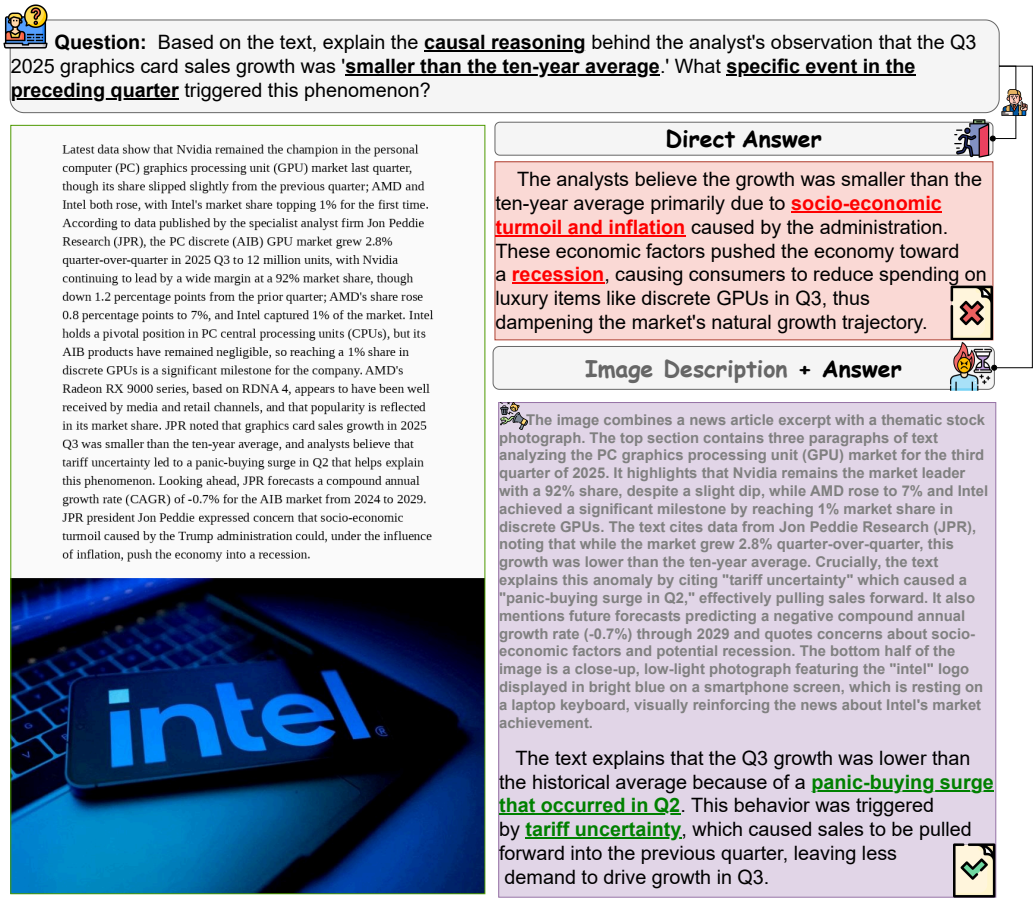


Figure 17: **Sample of Reason Causal (RC) Task** The **Direct Answer** (top panel) fails by misattributing the cause to general "socio-economic turmoil" mentioned in a future context. The **Correct Answer** (bottom panel) correctly identifies the logical cause: a "panic-buying surge" in Q2 due to tariff uncertainty.

Since $0 \le D \le \max(D, A)$, N lies in $[0, 100]$. When captioning helps, N measures the fraction of the caption score achieved by direct answering. When $D \ge A$ and $D > 0$, it saturates at 100 rather than rewarding a deterioration in the caption score. The signed Gap still records the direction of the difference, and Direct records absolute answer quality. Thus, equal weak answers remain distinguishable from equal strong answers in the primary quantities.

The detailed subset tables and ablation figures retain the relative diagnostic $S_{\texttt{InternVQA}} = 100(1 - (A - D)/D)$ for $D > 0$. For the same aggregate scores, $N = 100/(2 - S/100)$ when $A > D$, and $N = 100$ otherwise. Negative S values map to $N < 50$, while $S \ge 100$ maps to $N = 100$. The original sum-based definition is $S = 100[1 - \sum_i (s_{\text{cap}}^{(i)} - s_{\text{direct}}^{(i)})/(\sum_i s_{\text{direct}}^{(i)} + \epsilon)]$, with $\epsilon = 0.01$ only when the direct-score sum is zero and $\epsilon = 0$ otherwise. N is defined directly from D and A above. The two quantities are explicitly distinguished throughout the tables.

Tab. 2 computes N from the displayed three-run Direct and Cap means for each backbone, then averages these four normalized values. This differs from normalizing an already averaged S, because normalization is nonlinear. The same aggregation is used for the base model and `Internalization`. The primary Direct and Gap comparisons in Tab. 25 remain independent of either summary.

Length Type analysis. A clear pattern emerges on both model families: performance gaps are amplified on **Long** questions. While `WithDesc` alleviates the drop for the BaseModel, it does so by introducing explicit intermediate descriptions. In contrast, our method maintains stronger $S_{\texttt{InternVQA}}$ on Long, which matches our goal of reducing reliance on explicit description while keeping answer quality stable under longer contexts.

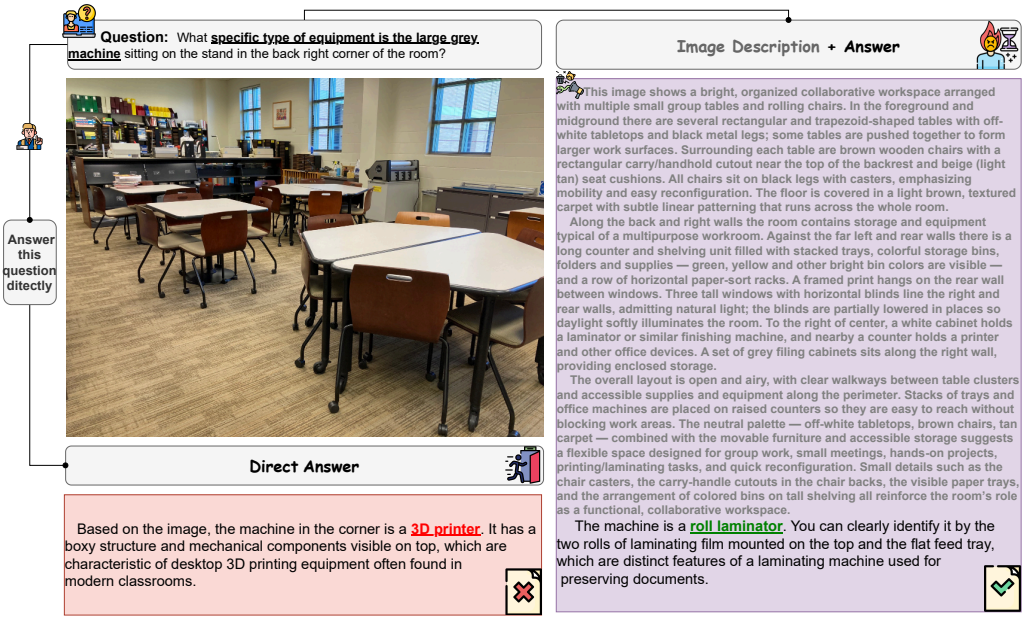


Figure 18: **Sample of Length Type (Short) Task.** The task requires a concise identification of the background equipment. The **Direct Answer** (left) is concise but incorrect, misclassifying the object as a "3D printer." The **Correct Answer** (right) accurately identifies it as a "roll laminator," grounding the short response in specific visual details like the film rolls.

