VERVE: VERIFIED VISUAL SEARCH

Anonymous authors
Paper under double-blind review

ABSTRACT

Recent visual search methods improve perception in high-resolution images, but these gains can come at a cost: reduced ability to recognize when no target satisfies the query (hallucination). We introduce Verified Visual Search (VERVE), a framework for measuring and mitigating this perception-hallucination trade-off. Our VERVE-MCQ benchmark pairs positive visual search queries with minimally edited negative counterparts for which no matching target exists, testing both target identification and query verification. We find that state-of-the-art visual search methods surprisingly underperform their own base models on VERVE-MCQ. To diagnose the root cause, VERVE-BINARY simplifies the queries into a yes/no task. Yet the failure persists for open methods, indicating a broader loss of discriminativeness. To tackle this, we propose VERVE-TRAIN, a GRPO-based method that trains models to verify rather than assume target existence, adding negative queries (*plain* mode) and further on-policy self-verification rewards (*OPSV* mode). Notably, our trained Qwen3.5-9B surpasses Gemini-3.7-Flash by 13.8% on VERVE-MCQ. Crucially, VERVE-TRAIN effectively reduces hallucinations across six benchmarks and remain competitive on eight perception benchmarks. Furthermore, these improvements transfer across tasks, domains, and modalities on seven external benchmarks, demonstrating benefits beyond high-resolution visual search.

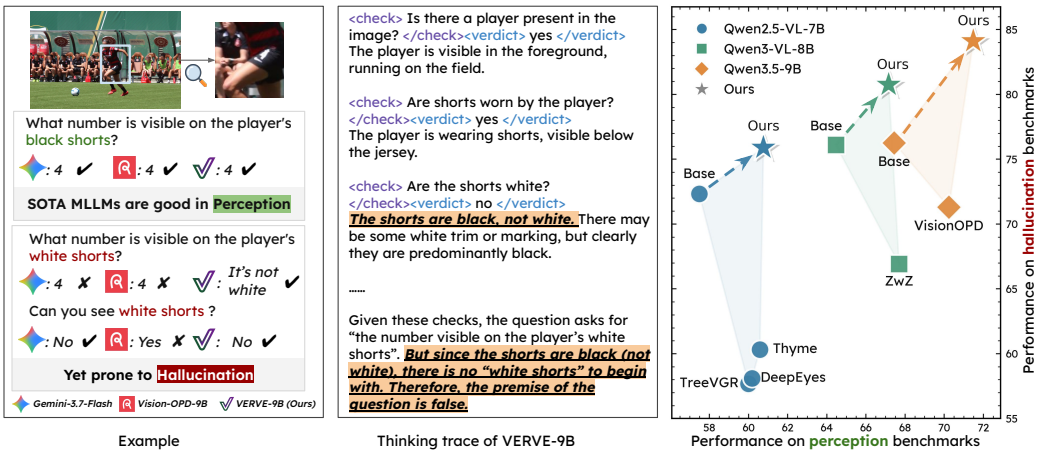

Figure 1: Left: Example shows how models succeed on perception questions yet struggle with simple hallucination questions. Middle: Thinking trace of Verve-9B shows it successfully identifies the false premise. Right: Average model performance across seven perception and six hallucination benchmarks (Sec. 4.3). VERVE-TRAIN significantly mitigates the perception-hallucination trade-off.

1 INTRODUCTION

High-resolution visual search (Wu & Xie, 2024) requires MLLMs to follow instructions, locate small objects, and answer related questions. Consider how a human approaches visual search in a complex scene (Fig. 1, left). If asked to find "the number on the player's black shorts" in the image, a human can locate the small target and identify the correct number. If the query is slightly altered to "the number on the player's *white* shorts," that same human will reliably recognize that the shorts are not white. Frontier MLLMs such as Gemini (Google DeepMind, 2026b) and specialized approaches such as Vision-OPD (Yuan et al., 2026) fail this robustness test. Given the altered query, both models accept the false premise and report a number; even when the query is simplified to the

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

binary question "Can you see white shorts?", Vision-OPD still fails. Reliable visual search thus demands a dual competency: precise target recognition (perception) *and* robust query verification (avoiding hallucination).

