SLVR: Structured Latent Visual Reasoning via Human-like Reasoning Flows

Bo (Albert) Gao
Carnegie Mellon University

Bing Xue*
Meta

Andrea Zanette*
Carnegie Mellon University

Abstract

Multimodal large language models (MLLMs) often answer visual reasoning questions by relying on linguistic priors rather than task-relevant visual evidence. Textual chain-of-thought reasoning can partially mitigate this issue by encouraging models to decompose visual questions into intermediate evidence-seeking steps, but generating these steps autoregressively increases inference cost. Latent reasoning avoids explicit rationale generation, but existing approaches provide limited control over what intermediate states encode, making it difficult to impose separate supervision for planning, grounding, and evidence selection. We propose Structured Latent Visual Reasoning (SLVR), a training framework that bridges explicit chain-of-thought and latent reasoning by organizing multimodal reasoning into typed latent stages for planning, grounding, evidence selection, and reasoning integration. SLVR first trains the model to rely on the image by masking answer-revealing text and contrasting the correct answer with visually plausible distractors. It then organizes reasoning into latent stages for planning, grounding, evidence selection, and integration, supervising each stage with the corresponding signal: plans, boxes, visual evidence, and final rationales. This gives latent reasoning an explicit functional structure while avoiding generated textual chains at inference time. Built on Qwen2.5-VL-7B, SLVR improves consistently across multimodal reasoning benchmarks, with absolute gains of +9.4 on MMVP and +14.2 on BLINK Relation, as well as improvements on V*, MathVista, and ChartQA. These results suggest that structured latent supervision can improve fine-grained visual reasoning without the decoding overhead of textual CoT. Project page is available here.

1 Introduction

Multimodal large language models (MLLMs), which generate language responses from textual and visual inputs, have become a leading approach to vision-language reasoning. Recent systems such as GPT-4o [24], Gemini [9], Qwen-VL [1], and GLM-V [16] show strong performance across visual question answering and multimodal understanding tasks. However, robust visual reasoning remains difficult [7, 32, 8, 38]: MLLMs can produce fluent but incorrect answers by relying on linguistic priors or shortcut correlations rather than task-relevant visual evidence. This failure is especially pronounced in fine-grained tasks that require localizing relevant regions, extracting precise evidence, and integrating it with the query.

A common strategy to improve reasoning is to externalize intermediate steps. Chain-of-thought (CoT) reasoning decomposes problems into natural-language reasoning steps [35], and multimodal extensions further represent intermediate reasoning as textual rationales, visual traces, tool calls, or programs [43, 23, 44, 18, 28, 17, 3, 11, 45, 12]. In training-based variants, these intermediate structures may be supervised with annotated rationales, region boxes, program traces, or tool-execution trajectories, encouraging the model to follow an explicit reasoning path rather than predict

*Co-corresponding authors: Bing Xue and Andrea Zanette.

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

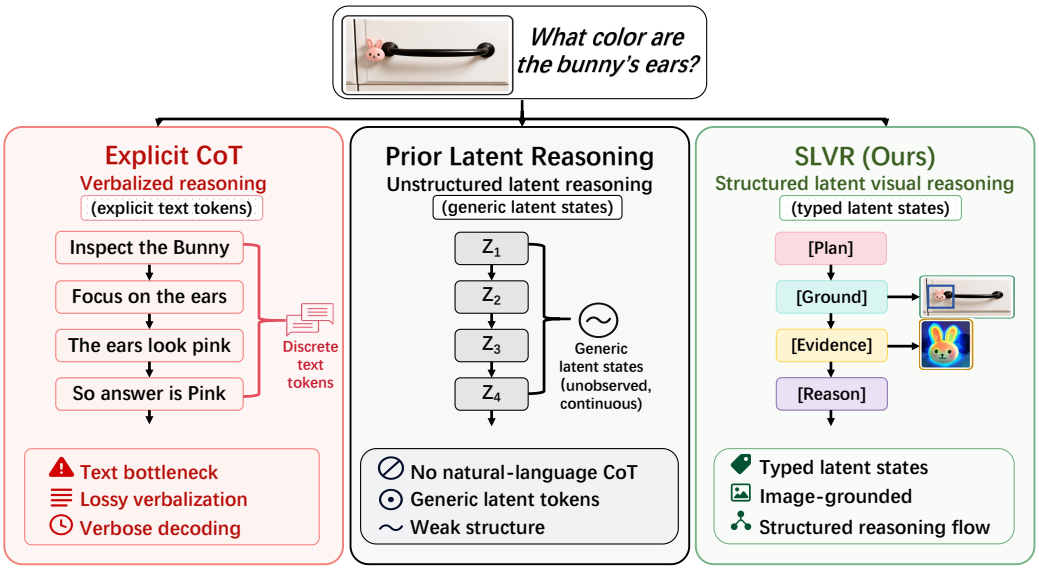


Figure 1: **Comparison of CoT, prior latent reasoning, and SLVR.** Explicit CoT serializes intermediate reasoning into natural-language tokens, introducing a representational bottleneck and decoding overhead. Prior latent reasoning operates in continuous space but typically treats intermediate states as unstructured. SLVR instead organizes latent computation into typed state segments with distinct functional roles for planning, grounding, evidence selection, and reasoning integration.

the answer directly. However, explicit reasoning introduces two key limitations. First, it imposes a *representational bottleneck*: high-dimensional multimodal representations must be serialized into discrete text, resulting in lossy compression of fine-grained visual information. Second, it incurs substantial *computational overhead*, as each reasoning step must be generated autoregressively. As a result, explicit CoT can be both inefficient and insufficient for capturing detailed visual relationships.

Latent reasoning [14, 42, 10, 39, 5, 47] addresses these issues by performing intermediate computation directly in continuous hidden states. Recent multimodal approaches extend this paradigm through continuous multimodal representations, visual token manipulation, or latent alignment techniques [25, 19, 2, 30, 26, 40, 33, 27, 20, 6, 41, 36]. While these methods improve efficiency and preserve richer representations, they typically treat latent reasoning as *unstructured computation*, without explicitly modeling different functional stages such as planning, grounding, and evidence selection.

In this work, we investigate whether latent multimodal reasoning can benefit from introducing *explicit structure and supervision within the latent space*. We propose *Structured Latent Visual Reasoning (SLVR)*, a training framework that organizes latent reasoning into a sequence of functionally distinct stages and supervises each stage with task-specific signals.

SLVR consists of two complementary components. First, we introduce *Visual Dependency Induction (VDI)*, a contrastive training scheme that reduces reliance on textual shortcuts by removing answer-revealing cues and introducing image-grounded distractors. This encourages the model to rely more on visual evidence during reasoning. Second, we structure the model's internal computation using *typed latent state segments*, implemented via special delimiter tokens (e.g., <plan>, <box>, <evidence>, <reason>, <answer>). These segments partition the latent space into functional roles corresponding to planning, visual grounding, evidence selection, and reasoning integration. During training, each segment is supervised with task-specific targets, including action plan embeddings, bounding box coordinates, and visual patch features. This design enforces a stage-wise decomposition of reasoning while retaining the efficiency of latent computation.

We evaluate SLVR on multiple multimodal reasoning benchmarks, including V*, MMVP, BLINK, MathVista, and ChartQA. SLVR consistently improves performance over strong baselines, with particularly large gains on tasks requiring fine-grained spatial and relational reasoning. Notably, when built on Qwen2.5-VL-7B, SLVR achieves improvements of +9.4 on MMVP and +14.2 on BLINK Relation, while avoiding the decoding overhead associated with textual CoT.

Our contributions are summarized as follows:

- **A structured alternative to explicit multimodal CoT.** We introduce a latent reasoning framework that organizes multimodal inference into typed continuous states, providing a compact alternative to decoding long natural-language rationales, visual traces, or program-like reasoning steps.
- **Functionally supervised latent reasoning states.** Rather than treating latent reasoning as an undifferentiated hidden bottleneck, we assign distinct roles to different latent states and supervise them with corresponding targets for planning, grounding, evidence selection, and final answer prediction.
- **A shortcut-resistant visual dependency objective.** We propose visual dependency induction, a contrastive training scheme that removes answer shortcuts from text and uses image-grounded distractors to force reliance on visual evidence.
- **Empirical evidence for grounded latent reasoning.** Experiments across multiple benchmarks show that our framework improves fine-grained visual grounding and compositional reasoning, while reducing reliance on externally decoded reasoning traces.

2 Related Work

2.1 Explicit Multimodal Reasoning

Recent work extends chain-of-thought [35] reasoning to multimodal settings by making intermediate reasoning steps explicit [34]. Early multimodal CoT [43, 23, 44, 18] methods elicit or supervise textual rationales for visual question answering and compositional reasoning, while later methods expand explicit reasoning into richer multimodal traces, such as visual sketchpads, generated visual thoughts, and interleaved vision-text reasoning steps [28, 17, 3, 11, 46, 4]. Another line externalizes visual reasoning through tools or executable programs: methods such as DeepEyes and DeepEyes-v2 [45, 15] actively inspect images with visual tools, while programming-based approaches [12, 13, 31] use code as an interface for cropping, measuring, or analyzing visual content.

These approaches improve interpretability and provide strong supervision by exposing how a model decomposes a visual reasoning problem. However, they still depend on externalized intermediate steps, including textual rationales, tool calls, programs, or generated visual traces, which introduce additional decoding overhead and may force early commitment to discrete reasoning paths. In contrast, we study whether multimodal reasoning can be organized in latent space, keeping intermediate computation continuous and internal while still grounding it in task-relevant visual evidence.