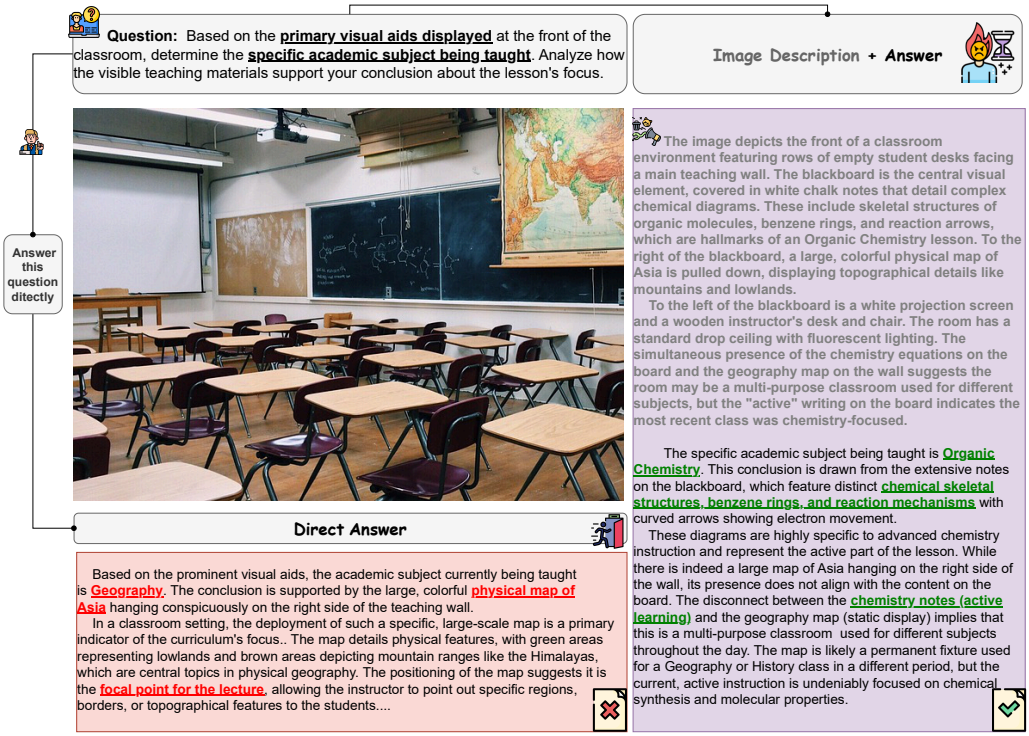


Figure 19: **Sample of Length Type (Medium) Task.** The task requires identifying the subject taught based on visual aids. The **Direct Answer** (left) is distracted by the large map, incorrectly concluding the subject is "Geography." The **Correct Answer** (right) correctly identifies "Organic Chemistry" by focusing on the active chalkboard notes (skeletal structures) rather than the static background decor.



Figure 20: **Sample of Length Type (Long) Task.** The task requires a detailed comparative analysis of two cocktails. The **Direct Answer** (left) generates a fluent narrative but hallucinates specific garnishes like "candied red cherries" and a "spiral of lemon zest." The **Caption-then-Answer** panel (right) shows the original final-answer segment, with the description omitted, and correctly identifies "fresh red currants" and "orange peel." This comparison highlights garnish identification within the long response.

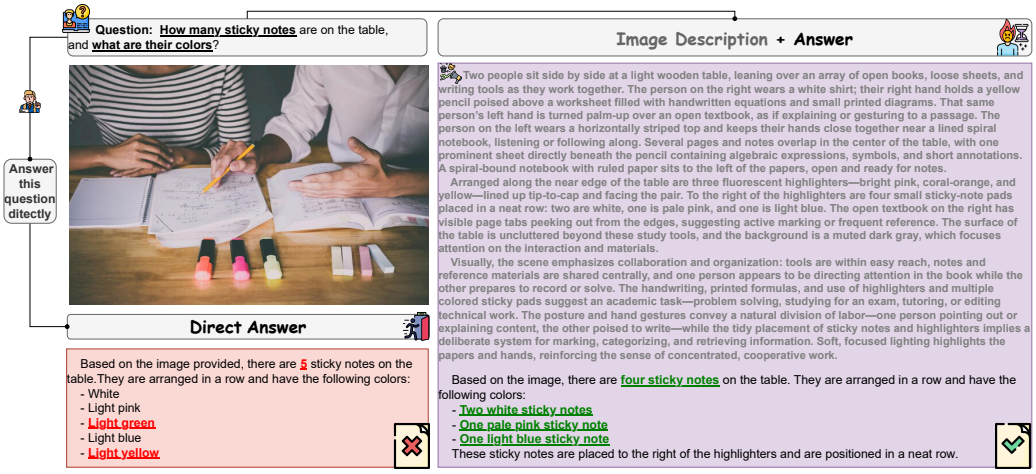


Figure 21: **Sample of Image Category (Images with No Text)** The task is to count and describe sticky notes in a natural scene. The **Direct Answer** (left) hallucinates an extra note (counting 5) and incorrect colors. The **Correct Answer** (right) accurately counts 4 notes and correctly identifies their specific colors (white, pink, blue).

Image Type analysis. The hardest setting is typically **Img+Txt**, where the model must bind visual evidence with text rendered in the image. Several baselines remain uneven across Img/Txt/Img+Txt, indicating modality-specific weaknesses. Our method shows more balanced behavior across modalities, and the gain is more pronounced on Img+Txt, suggesting improved cross-modal grounding without depending on verbose captioning.

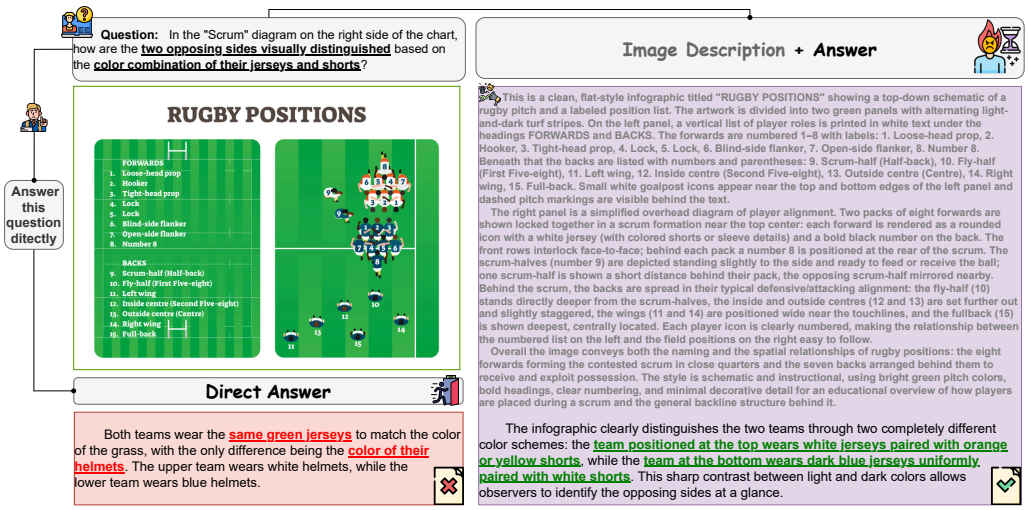


Figure 22: **Sample of Image Category (Mixed Image-Text)** The task requires identifying team jersey colors in a schematic diagram. The **Direct Answer** (left) is confused by the background color, incorrectly stating players wear "green jerseys" and hallucinates "helmets." The **Correct Answer** (right) accurately distinguishes the icons from the background, identifying the correct "white" and "dark blue" jersey combinations.

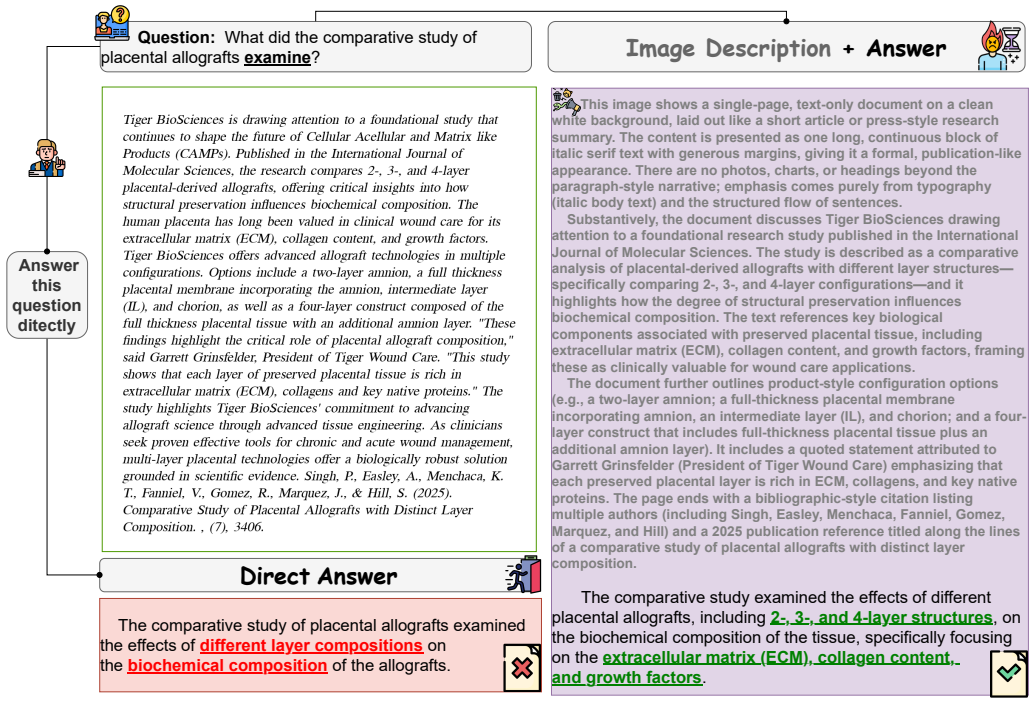


Figure 23: **Sample of Image Category (Images with Only Text)** The task requires extracting specific study parameters from a text document. The **Direct Answer** (left) provides a vague generalization, missing specific details like the layer counts and biological factors. The **Correct Answer** (right) performs precise information extraction, listing the exact "2-, 3-, and 4-layer structures" and specific components like "ECM" and "collagen."