Recent visual search methods have made massive strides in localizing small targets, using multi-turn agentic approaches (DeepEyes (Zheng et al., 2026), Thyme (Zhang et al., 2026b)) and internalized single-pass methods (TreeVGR (Wang et al., 2026a), ZwZ (Wei et al., 2026a), Vision-OPD (Yuan et al., 2026)). Yet the benchmarks that drive this progress universally assume a valid target is present (Wang et al., 2025; Wu & Xie, 2024), and existing hallucination probes (Li et al., 2023; Miyai et al., 2025; Xiao et al., 2026) only require coarse perception on standard-resolution images. Together, they fail to capture the dual challenge of perceiving fine details while rejecting false premises in visually intensive scenes. Consequently, this leaves a key gap in model training, as no existing paradigms address both aspects simultaneously.

To expose and mitigate this perception-hallucination trade-off, we introduce Verified Visual Search (VERVE). We first propose VERVE-MCQ (Sec. 2), a human-in-the-loop curated benchmark pairing high-res (positive) search queries with minimally edited counterfactual queries (altering an object, attribute, or relation). We evaluate models using paired accuracy, awarding credit only when a model correctly answers a positive query and rejects its counterfactual twin. Our evaluation reveals a striking finding: state-of-the-art (SOTA) visual search methods universally underperform their base models (models that they are fine-tuned from) in absent-target rejection (Fig. 1, right). A complementary VERVE-BINARY split strips queries down to pure presence/absence judgments toward the edited part, confirming that the discriminative capacity of open-source methods degrades fundamentally.

To mitigate this trade-off, we propose VERVE-TRAIN (Sec. 3), a post-training framework built on GRPO (Shao et al., 2024). We first introduce a data-driven approach (*Plain* mode), which trains the model on a balanced mix of positive and negative queries using a final-answer correctness reward. To further enhance discriminativeness and foster verification behavior, we additionally propose an On-Policy Self-Verification (*OPSV*) mode. OPSV encourages the model to first self-propose and answer verification questions purely in an on-policy manner, then commit to a final answer (Fig. 1, middle). Applied to three backbones (7B-9B) (Bai et al., 2025b;a; Team, 2026), VERVE-TRAIN improves VERVE-MCQ paired accuracy by up to 35.2%, allowing our 9B model to outperform Gemini-3.7-Flash by 13.8%. These gains generalize: across 14 perception and hallucination benchmarks, VERVE-TRAIN reduces hallucinations while maintaining robust perception (Fig. 1, right), achieving a Pareto improvement over its backbone, where prior methods sacrifice hallucination performance. Furthermore, VERVE-TRAIN shows strong transfer performance across seven external tasks (Sec. 4), confirming that verified search instills a general discriminative capability.

In summary, our contributions are: (1) **VERVE Benchmarks.** VERVE-MCQ and VERVE-BINARY, strictly *human-verified* diagnostic benchmarks that pair high-resolution search tasks with minimally edited counterfactual queries. (2) **Revealing the perception-hallucination trade-off.** A systematic evaluation revealing that search specialization universally *degrades* hallucination resistance across current SOTA agentic and internalized methods. (3) **VERVE-TRAIN.** A GRPO-based post-training framework and a scalable pipeline for synthesizing high-quality negative training queries, that resolves this trade-off through balanced data and on-policy self-verification (Plain and OPSV modes).