2.2 Latent Multimodal Reasoning

Latent reasoning methods question whether intermediate reasoning must be fully externalized as discrete natural-language tokens. Prior work explores continuous hidden-state thoughts [14], soft concept tokens [42], pause/thought tokens [10], and self-generated internal rationales [39, 5] as alternatives to explicit CoT. These studies motivate reasoning as continuous internal computation, a direction also summarized in recent latent reasoning surveys [47].

In multimodal settings, this direction is especially appealing because MLLMs already align visual and textual information in a shared latent space. Multimodal Coconut [25] extends continuous-thought reasoning to vision-language models. Latent Visual Reasoning [19] supervises hidden states using query-relevant ROI visual tokens, while Interleaved Vision-Text Latent Reasoning [2] injects selected image tokens into latent reasoning steps. LaCoT [30] instead formulates visual reasoning chains as latent variables learned through amortized inference. Other methods explore complementary forms of continuous visual thoughts, including compact visual tokens [26], latent visual scratchpads [40], generated visual embeddings [33], modality-agnostic latent tokens [27], and test-time latent think-token optimization [20]. Recent surveys further summarize the broader shift from explicit multimodal CoT toward image-grounded intermediate computation [29].

Our work follows this latent multimodal reasoning direction, but addresses a different limitation: how to organize latent reasoning rather than only how to represent it. Prior methods improve latent multimodal computation, but their intermediate latent states are often homogeneous or weakly structured. In contrast, we decompose latent visual reasoning into typed functional states for planning, grounding, evidence selection, reasoning integration, and answer decoding. This gives the model an

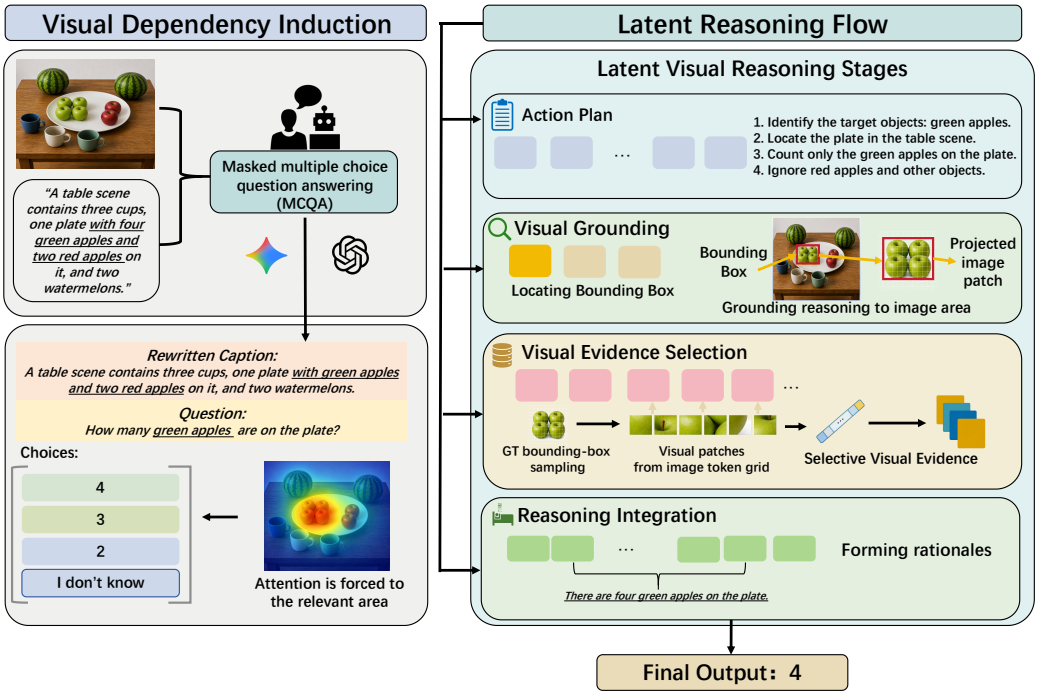


Figure 2: **Overview of SLVR.** SLVR consists of two stages. First, *Visual Dependency Induction* removes answer-revealing textual shortcuts from the caption and constructs masked multiple-choice question answering (MCQA) examples with plausible distractors, encouraging the model to rely on image evidence. Second, the *Latent Reasoning Pipeline* organizes multimodal reasoning into typed latent slots for planning, grounding, visual evidence selection, and reasoning integration. Each stage receives task-specific supervision, including CoT embeddings, bounding boxes, sampled image-token patches, and right answer embeddings.

explicit inductive bias for task-relevant visual reasoning while preserving the efficiency and flexibility of continuous latent computation. A detailed qualitative comparison is provided in Appendix A.4.

3 Method

3.1 Overview of SLVR

We propose **Structured Latent Visual Reasoning (SLVR)**, a two-stage framework for grounded latent multimodal reasoning, as illustrated in Fig. 2. The first stage, *Visual Dependency Induction* (Sec. 3.3), activates image-grounded reasoning by masking answer-revealing caption cues and introducing plausible image-grounded distractors, forcing the model to resolve the question from visual evidence rather than textual shortcuts and recover the missing fact from the image. The second stage, *Structured Latent Visual Reasoning* (Sec. 3.4), organizes the model's hidden states into a typed latent reasoning flow before answer generation, with functional latent stages for planning, grounding, visual evidence selection and reasoning integration. Together, these two stages first establish visual dependency and then structure it in latent space, preserving the compactness of latent reasoning while avoiding homogeneous, unstructured latent tokens. Their effectiveness is evaluated in Sec 4.4.

3.2 Intuition and Example

Consider the scenario illustrated in Figure 2: the original caption states that the scene contains "one plate with four green apples and two red apples," which directly reveals the answer to the question "How many green apples are on the plate?" VDI removes this answer-revealing cue, rewriting the caption as a partial description (e.g., "one plate with green and red apples"). The question remains unchanged, but the answer options are populated with hard, image-grounded distractors. The correct answer is *4*, while distractors such as *2* and *3* correspond to other visible quantities in the scene (e.g.,

red apples or cups). Unlike random negatives, these distractors are visually plausible yet incorrect under the query, encouraging the model to rely on visual evidence rather than textual shortcuts.

Building on this grounding, SLVR further structures how the model performs reasoning internally. Given the same example, the model processes the input through a sequence of latent stages. In the <plan> stage, it encodes a high-level strategy (e.g., counting green apples on the plate). In the <box> stage, it predicts a spatial region corresponding to the plate. In the <evidence> stage, it aggregates visual features from image patches within this region that correspond to green apples. Finally, in the <reason> and <answer> stages, the model integrates the extracted evidence to produce the final count. Each stage operates in continuous latent space and is supervised with task-specific targets, enabling structured multi-step reasoning without requiring explicit textual intermediate steps.

3.3 Visual Dependency Induction

Formalization. We first introduce *Visual Dependency Induction* (VDI), an initial MCQA training stage that converts coarse image-caption alignment into question-conditioned visual dependency. Each raw instance is an image-caption pair (I, c), where c is broadly aligned with the visual content. We use a caption-rewriting procedure to select an answer-critical visual fact e from c and produce a redacted context c^- by removing only e while preserving the remaining grounded description. A query q and ground-truth answer y are then generated from the removed fact e. We construct a candidate set $\mathcal{A} = \{y\} \cup \mathcal{D}$, where $\mathcal{D}$ contains image-grounded distractors derived from other objects, attributes, counts, or relations present within I.

For each candidate $a \in \mathcal{A}$, the model receives the multimodal prompt tuple (I, c^-, q) as input and generates the candidate text a. We define the compatibility score s_θ as the log-likelihood of generating candidate a given the prompt:

$$s_\theta(I, c^-, q, a) = \log p_\theta(a \mid I, c^-, q). \tag{1}$$

We then train the network using a softmax contrastive objective over these compatibility scores:

$$\mathcal{L}_{\mathrm{VDI}} = -\log \frac{\exp s_\theta(I, c^-, q, y)}{\sum_{a \in \mathcal{A}} \exp s_\theta(I, c^-, q, a)}. \tag{2}$$

By explicitly contrasting the correct answer against visually grounded but question-inconsistent alternatives, VDI compels the model to form representations that are strictly dependent on task-relevant visual evidence. This induction establishes the critical grounding foundation required for the subsequent structured latent reasoning stage.

3.4 Structured Latent Visual Reasoning

Latent reasoning flow. With visual dependency established, SLVR organizes the model's internal computation into a structured latent reasoning trajectory. Rather than compressing a visual chain-of-thought into a monolithic latent vector or an unstructured token sequence, we enforce a human-like cognitive path natively within the latent space. To make this functional structure concrete, we format each training instance by inserting an ordered sequence of typed latent states and transition delimiters prior to the final answer:

$$\begin{aligned} S = [&\langle \text{plan_start} \rangle\ Z_{\text{plan}}\ \langle \text{plan_end} \rangle,\ \langle \text{box_start} \rangle\ Z_{\text{box}}\ \langle \text{box_end} \rangle, \\ &\langle \text{evi_start} \rangle\ Z_{\text{evi}}\ \langle \text{evi_end} \rangle,\ \langle \text{full_start} \rangle\ Z_{\text{full}}\ \langle \text{full_end} \rangle,\ \langle \text{ans} \rangle\ y]. \end{aligned} \tag{3}$$

The special delimiter tokens are not merely formatting markers. They define the boundaries and types of latent states, and they are trained with next-token prediction so that the model learns an ordered transition structure: planning $\rightarrow$ grounding $\rightarrow$ evidence selection $\rightarrow$ reasoning integration $\rightarrow$ answer decoding. In contrast, the latent state tokens themselves are masked from the standard language modeling loss and are supervised through state-specific alignment objectives.