Task Type analysis. Improvements are not uniform across the 10 question types. Captioning-based baselines often lift scores globally but do not consistently address failure modes tied to concept identification and fine-grained grounding. Our method yields stronger gains on the more challenging categories that require tighter visual concept binding or multi-step reasoning, while showing smaller

Table 26: $S_{InternVQA}$ performance on Qwen3-VL.

Method		Length Type			Image Type			Task Type										Overall
		Short↑	Medium↑	Long↑	Img↑	Txt↑	Img+Txt↑	OC↑	PA↑	AE↑	SL↑	CM↑	OE↑	BR↑	SP↑	CC↑	RC↑	
Qwen3-VL-2B																		
BaseModel	w/o Desc	52.83	42.79	34.36	41.62	48.46	48.05	34.03	46.07	37.34	39.76	47.72	75.63	41.66	62.03	40.43	39.69	45.65
	WithDesc	75.73	56.39	44.44	64.88	62.25	61.80	46.27	75.73	58.80	63.46	82.46	87.31	53.81	76.93	56.64	52.38	63.09
	$S_{InternVQA}$	56.65	68.24	70.67	44.11	71.56	71.37	64.07	35.63	42.50	40.40	27.21	84.55	70.84	75.99	59.91	68.05	61.80
PromptCap-2023	w/o Desc	50.96	50.80	50.77	50.77	50.96	50.93	50.70	50.92	50.90	50.91	50.94	50.89	50.79	51.09	50.87	50.91	50.87
	WithDesc	61.06	60.73	60.72	60.99	60.91	60.81	60.85	61.07	61.05	61.02	61.11	61.03	60.85	60.85	60.74	60.64	60.90
	$S_{InternVQA}$	80.18	80.46	80.40	79.89	80.48	80.61	79.98	80.06	80.05	80.14	80.03	80.07	80.19	80.89	80.59	80.89	80.28
QD-Cap-2024	w/o Desc	53.07	52.80	52.77	52.91	52.94	52.95	52.86	53.06	53.06	52.90	53.06	52.86	52.83	53.07	52.90	52.95	52.93
	WithDesc	58.93	58.84	58.72	58.89	58.87	58.81	58.79	58.85	58.98	59.07	59.05	59.10	58.68	58.89	58.78	58.77	58.85
	$S_{InternVQA}$	88.97	88.57	88.72	88.68	88.80	88.94	88.79	89.09	88.85	88.35	88.72	88.19	88.92	89.04	88.89	89.01	88.82
PAI-2024	w/o Desc	51.91	51.73	51.85	51.77	51.80	51.97	51.81	52.05	52.01	51.99	52.06	51.89	51.70	52.06	51.82	51.66	51.86
	WithDesc	57.98	57.91	57.75	57.90	57.79	57.95	57.91	58.09	57.87	58.08	57.88	57.94	57.73	58.02	57.99	57.88	57.90
	$S_{InternVQA}$	88.32	88.05	88.61	88.16	88.43	88.49	88.23	88.39	88.74	88.29	88.82	88.35	88.34	88.55	88.09	87.96	88.35
OPERA-2024	w/o Desc	53.93	53.97	53.84	53.81	53.99	53.99	53.97	53.92	53.90	53.84	53.94	54.05	53.82	54.06	53.88	53.95	53.92
	WithDesc	58.94	58.67	58.82	59.00	58.75	58.79	58.66	59.07	58.97	59.09	58.87	59.12	58.68	58.88	58.92	58.72	58.86
	$S_{InternVQA}$	90.71	91.29	90.75	90.37	91.20	91.11	91.31	90.44	90.59	90.24	90.86	90.63	90.98	91.08	90.64	91.16	90.84
NaPO-2025	w/o Desc	58.94	59.00	58.82	59.03	58.84	58.86	58.80	58.94	58.89	59.08	59.11	58.92	58.78	59.03	59.05	58.89	58.92
	WithDesc	63.89	63.95	63.81	63.90	63.75	63.94	63.65	64.05	63.83	63.97	64.05	64.04	63.86	63.78	63.91	63.76	63.88
	$S_{InternVQA}$	91.59	91.60	91.51	91.75	91.65	91.37	91.75	91.33	91.61	91.72	91.65	91.31	91.36	91.95	91.77	91.73	91.58
Visual-RFT-2025	w/o Desc	59.96	59.82	59.94	59.88	60.11	59.89	60.04	59.89	59.83	60.11	59.99	59.84	59.90	59.92	60.00	59.87	59.93
	WithDesc	64.95	64.81	64.82	64.97	64.83	64.83	64.82	65.04	65.05	65.05	64.69	64.97	64.80	64.77	64.80	64.95	64.88
	$S_{InternVQA}$	91.68	91.67	91.86	91.49	92.14	91.74	92.03	91.40	91.29	91.78	92.16	91.44	91.82	91.91	92.00	91.52	91.74
Ours	w/o Desc	67.93	59.70	55.56	60.86	63.75	64.21	55.74	64.04	55.45	58.59	63.38	82.93	58.60	77.45	62.19	58.63	62.82
	WithDesc	74.57	63.40	55.55	67.01	67.70	66.61	55.65	72.44	62.84	65.29	74.26	85.84	61.19	80.45	65.16	60.62	67.00
	$S_{InternVQA}$	90.21	93.81	100.03	89.89	93.81	96.25	100.15	86.89	86.68	88.57	82.83	96.49	95.59	96.13	95.22	96.62	93.35
Qwen3-VL-8B																		
BaseModel	w/o Desc	63.26	66.31	58.15	56.18	66.25	66.49	56.32	54.46	55.76	52.56	55.84	80.15	63.54	76.56	61.08	63.22	62.45
	WithDesc	82.41	81.35	69.94	77.77	76.99	80.62	69.68	81.71	73.79	76.30	88.26	86.95	77.14	83.17	78.62	75.65	78.72
	$S_{InternVQA}$	69.71	77.33	79.73	61.57	83.79	78.75	76.27	49.97	67.66	54.82	41.95	91.52	78.60	91.37	71.29	80.34	73.95
PromptCap-2023	w/o Desc	61.90	61.87	61.94	61.85	62.07	61.87	61.97	61.94	62.13	61.89	61.90	61.94	61.81	62.07	61.92	61.77	61.90
	WithDesc	73.95	73.92	73.92	73.94	73.80	74.00	73.96	73.97	73.90	73.89	73.96	73.96	74.01	73.95	73.89	73.80	73.94
	$S_{InternVQA}$	80.53	80.52	80.66	80.46	81.11	80.39	80.65	80.59	81.05	80.61	80.52	80.59	80.27	80.87	80.66	80.53	80.55
QD-Cap-2024	w/o Desc	64.87	65.02	64.85	64.86	64.89	64.94	64.75	64.98	64.85	65.09	65.03	65.14	64.82	64.90	64.85	64.78	64.90
	WithDesc	74.89	75.08	74.73	74.85	74.76	74.97	74.56	75.05	74.77	75.09	75.07	75.11	74.78	75.00	74.98	74.69	74.88
	$S_{InternVQA}$	84.56	84.52	84.77	84.60	84.79	84.55	84.85	84.49	84.70	84.65	84.56	84.69	84.63	84.43	84.37	84.70	84.62
PAI-2024	w/o Desc	68.84	69.01	68.80	68.75	68.77	69.03	68.62	68.94	68.83	69.03	68.87	68.94	68.79	69.08	68.84	68.83	68.86
	WithDesc	75.95	75.95	75.73	75.97	75.85	75.85	75.81	76.13	75.89	76.10	76.14	75.94	75.81	75.89	75.78	75.71	75.89
	$S_{InternVQA}$	89.67	89.93	89.92	89.51	89.69	90.12	89.53	89.56	89.75	89.77	89.45	89.85	89.80	90.14	89.93	90.00	89.79
OPERA-2024	w/o Desc	64.95	64.89	64.72	64.80	64.99	64.89	64.58	65.02	64.95	65.00	64.79	64.90	64.94	65.07	64.74	64.72	64.88
	WithDesc	77.97	77.92	77.60	77.72	77.88	77.98	77.61	77.86	78.02	77.82	78.06	78.02	77.87	77.83	77.71	77.76	77.86
	$S_{InternVQA}$	79.96	79.92	80.10	80.06	80.17	79.83	79.82	80.26	79.88	80.29	79.52	79.79	80.08	80.40	79.97	79.85	79.99
NaPO-2025	w/o Desc	68.15	65.85	59.01	64.22	65.09	66.08	60.21	67.83	65.57	68.23	67.12	69.07	64.80	67.17	64.44	59.97	65.14
	WithDesc	76.02	74.20	66.16	72.15	72.29	73.99	66.97	76.08	73.54	75.27	75.57	75.93	72.34	75.34	72.30	68.75	72.91
	$S_{InternVQA}$	88.45	87.32	87.88	87.65	88.93	88.02	88.78	87.83	87.85	89.70	87.40	90.06	88.36	87.83	87.81	85.36	88.07
Visual-RFT-2025	w/o Desc	71.40	71.39	71.41	71.25	71.48	71.51	71.56	71.39	71.40	71.35	71.30	71.32	71.50	71.31	71.47	71.26	71.40
	WithDesc	75.21	75.15	75.08	75.28	75.09	75.09	75.14	75.30	75.26	75.31	75.03	75.11	75.08	75.24	75.05	75.21	75.16
	$S_{InternVQA}$	94.67	94.73	94.86	94.35	94.95	95.00	95.00	94.53	94.60	94.45	94.77	94.68	95.00	94.49	95.00	94.46	94.73
Ours	w/o Desc	74.49	78.29	71.87	70.21	77.10	77.35	71.84	71.54	68.56	67.16	65.24	85.22	76.08	84.91	73.11	75.74	74.53
	WithDesc	80.12	80.32	71.93	77.03	77.74	78.79	71.60	80.92	74.01	72.17	78.42	86.90	77.88	83.52	74.79	77.13	77.88
	$S_{InternVQA}$	92.45	97.41	99.92	90.28	99.18	98.14	100.33	86.88	92.06	92.54	79.80	98.03	97.63	101.64	97.70	98.17	95.51