2 THE VERVE BENCHMARK

To evaluate *verified visual search* (VERVE), we construct VERVE-MCQ (Sec. 2.1), a high-res visual search benchmark that pairs each positive MCQ query with a negative twin edited by a single unsupported object, attribute, or relation. We further propose VERVE-BINARY (Sec. 2.2), which dissects the edited components from the MCQ queries into simple yes/no questions to test if the model still hallucinates under standalone edits.

2.1 VERVE-MCQ

Task and Data We motivate the design of VERVE-MCQ by framing it as visual search for a target object in a high-resolution image, where the model must both locate the target when it is present and recognize when it is absent. Each instance is a pair $(Q_{\text{pos}}, Q_{\text{neg}})$ that shares one image and one set of five options (Fig. 2). In Q_{pos}, the referring expression is supported by the image, and exactly one option is the correct, image-supported answer. In Q_{neg}, a single object, attribute, or relation in the expression is edited so that it no longer matches anything in the image. One of the shared options is a

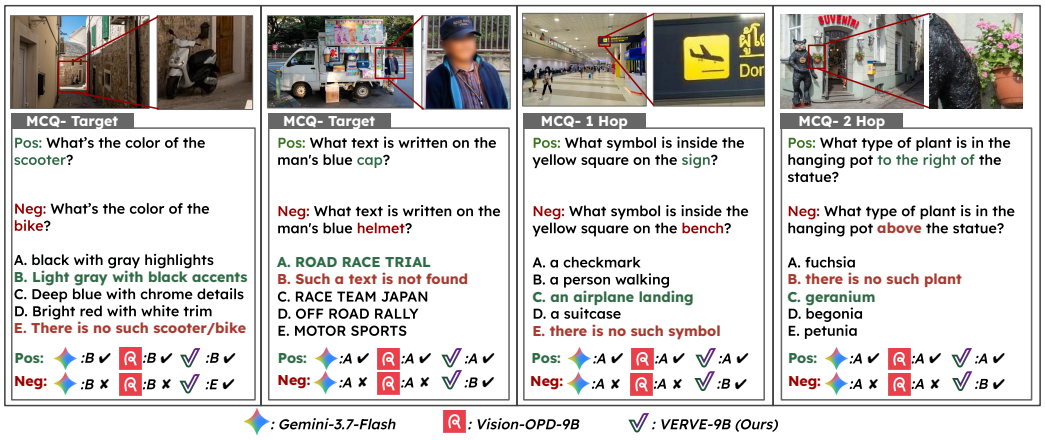

Figure 2: **VERVE-MCQ EXAMPLES** sorted by negative edited slots (TARGET, 1-HOP, 2-HOP) Both Q_{pos} (green) and Q_{neg} (red) with their ground-truth choice (same color) are shown, together with model predictions from Gemini-3.7-Flash, Vision-OPD, and our VERVE-9B model.

target denial option (e.g., "There is no such {target object}" or "{target object} is not found"). This option is wrong for Q_{pos} but is the only correct answer for Q_{neg}.

We build on high-resolution scenes and positive queries from ZOOMBENCH (Wei et al., 2026a) and TREEBENCH (Wang et al., 2026a). Candidate negative edits are proposed by Gemini-3-Flash (Google DeepMind, 2025) and GPT-5 (OpenAI, 2025), then verified by humans (details in App. A). Two annotators inspected all 2,736 candidate edits, discarding 977 and retaining 1,759 verified pairs, of which 617 were manually rewritten.

Construction and Validation We parse each positive query into an object-anchored referring chain that targets a specific object; we denote the target object and its optional attributes as TARGET.

$$\underbrace{(\text{obj},\ \text{attr})}_{\text{TARGET}} \xleftarrow{\text{rel}} \underbrace{(\text{obj},\ \text{attr})}_{\text{1-HOP}} \xleftarrow{\text{rel}} \underbrace{(\text{obj},\ \text{attr})}_{\text{2-HOP}}, \tag{1}$$

The referring chain consists of distinct HOPs, where each hop centers on an object anchor (optionally with attributes) linked to the preceding hop via a relation. We refer to these editable fields as *slots* ("obj", "attr", or "rel" within each hop). For example, in the 2-hop query from Fig. 2 ("*What type of plant is in the hanging pot to the right of the statue?*"), the referring chain is: statue → hanging pot → plant. We use these chains to construct controlled edits and categorize them by slot and hop.