State roles. The latent states in Eq. 3 serve as internal reasoning states with distinct semantic roles. The planning state Z_{plan} represents *what to reason about*, such as whether the question requires counting, attribute recognition, spatial comparison, or object presence verification. The grounding state Z_{box} represents *where to look* by predicting a target region in normalized bounding-box

coordinates. The evidence state Z_{evi} represents *what visual evidence to use* by carrying answer-relevant visual patch information from the grounded region. Finally, the integration state Z_{full} represents *how to integrate the reasoning process*, serving as a compact latent substitute for the full visual rationale before the model produces the short answer y.

This design separates two complementary learning signals: delimiter tokens teach the model the discrete reasoning-state transition, while latent state tokens learn the continuous content of each reasoning state. As a result, SLVR does not rely on a homogeneous block of latent computation, but instead induces a structured latent trajectory with explicit stage boundaries.

Here, "human-like reasoning flow" refers to a functional decomposition within one reasoning iteration, rather than a model of the full recursive reasoning process of an agent. The current model follows a fixed stage order and does not revisit earlier stages within that iteration. Repeated calls with new observations could support an iterative agent, but such interaction is outside our evaluation. The implementation names `plan`, `box`, `evi`, and `full` correspond to planning, grounding, evidence selection, and reasoning integration, respectively.

State-Specific Latent Alignment. We optimize the continuous latent states through a unified, state-specific alignment objective. The core intuition is to constrain each latent state to approximate a supervision target that strictly corresponds to its functional role in the reasoning trajectory. For example, the planning state is aligned with the textual embedding of an action plan, the grounding state with spatial bounding-box coordinates, the evidence state with relevant visual patch features, and the integration state with the embedding of the complete visual rationale.

Formally, let m denote a latent state type, with

$$\mathcal{M} = \{\text{plan}, \text{box}, \text{evi}, \text{full}\}.$$

For each type m, H_m denotes the hidden states at the corresponding latent positions, P_m is a type-specific projection head, and T_m is the supervision target for that type. We define the latent alignment loss as:

$$\begin{aligned}\mathcal{L}_{\text{SLA}} &= \sum_{m \in \mathcal{M}} \lambda_m D_m \left(P_m(H_m), T_m\right) \\ &= \lambda_{\text{plan}} \left\|P_{\text{plan}}(H_{\text{plan}}) - T_{\text{plan}}\right\|_2^2 + \lambda_{\text{box}} \left\|P_{\text{box}}(H_{\text{box}}) - T_{\text{box}}\right\|_2^2 \\ &\quad + \lambda_{\text{evi}} \left\|P_{\text{evi}}(H_{\text{evi}}) - T_{\text{evi}}\right\|_2^2 + \lambda_{\text{full}} \left\|P_{\text{full}}(H_{\text{full}}) - T_{\text{full}}\right\|_2^2 .\end{aligned} \tag{4}$$

While all terms optimize a standard ℓ_2 distance, they operate across distinct representational spaces to enforce functional modularity. The **planning** latent tokens are aligned with the embedding of the textual action plan, encouraging them to encode *what to reason about*. The **grounding** latent tokens are aligned with the normalized bounding box $b = (c_x, c_y, w, h)$, encouraging them to encode *where to look*. Specifically, we allocate three latent tokens for this state: the first predicts the object center (c_x, c_y), while the second and third predict the boundary (width w and height h, respectively). The **evidence** latent tokens are aligned with selected, query-relevant visual patch features from the grounded region, encouraging them to encode *which visual evidence to use*. Finally, the **integration** latent tokens are aligned with the embedding of the complete visual rationale, encouraging them to synthesize *how to logically deduce* the final answer.

The complete rationale is tokenized into token-level target embeddings. For the 16-position integration target, shorter sequences are padded and padded positions are masked out of the alignment loss; longer sequences are truncated. The integration budget approximately matches the 90th percentile of rationale target lengths. Training target lengths depend on the instance, whereas the default inference rollout uses fixed budgets. Appendix A.2 reports a dynamic alternative and held-out target-recovery measurements.

Evidence Target Generation. The evidence state Z_{evi} is designed to refine coarse spatial grounding into precise, answer-relevant visual features. However, because a ground-truth bounding box may still encompass background patches, irrelevant object parts, or distractor details, we must construct a refined supervision target for Z_{evi} during training.

We achieve this by utilizing the final answer to heuristically select the most relevant visual patches. Let $\mathcal{B}$ denote the set of visual patch indices overlapping with the ground-truth box, and let $v_i \in \mathbb{R}^d$ be the visual feature of patch i. During training, we encode the full answer y into an answer query vector $a \in \mathbb{R}^d$. We then compute a dot-product relevance score between each candidate patch and the

answer query:

$$s_i = v_i^\top a, \quad i \in \mathcal{B}. \tag{5}$$

In practice, we normalize both vectors before scoring, yielding a normalized dot product:

$$s_i = \left(\frac{v_i}{\|v_i\|_2}\right)^\top \left(\frac{a}{\|a\|_2}\right), \quad i \in \mathcal{B}. \tag{6}$$

We select patches whose semantic relevance score s_i exceeds a predefined threshold τ_{evi} to form the evidence supervision target. To prevent empty or overly sparse evidence sets, we enforce a minimum patch count $K_{\min}$ by falling back to the top-ranked candidates if necessary:

$$\mathcal{P}_{\text{evi}} = \begin{cases} \{\, i \in \mathcal{B} \mid s_i \geq \tau_{\text{evi}} \,\}, & \text{if } |\{\, i \in \mathcal{B} \mid s_i \geq \tau_{\text{evi}} \,\}| \geq K_{\min}, \\ \text{TopK}_{i \in \mathcal{B}}(s_i, K_{\min}), & \text{otherwise.} \end{cases} \tag{7}$$

$$T_{\text{evi}} = \{\, v_i \mid i \in \mathcal{P}_{\text{evi}} \,\}. \tag{8}$$

It is crucial to emphasize that this dot-product selection mechanism is strictly a **training-time procedure** used to formulate the supervision target T_{evi}. At inference time, the model does not have access to the answer query a. Instead, having been optimized via the alignment loss $\mathcal{L}_{\text{SLA}}$, the network autonomously projects the latent state Z_{evi} to implicitly isolate and attend to the correct query-relevant image patches.

Training Objective. The total training objective combines standard autoregressive supervision, state-specific latent alignment, and answer-transition supervision:

$$\mathcal{L} = \mathcal{L}_{\text{ce}} + \mathcal{L}_{\text{SLA}} + \mathcal{L}_{\text{ans}}. \tag{9}$$

Here, $\mathcal{L}_{\text{SLA}}$ is the state-specific latent alignment objective defined in Eq. 4, which supervises the planning, box-grounding, evidence, and full-answer latent states.

The answer-side objective is defined as:

$$\mathcal{L}_{\text{ans}} = \lambda_{\text{open}} \mathcal{L}_{\text{answer_open}} + \lambda_{\text{short}} \mathcal{L}_{\text{short_answer}}. \tag{10}$$

$\mathcal{L}_{\text{answer_open}}$ teaches the model when to stop internal latent reasoning and transition to the explicit `<answer>` segment, while $\mathcal{L}_{\text{short_answer}}$ supervises the concise final answer produced after this transition.

During training, $\mathcal{L}_{\text{ce}}$ is applied to the discrete transition delimiters and output tokens, while the continuous latent states are optimized through $\mathcal{L}_{\text{SLA}}$ rather than standard next-token prediction.

At inference time, the model operates autonomously without access to any auxiliary targets. Given only the input image and question, the network autoregressively decodes the discrete transition delimiters while internally projecting the corresponding continuous latent states to execute planning, grounding, evidence extraction, and integration. Following this structured internal rollout, the model decodes a concise final answer. Thus, SLVR exploits rich, structured multimodal supervision during training while maintaining a highly efficient, direct-answer interface during deployment.

4 Experiments

4.1 Experimental Setup

Backbone model. We use **Qwen2.5-VL-7B-Instruct** as the backbone model for SLVR. This backbone has been adopted by several recent open-source multimodal reasoning methods [33, 19], which makes it a natural choice for controlled comparison. Our controlled comparisons keep the backbone architecture and initialization fixed. Literature-reported methods use different training data and recipes, so their scores should not be interpreted as isolating the effect of the latent structure. Appendix A.2 provides same-data controls, baseline supervision details, and transfer experiments.

Training data. SLVR is trained with a two-stage data pipeline. For the first stage, we construct a **20K VDI dataset** from COVT [26] visual reasoning data. Each raw instance is an aligned image-text pair (I, c), where c describes concrete visual facts such as objects, attributes, counts, or relations.

Table 1: **Performance on challenging multimodal perception and reasoning benchmarks. V*** and **BLINK** are reported with their fine-grained subsets, while **MMVP** is reported using the standard overall accuracy. The best result for each metric is highlighted in **bold**. The green row reports the absolute improvement of **SLVR** over the **Qwen2.5-VL-7B** base model. Prior-method results are taken from the corresponding papers.