Table 27: $S_{InternVQA}$ performance on LLaVA-OneVision.

Method		Length			Image Type			Task Type										Overall
		Short↑	Medium↑	Long↑	Img↑	Txt↑	Img+Txt↑	OC↑	PA↑	AE↑	SL↑	CM↑	OE↑	BR↑	SP↑	CC↑	RC↑	
LLaVA-OneVision-0.5B																		
BaseModel	w/o Desc	27.86	12.62	8.77	25.95	15.22	15.44	12.50	36.12	19.17	23.35	43.98	30.11	11.04	14.38	12.61	11.89	19.45
	WithDesc	51.07	19.30	12.60	44.24	27.86	27.15	17.26	64.15	35.95	44.54	72.59	53.85	17.97	33.09	23.28	17.09	33.91
	$S_{InternVQA}$	16.71	47.09	56.38	29.50	16.93	24.20	61.88	22.37	12.42	9.22	34.96	21.16	37.27	-30.08	15.28	56.23	25.66
PromptCap-2023	w/o Desc	33.50	33.50	33.29	33.45	33.55	33.37	33.49	33.61	33.36	33.40	33.44	33.45	33.50	33.42	33.39	33.29	33.44
	WithDesc	48.51	48.34	48.37	48.55	48.48	48.31	48.34	48.56	48.58	48.58	48.39	48.45	48.43	48.34	48.35	48.40	48.44
	$S_{InternVQA}$	55.20	55.71	54.70	54.88	55.52	55.24	55.63	55.53	54.36	54.53	55.30	55.16	55.44	55.34	55.19	54.63	55.14
QD-Cap-2024	w/o Desc	35.26	35.12	34.91	35.31	35.05	35.02	35.19	35.05	35.21	35.13	35.33	35.15	35.11	35.30	34.98	35.07	35.14
	WithDesc	47.54	47.40	47.42	47.47	47.52	47.47	47.61	47.56	47.53	47.53	47.71	47.55	47.30	47.52	47.58	47.39	47.48
	$S_{InternVQA}$	65.17	65.01	64.17	65.57	64.39	64.44	64.72	64.31	64.99	64.71	64.95	64.72	65.27	65.38	63.99	64.85	64.88
PAI-2024	w/o Desc	46.16	46.10	46.00	46.20	46.07	46.02	46.17	46.13	46.03	46.08	46.07	46.13	46.09	46.25	46.02	46.07	46.10
	WithDesc	52.68	52.38	52.26	52.59	52.50	52.41	52.41	52.52	52.56	52.75	52.58	52.57	52.33	52.70	52.59	52.42	52.50
	$S_{InternVQA}$	85.87	86.39	86.39	86.17	86.02	86.12	86.49	86.16	85.82	85.53	85.85	86.04	86.45	86.07	85.72	86.20	86.12
OPERA-2024	w/o Desc	44.52	44.60	44.58	44.49	44.61	44.57	44.57	44.63	44.57	44.66	44.65	44.66	44.48	44.62	44.37	44.48	44.55
	WithDesc	51.61	51.55	51.65	51.76	51.50	51.52	51.55	51.84	51.78	51.88	51.81	51.68	51.46	51.65	51.51	51.41	51.61
	$S_{InternVQA}$	84.06	84.40	84.16	83.66	84.56	84.42	84.35	83.84	83.83	83.83	83.95	84.28	84.32	84.26	83.91	84.44	84.15
NaPO-2025	w/o Desc	47.77	47.76	47.66	47.78	47.67	47.74	47.87	47.71	47.74	47.89	47.71	47.78	47.72	47.69	47.67	47.68	47.74
	WithDesc	54.63	54.41	54.43	54.55	54.52	54.51	54.42	54.64	54.71	54.66	54.72	54.54	54.34	54.52	54.60	54.53	54.53
	$S_{InternVQA}$	85.66	86.08	85.80	85.83	85.62	85.84	86.31	85.47	85.42	85.86	85.29	85.86	86.12	85.69	85.47	85.64	85.78
Visual-RFT-2025	w/o Desc	45.23	45.22	45.07	45.27	45.03	45.18	45.32	45.42	45.20	45.36	45.15	45.35	45.00	45.18	45.29	45.03	45.18
	WithDesc	51.69	51.32	51.43	51.64	51.51	51.47	51.57	51.59	51.58	51.56	51.60	51.65	51.47	51.62	51.52	51.45	51.54
	$S_{InternVQA}$	85.72	86.50	85.88	85.95	85.61	86.08	86.23	86.41	85.88	86.34	85.72	86.09	85.62	85.74	86.23	85.75	85.92
Ours	w/o Desc	50.21	49.89	50.19	50.16	50.22	50.09	50.18	50.31	50.27	50.08	50.28	50.33	50.11	50.07	50.19	49.82	50.14
	WithDesc	54.64	54.44	54.45	54.70	54.61	54.36	54.38	54.62	54.65	54.47	54.66	54.69	54.54	54.41	54.59	54.47	54.55
	$S_{InternVQA}$	91.18	90.89	91.51	90.94	91.25	91.47	91.64	91.43	91.30	91.23	91.30	91.33	91.16	91.33	91.23	90.66	91.20
LLaVA-OneVision-7B																		
BaseModel	w/o Desc	44.01	24.79	19.50	36.17	31.00	31.73	26.23	46.01	31.42	29.62	49.93	55.46	22.81	46.51	29.74	21.61	33.29
	WithDesc	70.34	34.19	29.35	58.22	47.03	47.64	36.34	75.51	52.09	59.21	81.24	79.67	35.15	66.14	44.58	30.17	51.59
	$S_{InternVQA}$	40.17	62.05	49.48	39.03	48.30	49.84	61.42	35.89	34.20	0.13	37.30	56.34	45.88	57.80	50.12	60.39	45.03
PromptCap-2023	w/o Desc	50.98	50.78	50.75	51.01	50.90	50.73	50.75	50.93	51.02	51.08	51.07	50.93	50.78	51.10	50.78	50.66	50.87
	WithDesc	62.03	61.72	61.75	61.87	61.82	61.94	61.79	62.09	62.08	61.94	61.98	62.06	61.69	62.07	61.97	61.70	61.89
	$S_{InternVQA}$	78.33	78.45	78.33	78.72	78.54	77.89	78.23	78.09	78.33	78.74	78.64	78.15	78.50	78.55	77.95	78.21	78.34
QD-Cap-2024	w/o Desc	54.00	53.74	53.85	53.81	54.00	53.96	53.94	54.04	53.84	53.98	53.99	54.00	53.90	53.85	53.74	53.85	53.91
	WithDesc	63.02	62.70	62.78	62.92	62.85	62.88	62.75	63.10	63.02	62.82	63.10	63.09	62.89	62.81	62.75	62.64	62.89
	$S_{InternVQA}$	83.31	83.34	83.43	83.07	83.61	83.47	83.65	83.22	82.94	83.62	83.11	83.17	83.32	83.36	83.24	83.67	83.34
PAI-2024	w/o Desc	58.93	58.82	59.02	58.94	58.92	58.94	58.85	58.93	58.90	58.93	59.15	58.91	58.87	59.07	58.88	59.02	58.94
	WithDesc	63.96	63.74	63.89	63.94	63.90	63.86	63.87	64.02	64.00	64.08	63.88	63.89	63.90	63.85	63.75	63.79	63.90
	$S_{InternVQA}$	91.47	91.65	91.75	91.52	91.56	91.66	91.48	91.36	91.34	91.26	92.01	91.55	91.45	91.90	91.73	91.92	91.58
OPERA-2024	w/o Desc	56.97	56.89	56.85	57.02	56.78	56.91	56.95	56.91	57.02	57.04	56.90	57.10	56.75	57.04	57.01	56.90	56.92
	WithDesc	62.08	61.87	61.71	61.98	61.93	61.89	61.82	62.08	62.10	62.09	61.88	61.94	61.78	62.05	61.99	61.91	61.93
	$S_{InternVQA}$	91.04	91.24	91.45	91.30	90.92	91.25	91.46	90.92	91.08	91.13	91.26	91.52	91.13	91.20	91.27	91.20	91.20
NaPO-2025	w/o Desc	56.49	56.83	56.53	56.48	56.24	56.85	56.98	56.30	56.73	57.81	54.57	56.19	57.16	56.29	56.60	56.06	56.57
	WithDesc	65.16	65.06	65.23	64.91	65.23	65.36	65.48	64.61	65.22	65.41	63.95	65.10	65.52	65.42	65.01	65.09	65.16
	$S_{InternVQA}$	84.65	85.52	84.61	85.08	84.01	85.04	85.08	85.23	85.04	86.85	82.82	84.14	85.38	83.79	85.15	83.89	84.82
Visual-RFT-2025	w/o Desc	60.96	60.94	60.77	61.00	60.83	60.86	60.85	61.07	61.03	61.03	60.92	60.84	60.79	60.85	60.78	61.03	60.90
	WithDesc	65.96	65.71	65.81	65.85	65.82	65.92	65.76	65.90	65.73	65.93	65.74	65.95	65.94	65.84	65.88	65.84	65.87
	$S_{InternVQA}$	91.80	92.17	91.70	92.05	91.79	91.69	91.93	92.08	92.31	91.96	92.09	91.61	91.53	91.81	91.61	92.11	91.84
Ours	w/o Desc	65.00	64.90	64.81	64.87	64.88	65.01	64.76	65.03	64.86	64.93	65.03	65.03	64.83	65.07	65.00	64.95	64.93
	WithDesc	68.94	68.98	68.85	68.85	68.88	69.02	68.84	68.97	68.91	68.93	69.19	68.93	68.87	68.93	69.11	68.78	68.92
	$S_{InternVQA}$	93.94	93.71	93.76	93.86	93.84	93.84	93.69	93.94	93.75	93.83	93.59	94.01	93.77	94.07	93.68	94.10	93.85