Given the image and chain, Gemini-3-Flash (Google DeepMind, 2025) alters exactly one slot. A valid edit ensures the modified expression is unsatisfied in the image (e.g., changing "to the right of the statue" to "above the statue" when no hanging pot exists above). Human annotators then review all proposed edits to confirm that every positive query has a unique correct answer and that the edited expression in each negative twin is neither satisfied elsewhere in the scene nor ambiguous. Additional details regarding benchmark settings, statistics, and construction are provided in App. A.1.

Evaluation As our primary metric, we report paired accuracy over the set of test pairs $\mathcal{P}$:

$$\text{Acc}_{\text{paired}} = \frac{1}{|\mathcal{P}|} \sum_{p \in \mathcal{P}} \mathbb{I}\big(\hat{y}_p^+ = y_p^+\big)\, \mathbb{I}\big(\hat{y}_p^- = y_p^-\big)\,, \tag{2}$$

where y_p^+ and y_p^- are the ground-truth options for Q_{pos} and Q_{neg} in pair p, and $\hat{y}_p^+$ and $\hat{y}_p^-$ are the model's corresponding predictions. We choose $\text{Acc}_{\text{paired}}$ to prevent models from exploiting label biases: a model receives credit only when it correctly answers both Q_{pos} (fine-grained perception) and Q_{neg} (reliable verification), ensuring true verified visual search. **Failures on Paired Visual Search** As qualitatively illustrated in Fig. 2, models such as Gemini-3.7-Flash and the open-source SOTA Vision-OPD answer positive queries well but do not reliably reject their negative twins. Consequently, both models exhibit low paired accuracy; as reported in Tab. 1, Gemini-3.7-Flash achieves only $\text{Acc}_{\text{paired}} = 41.5\%$ and VISION-OPD achieves just $\text{Acc}_{\text{paired}} = 15.6\%$, underscoring the benchmark's difficulty. For example, in Fig. 2, models correctly handle the positive case but fail on the negative query, answering "*What text is written on the man's blue helmet?*" with a plausible string even though the man clearly is not wearing a helmet.

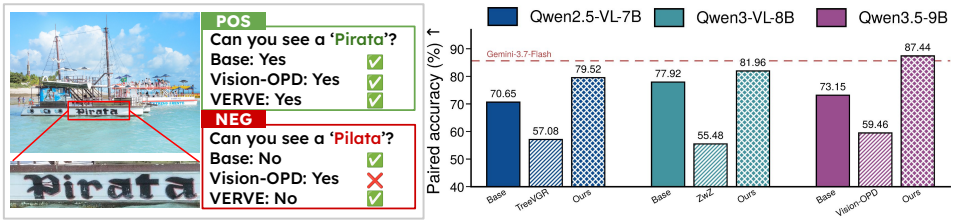

Figure 3: Left: One qualitative example on VERVE-Binary; Right: Avg. $\text{Acc}_{\text{paired}}$ on VERVE-Binary.

2.2 VERVE-BINARY

Because each negative query in VERVE-MCQ is built by making a single localized edit to a complex question, a failure could mean either (1) the model can detect the edit in isolation but overlooks it when embedded in the complex query, or (2) the model cannot recognize the altered element at all. To distinguish these, we introduce VERVE-BINARY to test whether these failures persist when models are asked directly to verify the premise. We construct VERVE-BINARY from the edit history of VERVE-MCQ: For each VERVE-MCQ pair $(Q_{\text{pos}}, Q_{\text{neg}})$, we extract the applied negative edit, an object, attribute, or relation modification, and prompt GPT-5 (OpenAI, 2025) to formulate a direct yes/no question about the edited content, such as "Can you see *[edit]* in this image?" Retaining the corresponding positive counterpart yields a balanced binary query pair $(B_{\text{pos}}, B_{\text{neg}})$, with reference answers *yes* and *no*, respectively (Fig. 3, left). Both questions are evaluated in separate contexts and scored with Eq. 2. Detailed construction process is in App. A.2.