Model	V*			MMVP	BLINK				MathVista	ChartQA
	Overall	Attribute	Spatial		Count	Jigsaw	Relation	IQ		
Qwen2.5-VL-7B	78.5	81.7	73.7	66.7	66.7	52.0	38.8	26.0	63.7	86.2
Monet† [33]	83.6	83.5	82.9	–	–	–	–	–	–	–
LVR† [19]	80.6	81.7	79.0	71.8	70.0	52.5	42.3	28.4	–	–
DeepEyes v2† [15]	81.8	–	–	–	–	–	–	–	71.9	88.4
COVT† [26]	78.5	–	–	56.7	–	–	–	–	–	–
CodeVision† [12]	83.7	–	–	–	–	–	–	–	–	–
LaViT† [36]	–	82.6	–	67.3	–	–	45.5	32.0	–	–
SLVR (Ours)	**85.2**	**85.1**	**84.3**	**76.1**	**72.4**	**62.3**	**53.0**	**34.5**	**73.4**	**89.6**
Improvement over Qwen	+6.7	+3.4	+10.6	+9.4	+5.7	+10.3	+14.2	+8.5	+9.7	+3.4

Table 2: **Ablation studies of SLVR.** Left: incremental ablation of the training pipeline on **MMVP** and **V***. Right: ablation of latent token budgets, where **P**, **E**, and **I** denote the number of latent tokens for planning, evidence selection, and reasoning integration, respectively.

Incremental Components					Latent Token Budget					
Variant	MMVP	V*			Setting	P	E	I	MMVP	V* Overall
		Overall	Attr.	Spatial						
Qwen2.5-VL-7B	66.7	78.5	81.7	73.7	**Short latent**	16	16	16	74.0	83.8
+ VDI-only	71.2	81.9	83.0	80.1	**Long grounding**	64	64	16	74.8	84.3
+ Latent Plan	73.0	83.0	83.7	81.5	**Long integration**	64	16	64	74.5	84.1
+ Latent Grounding	74.2	84.0	84.3	82.9	**Balanced**	32	32	32	75.1	84.5
+ Latent Evidence	75.2	84.6	84.8	83.6	**Ours**	**64**	**16**	**16**	**76.1**	**85.2**
+ Latent Integration / SLVR	**76.1**	**85.2**	**85.1**	**84.3**	*Improvement over Qwen*	–	–	–	+9.4	+6.7

We use an off-the-shelf VLM-assisted rewriting procedure to identify an answer-critical visual fact in c, remove only this fact while preserving the remaining grounded context, and generate the corresponding question, answer, and multiple-choice options. The resulting redacted context remains coherent but no longer reveals the answer, forcing the model to recover the missing fact from the image. Distractor options are visually plausible but incorrect under the generated query. This procedure is used only for training-data construction, not at inference time, and does not provide teacher predictions for evaluation examples. We manually verify a subset of the constructed examples and filter cases with ambiguous evidence, incorrect masking, answer leakage, or non-unique answers. More details are provided in Appendix A.5.

For the second stage, we train on Visual-CoT [28] to learn structured latent visual reasoning. Visual-CoT provides image-question-answer examples with CoT-style plans, region-level grounding annotations, and full reasoning/answer supervision. We use these annotations to supervise the typed latent states for *planning*, *grounding*, *evidence selection*, and *reasoning integration*, while avoiding the need to decode full visible CoT at inference time. More details about the Visual-CoT data format, latent-state construction, and training targets are provided in Appendix A.6.

Evaluation benchmarks. We evaluate SLVR on five challenging multimodal perception and reasoning benchmarks: **V*** [37], **MMVP** [32], **BLINK** [8], **MathVista** [21], and **ChartQA** [22]. For **V***, we report the overall score as well as the attribute and spatial subsets. For **BLINK** [8], we report fine-grained subsets including counting, jigsaw, relation, and IQ-style visual reasoning. For **MMVP** [32], we report the standard overall accuracy. Together, these benchmarks evaluate fine-grained visual perception, spatial reasoning, visual robustness, and multimodal reasoning beyond language-only shortcuts.

4.2 Implementation Details

We initialize SLVR from Qwen2.5-VL-7B-Instruct and train it with a two-stage procedure. All experiments are conducted on 8 NVIDIA A100 GPUs with mixed-precision (bf16) training. During training, we freeze the visual encoder and image processor, and update only the language model parameters. No additional vision encoder or projection module is trained. Images are processed with the default Qwen2.5-VL image processor.

We use a learning rate of 1×10^{-5} for the first-stage VDI training and 5×10^{-6} for the second-stage latent reasoning training. The global batch size is set to 64 with gradient accumulation. We train the first stage for 5 epochs and the second stage for 3 epochs. The maximum sequence length is set to 4096. We use AdamW as the optimizer with a cosine learning-rate schedule and a warmup ratio of 0.03. The weight decay is set to 0.0.

In the inference stage, we set the number of latent tokens for planning, evidence modeling, and answer integration to 64, 16, and 16, respectively. Grounding uses a fixed 3 tokens as defined in Sec 3.4. For evidence patch selection, we use an evidence threshold of $\tau_{\text{evi}} = 0.5$ and keep at least $K_{\min} = 5$ patches by falling back to the top-ranked candidates when necessary. For the overall training objective, we use fixed empirical weights:

$$\mathcal{L} = \mathcal{L}_{\text{ce}} + 0.1\mathcal{L}_{\text{plan}} + 1.0\mathcal{L}_{\text{box}} + 1.0\mathcal{L}_{\text{evi}} + 0.5\mathcal{L}_{\text{full}} + 2.0\mathcal{L}_{\text{answer_open}} + 3.0\mathcal{L}_{\text{short_answer}}. \quad (11)$$

These hyperparameter choices are analyzed in Appendix A.3.

The latent budgets reflect the complexity and length of the corresponding supervision targets and are checked using the token-budget ablation. Planning receives more capacity for natural-language plans; grounding, evidence, and integration use more compact targets. Loss weights reflect each target's role and reliability, with the weighting alternatives analyzed in Appendix A.3.

4.3 Main Results

Table 1 reports the main comparison on five multimodal perception and reasoning benchmarks. Compared with the Qwen2.5-VL-7B base model, SLVR improves consistently across all reported metrics. On **V*** [37], SLVR improves Overall, Attribute, and Spatial by +6.7, +3.4, and +10.6, respectively. On **MMVP** [32], SLVR achieves a substantial gain of +9.4, suggesting that visual dependency induction helps reduce reliance on language-prior shortcuts and encourages stronger image-grounded reasoning.

SLVR also improves all fine-grained **BLINK** [8] subsets, with gains of +5.7 on Counting, +10.3 on Jigsaw, +14.2 on Relation, and +8.5 on IQ. The strong improvements on Spatial and Relation-oriented subsets indicate that typed latent grounding and evidence selection are particularly beneficial for fine-grained visual reasoning. Beyond natural-image perception, SLVR further improves **MathVista** [21] and **ChartQA** [22] by +9.7 and +3.4, respectively, showing that the learned latent reasoning structure also transfers to broader multimodal reasoning tasks.

4.4 Ablation Study

Table 2 presents two ablation analyses. On the left, we progressively add each component on top of Qwen2.5-VL-7B. The first stage already brings clear gains, improving MMVP [32] from 66.7 to 71.2 and V* [37] Overall from 78.5 to 81.9. Adding the typed latent stages further improves performance step by step, with the full SLVR model reaching 76.1 on MMVP [32] and 85.2 on V* [37] Overall. These results show that visual dependency induction and structured latent reasoning are complementary.

On the right of Table 2, we study the effect of latent token budgets. We denote the number of latent tokens for planning, evidence selection, and reasoning integration as P, E, and I, respectively. As shown, SLVR performs optimally with an asymmetric token distribution ($P = 64$, $E = 16$, $I = 16$). This distribution mirrors the informational density of our supervision targets. The planning stage is supervised by natural language CoT rationales, which have a mean length of 65.11 tokens (median 62). Allocating 64 latent tokens provides a nearly 1:1 representational capacity to align with these complex syntactic structures. In contrast, once the global strategy is formulated, the downstream targets—such as low-dimensional spatial coordinates for grounding and highly localized patch features for evidence—require significantly less capacity. Consequently, these later functional states can be aggressively compressed to 16 tokens without performance degradation, maximizing inference efficiency while ensuring both the structure and allocation of latent states are fully optimized.

Additional controlled evaluations. With the same two-stage data, ordinary SFT reaches 71.8 on MMVP and 82.6 on V* Overall, and an unstructured latent control reaches 72.5 and 81.9, compared with SLVR's 76.1 and 85.2. These controls support a contribution from typed latent organization beyond the additional supervision. Appendix A.2 reports these comparisons, target recovery, efficiency, and transfer results.

Robustness to misleading textual context. We additionally evaluate grounding robustness under misleading captions. Specifically, we sample non-overlapping COVT [26] examples and use a VLM-assisted rewriting procedure to deliberately replace the answer-critical visual fact in the caption with an incorrect but plausible alternative. As reported in Appendix A.1, SLVR achieves higher misleading-context accuracy and lower caption-following error than Qwen2.5-VL-7B, VDI-only, and LVR [19], suggesting that the proposed structured latent flow better recovers answer-relevant evidence from the image when text and vision conflict.

5 Conclusion and Limitations

We introduce **SLVR**, a framework that enables MLLMs to perform structured visual reasoning in the latent space. Our visual dependency induction stage constructs 20K COVT-rewritten masked-text examples to reduce reliance on textual shortcuts, while our structured latent reasoning stage uses Visual-CoT supervision to organize internal reasoning into planning, grounding, evidence selection, and integration. Experiments show that SLVR consistently improves over Qwen2.5-VL-7B on multimodal benchmarks, with strong gains on fine-grained spatial and relational reasoning. Together, these results demonstrate SLVR as a promising direction toward more grounded and reliable multimodal reasoning.