changes on simpler categories. This aligns with the intended effect: transferring the useful semantics of description into an implicit prior, rather than producing extra text at inference time.

These expanded results support the same conclusion as in the main paper: explicit captioning (`WithDesc`) helps the BaseModel but at the cost of additional tokens and latency, whereas our method achieves stronger overall $S_{\texttt{InternVQA}}$ with better robustness across length, image, and task-type splits.

K INTERNVQA QUALITATIVE TEST

To qualitatively illustrate the output differences across methods and enable a more fine-grained comparison, we present several representative examples below, showing the generated responses under the same inputs.

K.1 QUALITATIVE EXAMPLE 1

Qualitative comparison. As shown in Fig. 24, we qualitatively compare how different post-training strategies affect *answering behavior* and *output style* under two inference settings on the same visual query: (i) **Direct Answer** (answer the question immediately) and (ii) **Description then Answer** (force a brief description first, abbreviated with "..." when long). This comparison isolates two practical requirements that matter in real usage: **(a) correctness under direct answering** and **(b) style retention** (remaining concise and question-focused rather than drifting into narration).

Baselines often rely on an implicit "describe-first" habit. A common failure pattern is that many baselines still prepend a short caption even in **Direct Answer**, and then produce a *hedged* or *partially grounded* answer. Concretely, they may start with a surface summary such as "a desk with stationery and notes" and then respond with uncertainty (e.g., "4 or 5 sticky notes") or with incomplete evidence (e.g., listing only a subset of colors). In this mode, errors frequently come from (1) **over-counting** by confusing sticky notes with similarly salient objects (e.g., highlighters or other colored items), (2) **hallucinating** a color that is not present, and (3) **style drift**, where the model keeps adding scene details unrelated to the question.

Forcing description can improve grounding, but it trades off efficiency and style. When we switch to **Description then Answer**, several baselines become more stable in counting and color identification, suggesting that they obtain useful intermediate evidence from the description. However, this comes with two drawbacks. First, the description injects extra tokens and latency, which is undesirable for deployment. Second, it changes the output toward verbose narration: the model enumerates pens, tapes, desk layout, and other objects before reaching the actual answer, and the final answer may still carry uncertainty or extra qualifiers. This shows that the baseline improvements often come from *relying on explicit intermediate text* rather than learning to answer directly.

Our method narrows this gap by internalizing description benefits into direct answering. In contrast, our method is designed to *retain the accuracy benefits of "describe-first" without emitting the description in the inference chain*. In **Direct Answer**, our outputs remain concise and question-aligned: we directly provide an exact count and a precise color list, without adding narrative inventory or hedging. Importantly, the output **preserves a stable answering style** across settings, instead of switching to a different "storytelling" mode when prompted to describe first. Overall, these qualitative results align with our quantitative findings: other methods may look competitive only when they are allowed (or implicitly choose) to describe, whereas our method consistently supports **direct answering** with both **accuracy** and **style retention**.

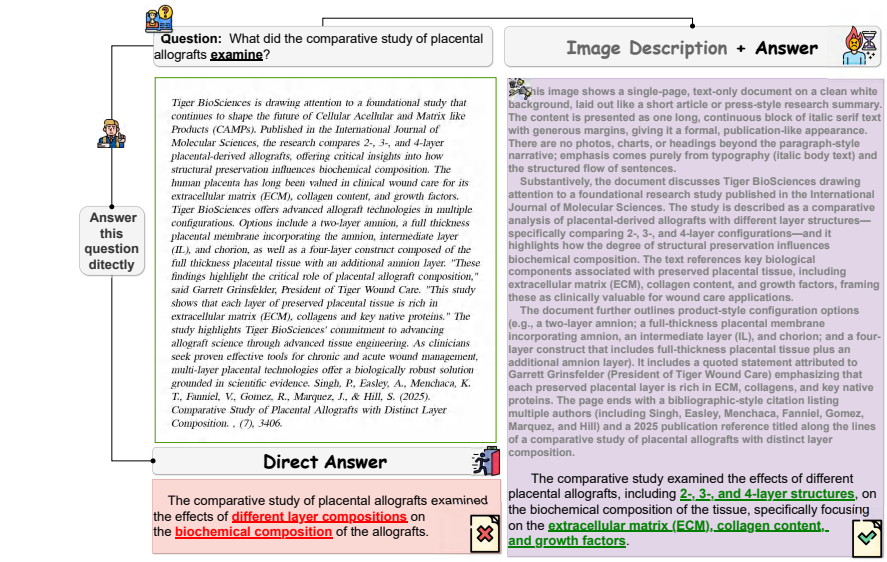
K.2 QUALITATIVE EXAMPLE 2

As shown in Fig. 25, this example requires extracting specific facts from a paragraph-style document. A correct answer must contain both (i) the compared variants, namely 2-layer, 3-layer, and 4-layer placental allografts, and (ii) the examined factors, namely biochemical composition with explicit mentions of ECM, collagen content, and growth factors.