Loss of Discriminativeness Fig. 3 (right) reports averaged $\text{Acc}_{\text{paired}}$ on VERVE-BINARY. Despite their reported gains on visual search and fine-grained perception benchmarks, TREEVGR(Wang et al., 2026a), ZWZ(Wei et al., 2026a), and VISION-OPD(Yuan et al., 2026) all score below their respective base models on this diagnostic, achieving only 57.08%, 55.48%, and 59.46% paired accuracy, respectively. We term this base-relative decrease a *loss of discriminativeness*: reduced success at jointly accepting supported content and rejecting unsupported content (negative edits). Fig. 3 illustrates this pattern: VISION-OPD accepts "Pilata" as the visible text, whereas its base model correctly rejects that claim. In contrast, Gemini-3.7-flash achieves 85.65% $\text{Acc}_{\text{paired}}$. This suggests its failures on VERVE-MCQ likely stem from ignoring the negative edits, rather than loss of discriminativeness. Per-model results, base-model comparisons, and further examples are provided in App. D.1. Together, these results show that gains on conventional visual benchmarks do not necessarily translate into reliable discrimination between supported and unsupported visual claims.

3 VERVE-TRAIN

To mitigate the "loss of discriminativeness" observed in current open-source methods, we propose VERVE-TRAIN, which optimizes the model using both positive queries (to enhance perception) and negative queries (to mitigate hallucinations). We first construct a balanced training dataset (Sec. 3.1) and find that fine-tuning on this dataset with a standard correctness reward already yields notable gains on VERVE-MCQ. Building on prompt engineering insights documented in App. D.9.1, we further elicit the model's explicit verification behavior via an on-policy self-verification (*OPSV*) training strategy (Sec. 3.2). Fig. 4 overviews the data construction pipeline and the reward design.

3.1 TRAINING DATA CONSTRUCTION

Overview We build our training set upon ZwZ-RL training set (Wei et al., 2026a), which contains 74K high-resolution images paired with (positive) questions and answers. For each positive question, we synthesize a *negative* counterpart whose premise is false with respect to the image, together with ground-truth answers and a set of atomic golden verification Q&As for OPSV mode. To ensure the negatives are genuinely false, we decouple perception from reasoning: Qwen3.5-27B performs all visual perception (as the *captioner* and *verifier*), while GPT-OSS-120B performs all language reasoning (as the *proposer* and *annotator*). The pipeline consists of three steps (Fig. 4, top) with the running example "What is the material of the container holding the nuts?" (answer: "plastic").

Step 1: Dense Captioning The captioner first produces a detailed description of each image. Since the proposer has no direct access to the image, this caption serves as its visual context, grounding the proposed edits in actual scene content.