We summarize the limitations of our work as follows. First, SLVR relies on a two-stage training pipeline, which increases data construction and training complexity. Second, the fixed stage order does not support backtracking within one reasoning iteration; dynamic token allocation is feasible but slightly underperforms the fixed budget in our evaluation. Third, our main evaluation uses Qwen2.5-VL-7B; additional transfer experiments cover Qwen2.5-VL-3B and Gemma-4-E4B, while broader scaling remains to be studied.

Acknowledgements and Disclosures

We thank Qianqian Wan at Harvard University for helpful discussions and suggestions on the ideas underlying this work. This work used NVIDIA GH200 computing resources on the DeltaAI system at the National Center for Supercomputing Applications through the Advanced Cyberinfrastructure Coordination Ecosystem: Services & Support (ACCESS) program. DeltaAI is supported by the National Science Foundation (award OAC-2320345) and the State of Illinois, and is a joint effort of the University of Illinois Urbana-Champaign and its National Center for Supercomputing Applications. ACCESS is supported by National Science Foundation grants 2138259, 2138286, 2138307, 2137603, and 2138296. The authors report no other funding or third-party support for this work and declare no competing interests.

References

[1] Shuai Bai, Keqin Chen, Xuejing Liu, et al. Qwen2.5-vl technical report. *arXiv preprint arXiv:2502.13923*, 2025.

[2] Chao Chen, Zhixin Ma, Yongqi Li, Yupeng Hu, Yinwei Wei, Wenjie Li, and Liqiang Nie. Reasoning in the dark: Interleaved vision-text reasoning in latent space. *arXiv preprint arXiv:2510.12603*, 2025.

[3] Xinyan Chen, Renrui Zhang, Dongzhi Jiang, Aojun Zhou, Shilin Yan, Weifeng Lin, and Hongsheng Li. MINT-cot: Enabling interleaved visual tokens in mathematical chain-of-thought reasoning. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2026.

[4] Ethan Chern, Zhulin Hu, Steffi Chern, Siqi Kou, Jiadi Su, Yan Ma, Zhijie Deng, and Pengfei Liu. Thinking with generated images. *arXiv preprint arXiv:2505.22525*, 2025.

[5] Yunlong Chu, Minglai Shao, Yuhang Liu, Bing Hao, Yumeng Lin, Jialu Wang, and Ruijie Wang. Spot: Span-level pause-of-thought for efficient and interpretable latent reasoning in large language models. *arXiv preprint arXiv:2603.06222*, 2026.

[6] Xuanming Cui, Jianpeng Cheng, Hong-You Chen, Satya Narayan Shukla, Abhijeet Awasthi, Xichen Pan, Chaitanya Ahuja, Shlok Kumar Mishra, Taipeng Tian, Qi Guo, Ser-Nam Lim, Aashu Singh, and Xiangjun Fan. Think then embed: Generative context improves multimodal embedding. In *The Fourteenth International Conference on Learning Representations*, 2026. URL `https://openreview.net/forum?id=AKXXMK5YTI`.

[7] Haiwen Feng, Long Lian, Lisa Dunlap, Jiahao Shu, XuDong Wang, Renhao Wang, Trevor Darrell, Alane Suhr, and Angjoo Kanazawa. Visually prompted benchmarks are surprisingly fragile. *arXiv preprint arXiv:2512.17875*, 2025.

[8] Xingyu Fu, Yushi Hu, Bangzheng Li, Yu Feng, Haoyu Wang, Xudong Lin, Dan Roth, Noah A Smith, Wei-Chiu Ma, and Ranjay Krishna. Blink: Multimodal large language models can see but not perceive. In *European Conference on Computer Vision*, pages 148–166. Springer, 2024.

[9] Gemini Team. Gemini 1.5: Unlocking multimodal understanding across millions of tokens of context. *arXiv preprint arXiv:2403.05530*, 2024.

[10] Sachin Goyal, Ziwei Ji, Ankit Singh Rawat, Aditya Krishna Menon, Sanjiv Kumar, and Vaishnavh Nagarajan. Think before you speak: Training language models with pause tokens. In *The Twelfth International Conference on Learning Representations*, 2024.

[11] Jiawei Gu, Yunzhuo Hao, Huichen Will Wang, Linjie Li, Michael Qizhe Shieh, Yejin Choi, Ranjay Krishna, and Yu Cheng. Thinkmorph: Emergent properties in multimodal interleaved chain-of-thought reasoning. In *The Fourteenth International Conference on Learning Representations*, 2026.

[12] Zirun Guo, Minjie Hong, Feng Zhang, Kai Jia, and Tao Jin. Thinking with programming vision: Towards a unified view for thinking with images. *arXiv preprint arXiv:2512.03746*, 2025.

[13] Tanmay Gupta and Aniruddha Kembhavi. Visual programming: Compositional visual reasoning without training. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 14953–14962, 2023.

[14] 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.

[15] Jack Hong, Chenxiao Zhao, Chenglin Zhu, Weiheng Lu, Guohai Xu, and XingYu. Deepeyesv2: Toward agentic multimodal model. In *The Fourteenth International Conference on Learning Representations*, 2026.

[16] Wenyi Hong et al. Glm-4.5v and glm-4.1v-thinking: Towards versatile multimodal understanding and reasoning. *arXiv preprint arXiv:2507.01006*, 2025.

[17] Yushi Hu, Weijia Shi, Xingyu Fu, Dan Roth, Mari Ostendorf, Luke Zettlemoyer, Noah A. Smith, and Ranjay Krishna. Visual sketchpad: Sketching as a visual chain of thought for multimodal language models. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*, 2024.

[18] Zixi Jia, Jiqiang Liu, Hexiao Li, Qinghua Liu, and Hongbin Gao. Dcot: Dual chain-of-thought prompting for large multimodal models. In *The 16th Asian Conference on Machine Learning (Conference Track)*, 2024.

[19] Bangzheng Li, Ximeng Sun, Jiang Liu, Ze Wang, Jialian Wu, Xiaodong Yu, Emad Barsoum, Muhao Chen, and Zicheng Liu. Latent visual reasoning. In *The Fourteenth International Conference on Learning Representations*, 2026.

[20] Chengzhi Liu, Yuzhe Yang, Yue Fan, Qingyue Wei, Sheng Liu, and Xin Eric Wang. Reasoning within the mind: Dynamic multimodal interleaving in latent space. *arXiv preprint arXiv:2512.12623*, 2025.

[21] Pan Lu, Hritik Bansal, Tony Xia, Jiacheng Liu, Chunyuan Li, Hannaneh Hajishirzi, Hao Cheng, Kai-Wei Chang, Michel Galley, and Jianfeng Gao. Mathvista: Evaluating mathematical reasoning of foundation models in visual contexts. In *International Conference on Learning Representations (ICLR)*, 2024.

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

[23] Chancharik Mitra, Brandon Huang, Trevor Darrell, and Roei Herzig. Compositional chain-of-thought prompting for large multimodal models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 14420–14431, 2024.

[24] OpenAI. Hello gpt-4o. `https://openai.com/index/hello-gpt-4o/`, 2024. Accessed: 2026-05-03.

[25] Tan-Hanh Pham and Chris Ngo. Multimodal chain of continuous thought for latent-space reasoning in vision-language models. *arXiv preprint arXiv:2508.12587*, 2025.

[26] Yiming Qin, Bomin Wei, Jiaxin Ge, Konstantinos Kallidromitis, Stephanie Fu, Trevor Darrell, and Xudong Wang. Chain-of-visual-thought: Teaching vlms to see and think better with continuous visual tokens. *arXiv preprint arXiv:2511.19418*, 2025.

[27] Arijit Ray, Ahmed Abdelkader, Chengzhi Mao, Bryan A Plummer, Kate Saenko, Ranjay Krishna, Leonidas Guibas, and Wen-Sheng Chu. Mull-tokens: Modality-agnostic latent thinking. *arXiv preprint arXiv:2512.10941*, 2025.

[28] Hao Shao, Shengju Qian, Han Xiao, Guanglu Song, Zhuofan Zong, Letian Wang, Yu Liu, and Hongsheng Li. Visual cot: Advancing multi-modal language models with a comprehensive dataset and benchmark for chain-of-thought reasoning. In *The Thirty-eight Conference on Neural Information Processing Systems Datasets and Benchmarks Track*, 2024.

[29] Zhaochen Su, Peng Xia, Hangyu Guo, Zhenhua Liu, Yan Ma, Xiaoye Qu, Jiaqi Liu, Yanshu Li, Kaide Zeng, Zhengyuan Yang, et al. Thinking with images for multimodal reasoning: Foundations, methods, and future frontiers. *arXiv preprint arXiv:2506.23918*, 2025.

[30] Guohao Sun, Hang Hua, Jian Wang, Jiebo Luo, Sohail Dianat, Majid Rabbani, Raghuveer Rao, and Zhiqiang Tao. Latent chain-of-thought for visual reasoning. *arXiv preprint arXiv:2510.23925*, 2025.

[31] Dídac Surís, Sachit Menon, and Carl Vondrick. Vipergpt: Visual inference via python execution for reasoning. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 11888–11898, 2023.

[32] Shengbang Tong, Zhuang Liu, Yuexiang Zhai, Yi Ma, Yann LeCun, and Saining Xie. Eyes wide shut? exploring the visual shortcomings of multimodal llms. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 9568–9578, 2024.

[33] Qixun Wang, Yang Shi, Yifei Wang, Yuanxing Zhang, Pengfei Wan, Kun Gai, Xianghua Ying, and Yisen Wang. Monet: Reasoning in latent visual space beyond images and language. In *CVPR*, 2026.

[34] 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*, 2025.