Under **direct answering**, several baselines exhibit a habitual *mini-caption-then-answer* behavior. This often leads to either **incompleteness** (e.g., stating "different layer compositions affect biochemical composition" but missing the explicit 2/3/4-layer comparison, as in BASEMODEL) or **topic drift** (e.g., shifting toward generic wound-care outcomes or product framing, as in PROMPTCAP-2023). When we **force description first**, most methods become more accurate because the description stage



Figure 24: Qualitative example 1, we show the output differences across methods.



PromptCap - Normal Answer

A paragraph style document summarizes a study on placental derived allografts and highlights differences across multiple layered constructs.

It examined what different placental allografts are and how their composition might relate to wound care outcomes.

PromptCap - Forced Description Answer

Description: The document describes a comparative study of placental derived allografts, explicitly contrasting 2 layer, 3 layer, and 4 layer constructs, and stresses that preserved structure influences biochemical composition for wound care...
Answer: It examined how different layer structures, specifically 2 layer, 3 layer, and 4 layer placental allografts, change biochemical composition such as ECM related components, collagen content, and growth factors.

QD-Cap - Normal Answer

The page reports a comparative analysis of placental tissues used for wound care and lists several multi layer configurations.
It examined differences among 2, 3, and 4 layer placental allografts and how those differences affect tissue composition.

QD-Cap - Forced Description Answer

Description: A text only article compares 2 layer, 3 layer, and 4 layer placental derived allografts and connects structural preservation to biochemical composition, mentioning ECM, collagen, and growth factors...
Answer: It examined how layer structure, including 2 layer, 3 layer, and 4 layer allografts, influences biochemical composition, with emphasis on ECM, collagen content, and growth factors.

PAI - Normal Answer

A formal research summary describes a comparative study on placental derived allografts with distinct layered constructs and notes key biomolecular components.
It examined how varying the number of preserved placental layers changes biochemical composition, especially ECM and collagen related factors.

PAI - Forced Description Answer

Description: The text describes a comparative analysis of placental derived allografts across distinct layer structures and emphasizes that preservation affects biochemical composition, citing ECM, collagen content, and growth factors...
Answer: It examined the impact of different layer structures, specifically 2 layer, 3 layer, and 4 layer placental allografts, on biochemical composition including extracellular matrix, collagen content, and growth factors.

OPERA - Normal Answer

The document summarizes a comparative study on placental allografts and discusses biochemical components relevant to wound care.
It examined biochemical composition differences in placental allografts, focusing on ECM, collagen, and growth factors.

OPERA - Forced Description Answer

Description: A text only page compares placental derived allografts with distinct layer structures and ties structural preservation to biochemical composition, listing ECM, collagen, and growth factors...
Answer: It examined how different layer structures, including 2 layer, 3 layer, and 4 layer constructs, affect biochemical composition such as extracellular matrix, collagen content, and growth factors.

NAPO - Normal Answer

It examined how different placental allograft layer structures, specifically 2 layer, 3 layer, and 4 layer, influence biochemical composition, including ECM, collagen content, and growth factors.

NAPO - Forced Description Answer

Description: The page describes a comparative study of placental derived allografts with distinct layered constructs, stressing that structural preservation shapes biochemical composition, and repeatedly mentions ECM, collagen content, and growth factors...
Answer: It examined the effects of different layered placental allografts, 2 layer, 3 layer, and 4 layer, on biochemical composition, focusing on extracellular matrix components, collagen, and growth factors.

Visual-RFT - Normal Answer

The study compared 2 layer, 3 layer, and 4 layer placental derived allografts to measure how layer structure affects biochemical composition, especially ECM, collagen content, and growth factors.

Visual-RFT - Forced Description Answer

Description: A single page text summary presents a comparative analysis of placental derived allografts across multiple layer structures and explains how preservation impacts biochemical composition for wound care applications...
Answer: It examined how different layer structures, 2 layer, 3 layer, and 4 layer, change biochemical composition such as extracellular matrix, collagen content, and growth factors.

Ours - Normal Answer

It examined how 2 layer, 3 layer, and 4 layer placental allografts differ in biochemical composition, focusing on extracellular matrix, collagen content, and growth factors.

Ours - Forced Description Answer

Description: The text compares placental derived allografts with 2 layer, 3 layer, and 4 layer structures and emphasizes that structural preservation influences biochemical composition, mentioning ECM, collagen, and growth factors...
Answer: It examined the impact of different layer structures, 2 layer, 3 layer, and 4 layer, on biochemical composition including extracellular matrix, collagen content, and growth factors.

Figure 25: Qualitative example 2, we show the output differences across methods.

re-anchors key tokens (2/3/4-layer, ECM, collagen, growth factors) before answering. However, this also introduces additional tokens and a noticeable style change from answering to narrating.

In contrast, **Ours** remains stable in **direct answering** and consistently hits all critical entities without relying on an explicit description chain. This illustrates the intended functional transfer: the trained model answers the document question directly while preserving a concise, question-focused output style.

K.3 Qualitative Example 3

As shown in Fig. 26, this sample tests a precise max-selection rule: the answer must be the *wrap* with the *absolute highest* calories in the chart, and it must report the full macro breakdown (fat, carbs, protein). A common failure mode is **local-maximum confusion**: several wraps cluster at **820 Calories** (e.g., *Tuna Wrap*, *Spicy Italian Wrap*), which can mislead methods that rely on quick pattern matching or partial scanning.

In our simulation, BaseModel and PromptCap-2023 illustrate this error. Under **direct answering**, they produce a short caption and then select an 820-calorie wrap, yielding an answer that appears plausible yet violates the "absolute highest" constraint. When we **force description first**, many methods become more accurate because the description stage enumerates multiple candidates and exposes the true maximum.

In contrast, **Ours** remains stable in **direct answering** and correctly identifies *Chicken & Bacon Ranch Wrap* at **840 Calories** with **45g Fat, 56g Carbs, and 55g Protein**. This highlights our key advantage: the model can directly apply the selection rule and extract the required fields without relying on a verbose description chain, thereby preserving a concise, question-focused output style.

L Failure Cases and Discussion

Although our method improves direct answering and reduces the reliance on verbose descriptions, we still observe several recurring failure cases.

(i) Fine-grained counting under clutter or partial occlusion. When the target objects are small, visually similar, or partially occluded, the model may miss instances or double-count. The desk example in Fig. 1 illustrates a counting and color-enumeration error in the base model: its direct answer lists five notes, including light green and light yellow, whereas the image shows four pads to the right of the highlighters. The caption-assisted answer identifies two white, one pale pink, and one light blue pad. This example illustrates how preliminary description can correct the base model's instance count and color list.

Correctly answering this example does not establish reliable counting under other occlusions or layouts. In cluttered scenes, small objects (e.g., sticky notes) often share similar appearance and can be partially covered, which increases ambiguity at the patch level. In these cases, the model may anchor on the most salient instances and ignore low-visibility ones, leading to under-counting, or may conflate adjacent instances, leading to unstable color enumeration. A practical takeaway is that *direct answering* alone is not sufficient for reliable counting, and the model would benefit from auxiliary signals that explicitly encourage instance separation, such as training on dense counting supervision, region-level consistency constraints, or lightweight visual prompting that highlights candidate instances.

(ii) Limited multilingual robustness due to data imbalance. Our training data contains limited multilingual coverage and is dominated by Chinese and English, which can reduce robustness on non-CH/EN inputs. For example, when the question is asked in Spanish (or the image contains non-Latin scripts), the model may correctly recognize the relevant region but produce an incomplete answer, mix languages in the response, or fail to follow the requested output format. This issue becomes more noticeable for text-heavy images where accurate OCR and language-specific normalization are both required.

The multilingual weakness is consistent with the distribution of our training data, which is dominated by Chinese and English. When the question is asked in a low-resource language or the image



PromptCap - Normal Answer

A grid style nutrition poster compares Subway wraps and sandwiches by listing calories plus fat, carbs, and protein for each item.
Highest calorie wrap: Tuna Wrap, 820 Calories, 54g Fat, 53g Carbs, 33g Protein.