[35] Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Fei Xia, Ed Chi, Quoc V Le, Denny Zhou, et al. Chain-of-thought prompting elicits reasoning in large language models. *Advances in neural information processing systems*, 35:24824–24837, 2022.

[36] Linquan Wu, Tianxiang Jiang, Yifei Dong, Haoyu Yang, Fengji Zhang, Shichaang Meng, Ai Xuan, Linqi Song, and Jacky Keung. Lavit: Aligning latent visual thoughts for multi-modal reasoning. *arXiv preprint arXiv:2601.10129*, 2026.

[37] Penghao Wu and Saining Xie. V?: Guided visual search as a core mechanism in multimodal llms. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 13084–13094, 2024.

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

[39] Eric Zelikman, Georges Raif Harik, Yijia Shao, Varuna Jayasiri, Nick Haber, and Noah Goodman. Quiet-STar: Language models can teach themselves to think before speaking. In *First Conference on Language Modeling*, 2024.

[40] Huanyu Zhang, Wenshan Wu, Chengzu Li, Ning Shang, Yan Xia, Yangyu Huang, Yifan Zhang, Li Dong, Zhang Zhang, Liang Wang, Tieniu Tan, and Furu Wei. Latent sketchpad: Sketching visual thoughts to elicit multimodal reasoning in mllms. *arXiv preprint arXiv:2510.24514*, 2025.

[41] Yiming Zhang, Qiangyu Yan, Borui Jiang, and Kai Han. Multimodal latent reasoning via hierarchical visual cues injection. *arXiv preprint arXiv:2602.05359*, 2026.

[42] Zhen Zhang, Xuehai He, Weixiang Yan, Ao Shen, Chenyang Zhao, and Xin Eric Wang. Soft thinking: Unlocking the reasoning potential of LLMs in continuous concept space. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2026.

[43] Zhuosheng Zhang, Aston Zhang, Mu Li, Hai Zhao, George Karypis, and Alex Smola. Multimodal chain-of-thought reasoning in language models. *arXiv preprint arXiv:2302.00923*, 2023.

[44] Ge Zheng, Bin Yang, Jiajin Tang, Hong-Yu Zhou, and Sibei Yang. Ddcot: Duty-distinct chain-of-thought prompting for multimodal reasoning in language models. *Advances in Neural Information Processing Systems*, 36:5168–5191, 2023.

[45] Ziwei Zheng, Michael Yang, Jack Hong, Chenxiao Zhao, Guohai Xu, Le Yang, Chao Shen, and XingYu. Deepeyes: Incentivizing "thinking with images" via reinforcement learning. In *The Fourteenth International Conference on Learning Representations*, 2026.

[46] Qiji Zhou, Ruochen Zhou, Zike Hu, Panzhong Lu, Siyang Gao, and Yue Zhang. Image-of-thought prompting for visual reasoning refinement in multimodal large language models. *arXiv preprint arXiv:2405.13872*, 2024.

[47] Rui-Jie Zhu, Tianhao Peng, Tianhao Cheng, Xingwei Qu, Jinfa Huang, Dawei Zhu, Hao Wang, Kaiwen Xue, Xuanliang Zhang, Yong Shan, Tianle Cai, Taylor Kergan, Assel Kembay, Andrew Smith, Chenghua Lin, Binh Nguyen, Yuqi Pan, Yuhong Chou, Zefan Cai, Zhenhe Wu, Yongchi Zhao, Tianyu Liu, Jian Yang, Wangchunshu Zhou, Chujie Zheng, Chongxuan Li, Yuyin Zhou, Zhoujun Li, Zhaoxiang Zhang, Jiaheng Liu, Ge Zhang, Wenhao Huang, and Jason Eshraghian. A survey on latent reasoning, 2025.

Table 3: **Diagnostic evaluation under misleading captions.** We evaluate 300 non-overlapping COVT examples where the answer-critical visual fact in the textual context is deliberately replaced with an incorrect but plausible alternative. The image and question remain unchanged. A grounded model should recover the correct answer from the image rather than follow the misleading caption.

Method	Clean Acc. ↑	Misleading Acc. ↑	Caption-following Err. ↓
Qwen2.5-VL-7B	72.3	41.7	46.0
VDI-only	78.0	58.3	27.7
LVR [19]	80.7	62.0	23.3
SLVR	**84.3**	**70.7**	**14.7**

A Technical appendices and supplementary material

A.1 Diagnostic Evaluation under Misleading Captions

To further verify whether SLVR relies on visual evidence rather than textual shortcuts, we construct a diagnostic evaluation set with misleading textual contexts. We sample 300 image-text examples from COVT [26] that do not overlap with our training set. For each example, we use an off-the-shelf VLM-assisted rewriting procedure to identify the answer-critical visual fact in the textual context and deliberately replace it with an incorrect but plausible alternative. The image and question are kept unchanged, so the correct answer is still determined by the image rather than the modified caption. For instance, if the image contains four green apples, the misleading context may state that there are two green apples. This creates a controlled conflict between the textual context and the visual evidence.

We evaluate four variants: the Qwen2.5-VL-7B base model, the VDI-only model, LVR [19], and the full SLVR model. This comparison tests whether each model can recover the correct answer from the image when the caption provides a plausible but incorrect shortcut. We report two metrics. *Misleading accuracy* measures the exact-match accuracy under the perturbed textual context. *Caption-following error* measures the percentage of examples where the model predicts the incorrect answer implied by the misleading caption. A lower caption-following error indicates stronger resistance to textual shortcuts.

A.2 Additional Controlled Evaluations and Transfer

Same-data controls. Ordinary SFT uses the same 20K VDI examples followed by Visual-CoT, generating textual rationales and answers autoregressively without latent alignment objectives. We match the backbone, optimization settings, and overall training procedure as closely as possible. The unstructured latent control retains the same supervision content, target order, and latent-token budget as SLVR. Its plan, grounding, evidence, and rationale targets are mapped to a shared representation space and concatenated into one target sequence, supervised with a single MSE objective without typed boundaries or stage-specific objectives.

Table 4: Controlled comparisons on Qwen2.5-VL-7B. Both ordinary SFT and the latent variants use the same two-stage training data.

Method	MMVP	V* Overall
Base	66.7	78.5
VDI-only	71.2	81.9
Ordinary SFT	71.8	82.6
Unstructured latent	72.5	81.9
SLVR	**76.1**	**85.2**

Held-out target recovery. We evaluate the existing stage-specific projection heads on a held-out Visual-CoT split. For embedding targets, we compute token-wise cosine similarity after length alignment and average over valid latent positions and examples. Grounding is evaluated by IoU between recovered and ground-truth boxes. Planning, evidence, and integration cosine similarities are

0.913, 0.842, and 0.891, respectively; grounding IoU is 0.684. These measurements support recovery of the supervised targets, without establishing that the latent states reproduce human cognitive processes.

Dynamic inference budgets. A lightweight predictor pools the plan latent states to predict the instance-specific numbers of evidence and integration tokens. Its training labels are the selected evidence-patch count and tokenized full-answer length. At inference, the predicted integer lengths determine downstream token counts. Grounding remains fixed at three tokens in both variants. The

Table 5: Fixed versus plan-conditioned dynamic inference allocation. E and I denote average evidence and integration token counts.

Allocation	MMVP	V* Overall	E	I
Fixed (64/3/16/16)	76.1	85.2	16.0	16.0
Plan-conditioned dynamic	75.5	84.7	13.2	10.8

dynamic variant uses fewer downstream tokens but slightly reduces accuracy. We retain the fixed budget for the main evaluation; adaptive computation remains a direction for further study.

Explicit textual CoT and inference efficiency. For the explicit CoT SFT comparator, we use the published LVR [19] accuracy for Qwen2.5-VL-7B trained on 438K Visual-CoT examples and additionally benchmark inference efficiency. This is distinct from our same-two-stage-data ordinary SFT control. Explicit CoT SFT scores 65.7 on MMVP and 79.1 on V* Overall, compared with SLVR's 76.1 and 85.2. Across samples from these benchmarks, observed end-to-end latency ranges are approximately 5.0–7.0 seconds for explicit CoT and 3.5–4.5 seconds for SLVR. Relative inference FLOPs are 1.00 for explicit CoT and approximately 0.60–0.75 for SLVR. Measurements use the same GPU, BF16, batch size one, greedy decoding, image resolution, and decoding configuration, and include image encoding, prompt prefill, intermediate reasoning, and final-answer generation. These ranges reflect variation across benchmark samples rather than confidence intervals.

Transfer across scales and families. We apply the same training framework to Qwen2.5-VL-3B and Gemma-4-E4B. Improvements on both models indicate transfer beyond the main 7B backbone, although the evaluation remains limited to two benchmarks.

Table 6: Transfer of SLVR across backbone models.

Backbone	Method	MMVP	V* Overall
Qwen2.5-VL-3B	Base	64.7	75.4
	SLVR	71.7	81.0
Gemma-4-E4B	Base	64.0	37.7
	SLVR	70.8	44.9
Qwen2.5-VL-7B	Base	66.7	78.5
	SLVR	76.1	85.2

Provenance of literature baselines. The Table 1 comparators use Qwen2.5-VL-7B, but their training supervision differs. LVR [19] uses Visual-CoT region-grounded data and additional VQA prompts for latent RL. Monet [33] uses Monet-SFT-125K (constructed from Visual-CoT, ReFocus, CoCoM, and Zebra-CoT) and 3.2K Thyme-RL examples. CoVT [26] uses approximately 1.17M multimodal QA examples with visual-expert supervision from LLaVA-OneVision, TallyQA, and ADE20K-Depth. The DeepEyesV2 comparator described in the additional comparison uses curated multimodal reasoning and tool-use trajectories, with its training scale not explicitly reported in the comparison. CodeVision [12] uses 35K visual-programming RL trajectories; its SFT scale is not explicitly reported. LaViT [36] uses LaViT-SFT-15K, distilled from a teacher for visual semantics and attention trajectories. These literature-reported results provide context rather than a same-data causal comparison. The base model and SLVR are evaluated under our setting.