PromptCap - Forced Description Answer

Description: The infographic lists multiple wraps with macros, including Chicken and Bacon Ranch Wrap (840 cal) and Tuna Wrap (820 cal), plus other items like Steak and Cheese Wrap and Spicy Italian Wrap...

Answer: The highest calorie wrap is the Tuna Wrap at 820 Calories, with 54g Fat, 53g Carbs, and 33g Protein.

QD-Cap - Normal Answer

The infographic summarizes Subway menu items and shows calories and macro nutrients for each wrap and sandwich.
Chicken & Bacon Ranch Wrap, 840 Calories, 45g Fat, 56g Carbs, 55g Protein.

QD-Cap - Forced Description Answer

Description: A "Highest Protein Subway" infographic presents wraps and sandwiches with calories and macros. The Chicken & Bacon Ranch Wrap is listed at 840 calories with 45g fat, 56g carbs, 55g protein...

Answer: Chicken & Bacon Ranch Wrap, 840 Calories, 45g Fat, 56g Carbs, 55g Protein.

PAI - Normal Answer

A ranked style chart shows Subway items with photos and nutrition stats including calories and macro breakdown.
Chicken & Bacon Ranch Wrap, 840 Calories, 45g Fat, 56g Carbs, 55g Protein.

PAI - Forced Description Answer

Description: The infographic lists multiple wraps, with some at 820 calories and one higher entry at 840 calories. The top left entry "Chicken & Bacon Ranch Wrap" shows 840 calories, 45g fat, 56g carbs, 55g protein...

Answer: Chicken & Bacon Ranch Wrap, 840 Calories, 45g Fat, 56g Carbs, 55g Protein.

OPERA - Normal Answer

Chicken & Bacon Ranch Wrap, 840 Calories, 45g Fat, 56g Carbs, 55g Protein.

OPERA - Forced Description Answer

Description: A menu infographic lists Subway wraps and sandwiches with calories and macros. The highest calorie wrap shown is Chicken & Bacon Ranch Wrap at 840 calories...

Answer: Chicken & Bacon Ranch Wrap, 840 Calories, 45g Fat, 56g Carbs, 55g Protein.

NAPO - Normal Answer

Chicken & Bacon Ranch Wrap, 840 Calories, 45g Fat, 56g Carbs, 55g Protein.

NAPO - Forced Description Answer

Description: The grid includes several high calorie wraps like Tuna Wrap and Spicy Italian Wrap at 820 calories, but Chicken & Bacon Ranch Wrap is higher at 840 calories with 45g fat, 56g carbs, and 55g protein...

Answer: Chicken & Bacon Ranch Wrap, 840 Calories, 45g Fat, 56g Carbs, 55g Protein.

Visual-RFT - Normal Answer

The image show that ...
Chicken & Bacon Ranch Wrap, 840 Calories, 45g Fat, 56g Carbs, 55g Protein.

Visual-RFT - Forced Description Answer

Description: The infographic lists items with their macro breakdown. The "Chicken & Bacon Ranch Wrap" is explicitly listed at 840 calories, higher than other wraps such as Tuna Wrap and Spicy Italian Wrap at 820 calories...

Answer: Chicken & Bacon Ranch Wrap, 840 Calories, 45g Fat, 56g Carbs, 55g Protein.

Ours - Normal Answer

Chicken & Bacon Ranch Wrap, 840 Calories, 45g Fat, 56g Carbs, 55g Protein.

Ours - Forced Description Answer

Description: The infographic shows a set of wraps and sandwiches with calories and macros. Among wraps, Chicken & Bacon Ranch Wrap is the maximum at 840 calories with 45g fat, 56g carbs, 55g protein...

Answer: Chicken & Bacon Ranch Wrap, 840 Calories, 45g Fat, 56g Carbs, 55g Protein.

Figure 26: Qualitative example 3, we show the output differences across methods.

contains non-Latin scripts, the model must handle OCR noise, language-specific tokenization, and format-following simultaneously, which amplifies small errors in recognition into larger errors in generation. This indicates that our current post-training improves answer grounding primarily within the high-frequency language regimes, but does not guarantee uniform gains across languages. A direct path forward is to expand multilingual coverage during post-training and evaluation, including (i) balanced instruction data across languages, (ii) script-diverse text-heavy images, and (iii) language-aware normalization for OCR tokens. These changes can improve robustness without altering the core method, while keeping the benefit of concise, direct answers.

Overall, both failure modes arise from a mismatch between what our post-training optimizes (identification and concise answering) and what these cases require (instance-level separation and language/script generalization). Addressing them likely needs targeted data and constraints rather than increasing description length, preserving our central goal of answering directly with minimal verbosity.

M Prompts for Data Construction and Benchmark Evaluation

This appendix provides the prompts used in our pipeline. We include two prompt families. The first family constructs Stage I instruction tuning data by asking the teacher model to propose a challenging visual question and answer it. The second family evaluates model outputs on `InternVQA` by scoring both **direct answering** and **caption-then-answering** with a structured rubric conditioned on metadata.

We pass the following fields to the teacher or judge: `{image}`, `{image_type}`, `{length_type}`, `{task_type}`, `{question}`, `{prediction_direct}`, `{prediction_cap}`. The judge returns a scalar score in $[0, 1]$ for each prediction, plus a short rationale.

Stage I data prompt: difficult visual question generation and solving

```
You are given one image.
Your job is to create one VQA sample that is hard and not trivial.
The question must require careful visual grounding and reasoning, not
generic description.  Avoid easy questions such as color only, single
object naming, or yes or no.

Input
Image:  {image}

Requirements
1) Write one question that matches one of these task types:
OC open caption, OE OCR extract, PA part attribute, AE action event,
SL spatial layout, CM count measure, CC compare change, SP schema
parse,
BR brief recap, RC reason causal.
2) The question must be answerable from the image, but should require
multiple cues.
3) If the image contains text, you must incorporate the text in a
meaningful way, not copy only.
4) The answer must be precise, grounded, and self consistent.
5) The sample must be difficult.  Prefer multi step reasoning,
disambiguation, comparison, or constraints.
6) Do not mention that you are an AI. Do not mention policies.  Do not
add extra commentary.

Output format
Return exactly the following fields.
TASK_TYPE: <one of OC, OE, PA, AE, SL, CM, CC, SP, BR, RC>
QUESTION: <one sentence or two sentences>
ANSWER: <concise but complete, include key evidence>
DIFFICULTY_CHECK: <one short sentence explaining why it is hard>
```

Figure 27: **Stage I prompt.** The teacher model generates one challenging question and a grounded answer for each image.

Scoring rationale. The judge scores both answers against one image- and question-conditioned checklist, using the same weights: correctness contributes 60%, coverage 30%, and language quality 10%. The rubric assigns no separate reward to either inference setting or to answer length. Human evaluation hides method identities and randomizes method order, while cross-family judges test whether the leading method is reproduced by different evaluators (Appendix E.2, Sec. 4). These checks support the reported comparisons under the specified scoring protocol.

Format normalization. We apply the same GPT-based format checker uniformly to all methods to ensure that responses conform to the output schema required by each evaluation setting. Non-conforming generations are handled using the same re-decoding procedure across methods. Accepted responses are evaluated as generated, without post-hoc rewriting. We additionally report the description-trigger rate to characterize models' tendency to produce explicit descriptions.

Benchmark judge prompt: structured scoring for InternVQA

You are a strict evaluator for visual question answering. You will receive one image, one question, and two candidate answers produced under two inference settings. You must score each answer in $[0, 1]$.

Inputs

Image: {image}
Image type: {image_type} (one of: Img, Txt, Img+Txt)
Length type: {length_type} (one of: Short, Medium, Long)
Task type: {task_type} (one of: OC, OE, PA, AE, SL, CM, CC, SP, BR, RC)
Question: {question}
Answer A (direct answering): {prediction_direct}
Answer B (caption-then-answering): {prediction_cap}

Step 1: plan the evaluation

First, write a brief checklist of what must be correct to answer this question. The checklist must be adapted to the metadata:

- If image type is Txt, focus on OCR correctness and text based reasoning.
- If image type is Img, focus on visual grounding and non text cues.
- If image type is Img+Txt, require consistency between visual cues and text cues.
- If task type is OE, exact text extraction is required, penalize any mismatch strongly.
- If task type is CM, counting and units must be correct, penalize off by one unless ambiguity is visible.
- If task type is SL, spatial relations must be correct, require explicit relation words.
- If length type is Long, allow longer reasoning but still require correctness and coherence.