Table 7: **Hyperparameter analysis of SLVR.** Left: sensitivity to the evidence threshold τ_{evi} for patch selection. When $\tau_{\text{evi}} = 0$, the selection degenerates to using nearly all candidate patches, introducing noisy visual evidence. If fewer than $K_{\min}$ patches satisfy the threshold, we fall back to the top-ranked candidates. Right: ablation on auxiliary loss weighting strategies. All variants are evaluated on **MMVP** and **V* Overall**.

Evidence Threshold τ_{evi}			**Auxiliary Loss Weighting**		
Threshold	**MMVP**	**V* Overall**	**Weight Setting**	**MMVP**	**V* Overall**
0.0	73.6	83.9	Uniform weights	74.8	84.3
0.3	75.0	84.4	w/o up-weighted answer	75.2	84.6
0.5	**76.1**	**85.2**	w/o down-weighted plan loss	75.0	84.5
0.7	75.4	84.6	**Ours**	**76.1**	**85.2**

Table 8: **High-level comparison with representative latent multimodal reasoning methods.** Abbreviations: **Latent Sup.** = continuous latent states with explicit latent supervision; **Visual Feat.** = explicit use, selection, injection, or reconstruction of visual features during latent reasoning; **Grounding Sup.** = explicit region-level grounding supervision, such as boxes or ROI targets; **Typed** = typed plan/grounding/evidence/integration stages; **Ans. Evi. Sup.** = answer-specific evidence supervision. The columns distinguish design choices rather than overall capability.

Method	**Latent Sup.**	**Visual Feat.**	**Grounding Sup.**	**Typed**	**Ans. Evi. Sup.**
Multimodal Coconut [25]	✓	–	–	–	–
LVR [19]	✓	✓	✓	–	✓
IVT-LR [2]	✓	✓	–	–	–
LaCoT [30]	✓	✓	–	–	–
COVT [26]	✓	✓	✓	–	✓
Latent Sketchpad [40]	✓	✓	–	–	–
MoNet [33]	✓	✓	–	–	–
LaViT [36]	✓	✓	–	–	–
SLVR (Ours)	✓	✓	✓	✓	✓

A.3 Hyperparameter Analysis

Table 7 analyzes two key hyperparameter choices. For evidence selection, setting $\tau_{\text{evi}} = 0$ keeps nearly all candidate patches and introduces noisy visual evidence, while a high threshold may discard useful patches and more frequently trigger the top-ranked fallback. We use $\tau_{\text{evi}} = 0.5$ and $K_{\min} = 5$ in all experiments, which provides the best balance between evidence recall and precision.

For auxiliary loss weighting, we weight each objective according to its role in the reasoning pipeline. We down-weight the plan loss since high-level reasoning plans are semantically flexible and should not over-constrain the model. Grounding and patch losses are kept at standard weight because they provide concrete visual supervision. The answer embedding loss is weighted moderately to align latent states with textual answers without dominating the language modeling objective. In contrast, we up-weight the answer-opening and short-answer losses, since they respectively control the transition from latent reasoning to the explicit `<answer>` segment and the correctness of the final prediction. The lower performance of uniform weighting and removing these tailored weights shows that such role-aware balancing is important for SLVR.

A.4 Comparison with Latent Multimodal Reasoning Methods

Table 8 summarizes how SLVR differs from representative latent multimodal reasoning methods. We emphasize that several prior methods introduce meaningful forms of structure, including continuous thought tokens, reconstructed visual tokens, interleaved vision-text states, latent visual embeddings, and visual scratchpads. Our comparison focuses on whether a method explicitly decomposes latent reasoning into the same functional stages used by SLVR: planning, grounding, evidence selection, and reasoning integration.

A.5 Details of Visual Dependency Induction

This appendix shows the data format used in the Visual Dependency Induction stage. The purpose of this stage is to create image-conditioned multiple-choice examples where the rewritten text remains aligned with the image but does not directly contain the answer. This forces the model to recover the missing visual fact from the image rather than relying on textual shortcuts or answer priors.

Filtering statistics and criteria. Approximately 20K of 46.5K generated candidates are retained (about 43% retention and 57% rejection). We manually inspect approximately 1K samples and balance the retained data across object presence, color, counting, spatial relations, and physical/structural attributes. Automatic filtering checks answer leakage, ambiguity, non-unique answers, insufficient image dependence, and the correctness and consistency of the rewritten question-answer pair with the image.

Additional retained example. An original micrograph caption states that a very large dark area occupies much of the right side. The rewrite removes that location while preserving the grayscale micrograph description, irregular dark regions, bright interconnected network, and a separate upper-central dark cluster. The question asks which side contains the dominant large dark region, with options right, left, neither, and unknown; the answer is right. This example is retained because the rewrite does not reveal the requested left-versus-right fact and the answer is visually unambiguous.

Rejected example. An original caption describes a wide sandy beach with a pale yellow appearance. Removing "pale yellow appearance" leaves "sandy beach" in the rewrite. A question asking the beach color, with yellow, blue, green, and unknown as options, is rejected: "sandy beach" still permits a language-prior guess of yellow, and plausible descriptions such as yellow, tan, beige, or sandy-colored make the intended answer insufficiently unique. This illustrates the image-dependence and answer-uniqueness filters.

Visual Dependency Induction data format. Each training instance in the Visual Dependency Induction stage is organized as an image-conditioned multiple-choice example. As shown in Figure 3, each example contains an image, an original caption, a masked target, a rewritten caption, a question, a set of answer options, and the correct answer. The original caption provides a complete image-grounded description and may explicitly mention the target visual fact. The masked target specifies the visual evidence to be removed from the text. The rewritten caption preserves the remaining image-caption context but removes this target fact, so that the answer is no longer available from text alone. The question is then written to query the removed target fact, and the options contain one correct answer together with visually plausible distractors. Therefore, during training, the model must condition on both the rewritten caption and the image to recover the correct answer.

Example. Figure 3 shows one Visual Dependency Induction training instance. The original caption explicitly mentions the target object, "a yellow umbrella." The rewritten caption removes this target while preserving the other scene information, including the blue construction wall, white signs, pedestrian signal, shopping cart, and people on the sidewalk. The resulting question asks the model to identify the object visible at the far right above the man in dark blue clothing. Since the rewritten caption no longer contains the target object, the answer can only be recovered by looking at the image.

Instance fields. The example in Figure 3 contains the following fields.

Original caption. A bright blue construction wall fills the background, with two white rectangular signs attached to it, a gray pedestrian signal mounted above, an orange metal shopping cart piled with clothes, and three people on the sidewalk. The woman in a green-and-black outfit is to the left of the cart, the bearded man in a light shirt stands next to the cart, and the man in dark blue clothing is to the right near a yellow umbrella. The poles are vertical metal cylinders, the signs are flat rectangles, and the cart has a wire-grid structure. There are exactly three people, one shopping cart, one pedestrian signal, and two white signs clearly visible.

Masked target. A yellow umbrella.



Original caption: A bright blue construction wall fills the background, with two white rectangular signs attached to it, a gray pedestrian signal mounted above, an orange metal shopping cart piled with clothes, and three people on the sidewalk. The woman in a green-and-black outfit is to the left of the cart, the bearded man in a light shirt stands next to the cart, and the man in dark blue clothing is to the right near a yellow umbrella. The poles are vertical metal cylinders, the signs are flat rectangles, and the cart has a wire-grid structure. There are exactly three people, one shopping cart, one pedestrian signal, and two white signs clearly visible.

Masked target: a yellow umbrella

Rewritten caption: A bright blue construction wall fills the background, with two white rectangular signs attached to it, a gray pedestrian signal mounted above, an orange metal shopping cart piled with clothes, and three people on the sidewalk. The woman in a green-and-black outfit is to the left of the cart, the bearded man in a light shirt stands next to the cart, and the man in dark blue clothing is to the right. The poles are vertical metal cylinders, the signs are flat rectangles, and the cart has a wire-grid structure. There are exactly three people, one shopping cart, one pedestrian signal, and two white signs clearly visible.

Question: What object is visible at the far right above the man in dark blue clothing?

Options:
1.a yellow umbrella
2.a cart
3.a streetlight
4.I don't know

Answer: a yellow umbrella

Figure 3: **Example training instance for Visual Dependency Induction.** Each instance contains an original caption, a masked target, a rewritten caption, a question, answer options, and the correct answer. The target visual fact, *a yellow umbrella*, is removed from the rewritten caption while the remaining image-caption context is preserved. The model must ground the question in the image to recover the missing fact.

Rewritten caption. A bright blue construction wall fills the background, with two white rectangular signs attached to it, a gray pedestrian signal mounted above, an orange metal shopping cart piled with clothes, and three people on the sidewalk. The woman in a green-and-black outfit is to the left of the cart, the bearded man in a light shirt stands next to the cart, and the man in dark blue clothing is to the right. The poles are vertical metal cylinders, the signs are flat rectangles, and the cart has a wire-grid structure. There are exactly three people, one shopping cart, one pedestrian signal, and two white signs clearly visible.

Question. What object is visible at the far right above the man in dark blue clothing?

Options.

1. A yellow umbrella.
2. A cart.
3. A streetlight.
4. I don’t know.

Answer. A yellow umbrella.

Data construction pipeline. We construct the Visual Dependency Induction data with a teacher-model-assisted generation and filtering pipeline. The teacher model is stronger than the student model used for training, and is used only to generate supervision data. Given an image-caption pair, the teacher model first identifies a concrete visual target fact that is directly observable in the image and is also mentioned in the caption. The target fact is selected from common visually grounded categories, including object presence, color, counting, spatial relations, and physical or structural attributes. We require the target fact to be visually verifiable, unambiguous, and difficult to infer from the remaining text alone.

After selecting the target fact, the teacher model rewrites the caption by removing only the target information while preserving the remaining image-caption context. The rewritten caption is required to remain fluent and aligned with the image, but it must not reveal the answer through synonyms, paraphrases, indirect descriptions, or commonsense shortcuts. The teacher model then generates a multiple-choice question that queries the removed visual fact. Each question contains one correct answer, visually plausible hard distractors, and an “I don’t know” option. The distractors are chosen to be plausible under the same image context but incorrect with respect to the queried visual evidence, so that the model cannot answer reliably from the rewritten caption alone.

Automatic verification. To reduce caption leakage and ambiguous supervision, we use a second VLM/LLM-based verifier to review the generated samples. The verifier checks whether: (1) the rewritten caption removes the target fact; (2) the rewritten caption does not leak the answer through synonyms or indirect clues; (3) the question is answerable from the image; (4) the correct answer is unique among the options; (5) the distractors are plausible but visually incorrect; and (6) the answer cannot be inferred from the rewritten caption alone. Samples that fail any of these checks are discarded. In our implementation, we generate the initial VDI candidates with Gemini and use GPT-based verification to filter invalid examples. Starting from 46.5K generated candidates, the verification stage retains approximately 20K high-quality examples.

Human inspection. In addition to automatic verification, we manually inspect a subset of the generated data to estimate data quality. We review approximately 1K examples sampled from the generated pool, focusing on caption leakage, answer uniqueness, visual ambiguity, and whether the question truly requires image grounding. This inspection confirms that the retained examples generally follow the intended VDI format: the rewritten caption preserves the scene context while removing the answer-critical visual fact, and the correct answer is uniquely recoverable from the image.

Category balance. We balance the final VDI data across major visual-dependency categories to avoid overfitting to a single type of visual cue. The final data approximately covers object presence, color, counting, spatial relations, and physical or structural attributes. Each category contributes roughly a comparable portion of the training set. This balanced design encourages the model to learn different forms of image-text dependency rather than relying on one dominant shortcut type.

A.6 Visual-CoT Data Format and Latent Supervision

We use Visual-CoT for supervised training. Each example is stored as one JSONL row and contains an image-grounded question-answer pair, a grounding bounding box, and textual reasoning annotations.

Each data row contains the following fields:

- `question`: the natural-language visual question.
- `answer`: the short answer used as the final explicit answer target.
- `full_answer`: a complete-sentence answer, used as latent answer supervision.
- `image`: the image filename.
- `width`, `height`: original image size.
- `bboxs`: a list of relevant visual bounding boxes in pixel coordinates, formatted as $[x_0, y_0, x_1, y_1]$. We use the first box as the grounding target.
- `reasoning`: symbolic GQA-style reasoning operations, such as `select`, `relate`, `query`, and attribute filters.
- `thought`: a natural-language chain-of-thought style explanation describing the visual reasoning process.
- `dataset` and `split`: metadata indicating the source dataset and split.

An example row is:


```
{
  "question": "Who is wearing a shirt?",
  "answer": "girl",
  "full_answer": "The girl is wearing a shirt.",
  "image": "2331819.jpg",
  "width": 500,
  "height": 375,
  "bboxs": [[214, 0, 433, 374]],
  "dataset": "gqa",
  "split": "train",
```

```
  "plan": "1. We are looking for someone who is wearing a shirt in the image.\n2.
  ↪  First, let's identify all the people in the image.\n3. Next, we need to
  ↪  determine who among the people is wearing a shirt.\n4. After identifying the
  ↪  person, we can state the answer based on the observation from the image."
}
```

Latent-state construction. Instead of exposing the full reasoning text as ordinary output tokens, we convert different supervision signals into typed latent blocks. The assistant-side training sequence is constructed as:

$$\texttt{<plan_lvr_start>}\,\texttt{<lvr>}^{K_1}\,\texttt{<plan_lvr_end>}$$
$$\texttt{<bbox_lvr_start>}\,\texttt{<lvr>}^{3}\,\texttt{<bbox_lvr_end>}$$
$$\texttt{<patch_lvr_start>}\,\texttt{<lvr>}^{K_3}\,\texttt{<patch_lvr_end>}$$
$$\texttt{<full_lvr_start>}\,\texttt{<lvr>}^{K_4}\,\texttt{<full_lvr_end>}$$
$$\texttt{<answer>}\,a_{\text{short}}\,\texttt{</answer>}.$$

The four latent blocks correspond to different supervision sources:

- **Plan latent block.** The `thought` field is tokenized and truncated to at most K_1 tokens. These token embeddings are used as latent targets for the plan block.
- **Bounding-box latent block.** The selected bounding box is normalized by image width and height, converted from (x_0, y_0, x_1, y_1) to (c_x, c_y, w, h), and represented by three latent targets: center coordinate (c_x, c_y), width w, and height h.
- **Patch latent block.** The normalized box is mapped to Qwen2.5-VL image patch indices. Candidate patch embeddings inside the box are ranked by cosine similarity to the mean embedding of the full answer. The top patches are used as latent visual targets.
- **Full-answer latent block.** The `full_answer` field is tokenized and used as latent answer-level supervision.

NeurIPS Paper Checklist

1. **Claims**

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

   Answer: [Yes]

   Justification: The main claims in the abstract and introduction are supported by the proposed SLVR framework, as shown in Abstract and Introduction 1.

   Guidelines:

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

2. **Limitations**

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

   Answer:[Yes]

   Justification: We discuss limitations in the Conclusion and Limitations section 5.

   Guidelines:

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

3. **Theory assumptions and proofs**

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

   Answer: [N/A]

Justification: None

Guidelines:

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

4. **Experimental result reproducibility**

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

   Answer: [Yes]

   Justification: We provide the backbone model, training data sources, two-stage training procedure, hyperparameters, loss weights, evaluation benchmarks, and decoding settings in the experimental setup and implementation details, as shown in 4.

   Guidelines:

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

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

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

Answer: [Yes]

Justification: We submit the code, data construction prompts, and processed training metadata after the review process. At submission time, the paper provides detailed implementation and data construction details to support reproducibility.

Guidelines:

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

6. **Experimental setting/details**

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

Answer: [Yes]

Justification: The paper specifies the backbone, training data, data construction process, optimizer, learning rates, batch size, training epochs, sequence length, loss weights, evidence-selection threshold, and evaluation protocol in the experimental setup and implementation details,as shown in 4 and A.3.

Guidelines:

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

7. **Experiment statistical significance**

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

Answer: [No]

Justification: We do not report error bars or confidence intervals because training Qwen2.5-VL-7B variants multiple times is computationally expensive. We instead provide consistent comparisons across multiple benchmarks and ablation settings.

Guidelines:

- The answer [N/A] means that the paper does not include experiments.
- The authors should answer [Yes] if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.

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

8. **Experiments compute resources**

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

   Answer: [Yes]

   Justification: The implementation details specify that experiments are conducted on 8 NVIDIA A100 GPUs with bf16 mixed-precision training. We also report the main training settings needed to estimate compute requirements, as shown in 4.2.

   Guidelines:

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

9. **Code of ethics**

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

   Answer: [Yes]

   Justification: The research uses existing public multimodal datasets and models for visual reasoning research and does not involve harmful data collection, human-subject intervention, or deployment in high-risk settings, as shown in 4.

   Guidelines:

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

10. **Broader impacts**

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

    Answer: [Yes]

Justification: The proposed method may improve grounded multimodal reasoning and reduce reliance on language shortcuts. Potential risks include improved capability of MLLMs that could be misused in downstream applications, and we discuss limitations related to hallucination and robustness, as shown in 5.

Guidelines:

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

11. **Safeguards**

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

Answer: [N/A]

Justification: This work does not release a high-risk pretrained foundation model, image generator, or scraped dataset. The method is evaluated as a research framework built on existing public models and datasets.

Guidelines:

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

12. **Licenses for existing assets**

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

Answer: [Yes]

Justification: We cite the existing models and datasets used in this work, including Qwen2.5-VL, COVT, Visual-CoT, V*, MMVP, BLINK, MathVista, and ChartQA. We follow their intended research usage and licensing terms where available.

Guidelines:

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

13. **New assets**

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

    Answer: [Yes]

    Justification: We introduce a 20K COVT-rewritten masked-text dataset and describe its construction process, filtering protocol, and data format in the paper and appendix. Documentation will be provided with the released assets if they are made public.

    Guidelines:

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

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

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

    Answer: [N/A]

    Justification: This work does not involve crowdsourcing experiments or research with human subjects. Human verification is used only as internal data quality control for a subset of rewritten examples.

    Guidelines:

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

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

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

Answer: [N/A]

Justification: The paper does not involve human-subject research or crowdsourcing studies requiring IRB approval. Internal manual inspection is used only for dataset quality filtering.

Guidelines:

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

16. **Declaration of LLM usage**

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

Answer: [Yes]

Justification: We use Gemini 3.1 as part of the data construction pipeline to rewrite COVT examples into masked-text training instances. This usage is described in the training data section and detailed in the appendix.

Guidelines:

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