Step 2: score each answer with the same rubric

Use three aspects, each in $[0, 1]$, then combine them:

- **Correctness**: factual accuracy with respect to the image and question. This is the most important.
- **Coverage**: whether the answer addresses all required key points from the checklist.
- **Quality**: language logic, fluency, and self consistency. Penalize contradictions or unclear claims.

Compute **FinalScore** = $0.6\cdot$ Correctness $+0.3\cdot$ Coverage $+0.1\cdot$ Quality. If the answer is clearly not grounded in the image, set FinalScore near 0.

Step 3: output

Return exactly the following JSON with no extra keys:

```
{
  "checklist": ["...", "...", "..."],
  "A": {"correctness": x, "coverage": x, "quality": x, "final": x, "
      notes": "..."},
  "B": {"correctness": x, "coverage": x, "quality": x, "final": x, "
      notes": "..."}
}
```

Figure 28: **Benchmark judge prompt.** The judge first states evaluation key points conditioned on metadata, then scores both settings with a consistent rubric.

We pass `image_type`, `length_type`, and `task_type` to the judge so the rubric matches the sample characteristics. This reduces systematic bias across categories and makes the score more diagnostic for caption reliance.

N LIMITATIONS

Internalization relies on preference pairs built from a caption-then-answering pipeline and judge-based scoring. As a result, the training signal inherits the quality and biases of the generated captions and the judge. If captions omit key evidence or introduce spurious details, Stage II may internalize these patterns and yield unstable gains on some task types. Furthermore, Stage I constructs an online subspace and selects high projected-energy tokens based on per-sample statistics. For very short questions, visually sparse images, or noisy OCR inputs, the subspace estimate can be less stable, which affects token selection and the resulting implicit prior. This sensitivity also implies that rank and top-k hyperparameters may require re-tuning across backbones and data regimes.

Failure cases and training cost. `Internalization` still struggles with fine-grained counting in cluttered scenes, overlapping instances, and non-Latin scripts, where an explicit description can provide grounding that the implicit prior does not fully reproduce (Appendix L). Training costs more than one SFT pass because it uses two stages, a frozen teacher, and preference pairs; the Stage I logits head and frozen teacher are training-only (Appendix F). Mixed Stage I labels, judges from other model families, and a human study help assess dependence on caption and judge quality.

We evaluate four backbones from two model families; broader scaling to larger models and additional architectures remains future work.

Scope of description reliance. We measure the answer-quality benefit of the complete caption-then-answer protocol under the prompts reported in Appendix M. Internalization transfers supervision from this protocol to direct answering through an image-question prior. The resulting Direct and Gap improvements quantify functional transfer under this protocol; caption content, preliminary computation, and self-conditioning jointly define the teacher condition. Alternative prompt templates or teacher protocols can change the size of this benefit.

Evidence for internalization. The SFT comparison, label-source ablation, and prior-injection ablation support complementary parts of the training design. Removing prior injection reduces average $S_{\texttt{InternVQA}}$ from 93.48 to 81.52 and RetainPct from 96.45% to 91.87% (Appendix C). These are behavioral and component-level results. They do not identify which caption attributes are represented in the prior or separate all effects of teacher supervision from those of the architecture.

InternVQA scoring. We treat the absolute Direct score and caption-direct Gap as the primary quantities (Tab. 2) and report bounded $N_{\texttt{InternVQA}}$ alongside them, with $S_{\texttt{InternVQA}}$ retained as a relative diagnostic in the detailed results. Tab. 25 shows that the ranking at the top is the same under Direct, Gap, and $S_{\texttt{InternVQA}}$. All InternVQA scores are means over three runs. The reported rankings order these means, rather than establishing statistical separation between nearby methods. Evaluation combines image-conditioned judge scores, reference-based Token F1 and numeric accuracy, cross-family judges, and a single-blind human study. The human study yields a method-level rank correlation of $\rho = 0.944$ with the automatic ranking. These complementary checks support the observed answer-quality gains under the reported evaluation protocol (Appendix E).

O FUTURE WORKS

We hope this work helps push the VLM community toward "stop describing, start answering": reducing unnecessary intermediate text while preserving accuracy and generalization. Concretely, we plan to extend InternVQA to broader modalities and settings, such as multi-image and video inputs, and to develop more reliable automatic evaluation that is less sensitive to caption and judge noise. We also aim to study scaling behaviors under larger backbones and longer contexts, and to design training objectives that better preserve general-purpose capabilities while improving direct-answering behavior.

P GENERATION BENCHMARK OVERVIEW

We evaluate generalization in the main paper using a set of representative VLM benchmarks. Below we provide a brief summary of these benchmarks.

P.1 GQA

GQA is a visual reasoning benchmark designed to address language priors and weak compositional structure in VQA. Built on Visual Genome scene graphs, it features over 22 million questions paired with functional programs that specify multi-step reasoning operations. By promoting a balanced answer distribution and providing diagnostic metrics like consistency and plausibility, GQA encourages models to move beyond superficial correlations toward genuine, grounded scene understanding.

P.2 MMBENCH

MMBench is a competency-centric evaluation suite comprising nearly 3,000 questions across 20 skill areas. A key feature is its *Circular Evaluation* protocol, which requires models to consistently select the correct option across all choice permutations to mitigate positional bias. To handle open-ended outputs, it employs a robust two-stage parsing pipeline (heuristics and LLM-based judgment), ensuring high conversion accuracy for a stable measurement of multimodal building blocks.

P.3 POPE

POPE targets object-level hallucination by formulating evaluation as a binary (yes/no) verification task. It employs three sampling schemes—**Random**, **Popular**, and **Adversarial**—to stress-test a model's visual grounding and robustness against misleading queries. This simple yet effective framework allows researchers to isolate hallucination tendencies from general VQA performance, serving as a vital testbed for mitigation strategies like Visual Contrastive Decoding.

P.4 SCIENCEQA (SQA)

ScienceQA evaluates multimodal reasoning through over 21,000 science problems across Natural, Social, and Language Sciences. It stands out by providing lesson-style lectures and step-by-step rationales for the majority of items, enabling the assessment of a model's ability to generate human-like reasoning processes. The benchmark spans knowledge recall to complex logical inference, emphasizing explainability alongside standard accuracy.

P.5 AI2D

AI2D focuses on reasoning over 1,000 STEM diagrams, such as ecosystems and anatomy, which combine graphical cues (arrows, labels) with text. Models must perform cross-modal alignment to interpret hierarchies, sequences, and part-whole relationships. While domain-specific, it remains a critical reference for OCR-assisted diagram understanding and discourse-level reasoning within educational contexts.

P.6 CHARTQA

ChartQA evaluates the ability to parse and reason over structured visual data like bar, line, and pie charts. It requires models to integrate visual perception with mathematical and trend analysis. Accompanied by detailed annotations of coordinates and styles, ChartQA is a prominent benchmark for applications in business analytics, scientific fact-checking, and automated tutoring tools.

P.7 OCRVQA

OCRVQA integrates optical character recognition into the VQA pipeline, featuring over 200,000 book-cover images and one million QA pairs. It covers 32 categories and requires models to extract and reason over titles, authors, and genres. The benchmark introduces real-world challenges like diverse typography and complex layouts, forcing models to achieve genuine multimodal integration rather than relying on vision-only shortcuts.

P.8 HALLUSIONBENCH (HB)

HallusionBench is a diagnostic tool for disentangling **linguistic hallucination** (knowledge bias) from **perceptual illusion** (visual sensitivity). It uses 1,129 QA pairs with systematic image perturbations, such as flipping or color shifts, to test model robustness. Evaluated via GPT-4, it reveals significant weaknesses in temporal and quantitative reasoning, even among state-of-the-art systems.

P.9 MMSTAR

MMStar is an elite benchmark of 1,500 curated samples designed to ensure **visual dependency** and minimize data leakage. It introduces **Multi-modal Gain (MG)** and **Multi-modal Leakage (ML)** metrics to distinguish genuine visual grounding from text-only solvability. By requiring strict manual review of samples, it exposes models that rely on memorized training data or textual shortcuts rather than visual evidence.

P.10 REALWORLDQA (RWQA)

RealWorldQA is a real world understanding benchmark introduced by xAI to evaluate the *basic real world spatial understanding* of multimodal models, focusing on physically grounded comparisons and spatial reasoning (e.g., relative size, drivable space, lane constraints, and cardinal directions) in everyday scenes. The initial release contains over 700 images, each paired with a question and an easily verifiable answer; the images include anonymized vehicle captured views as well as other real world photographs.