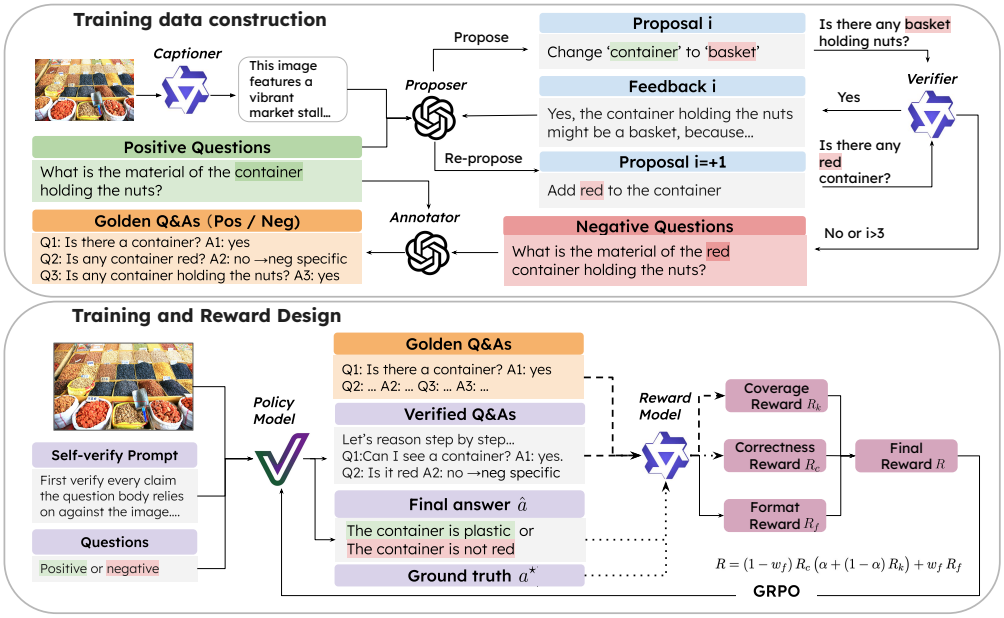

Figure 4: Top: Our training data construction pipeline. Bottom: Demonstration of VERVE-TRAIN in on-policy self-verification (OPSV) mode. The plain mode uses default prompt with $R = R_c$.

Step 2: Propose-verify Loop Given the caption, the positive question, and its answer, the proposer creates a negative question by editing one entity in the question's premise, e.g., replacing "container" with "basket". A negative is valid only if the edited premise does not hold in the image. To test this, the proposer also emits a minimal binary question targeting the edit ("Is there a basket holding the nuts in the image?"), which the verifier answers by inspecting the image. If the verifier answers "Yes", the edit is rejected and we append the verifier's feedback to the proposer's context as new visual evidence, and the proposer proposes an alternative edit (e.g., "red container"). The proposal will be accepted (loop terminated) when the verifier answers "No", or after three iterations.

Step 3: Ground-truth Annotation We annotate two types of supervision. *(a) Answers.* Positive questions retain their original ground truth from ZwZ-RL ("plastic"). For negative questions, GPT-OSS-120B writes an answer that identifies the false premise ("The container is not red."). *(b) Golden verification Q&As.* As additional supervision for the OPSV mode, GPT-OSS-120B decomposes each question into atomic binary checks. Checks on unedited premises are labeled "yes" (e.g., "Is there a container holding the nuts?"), while the check on the edited premise is labeled "no" (e.g., "Is the container red?"). On average, each question yields 3.8 checks.

Final Training Set To optimize training compute, we subsample a 57K subset maintaining a 2:1 positive-to-negative ratio (ablated in Sec. 4.5). We visualize representative training samples in Fig. 7.

3.2 TRAINING PROCESS AND REWARD DESIGN

Setup We train VERVE-TRAIN in two modes: Plain (standard prompt with a correctness-based reward) and OPSV (self-verification prompt, illustrated in Fig. 4). The full prompts are in App. F. Given an image-question tuple (I, q) as input, the policy π_θ generates a response y ending with a predicted answer $\hat{a}$ inside `<answer>` tags. We optimize π_θ using GRPO (Shao et al., 2024), sampling $K = 8$ candidate responses per prompt at a temperature of 1.0.

Rewards in Plain Mode In plain mode, we employ solely a correctness reward R_c based on whether the final predicted answer $\hat{a}$ matches the ground truth $a^\star$:

$$R = R_c = \mathbb{1}\left[\hat{a} = a^\star\right]. \tag{3}$$

We evaluate correctness primarily via option-letter matching ($\hat{a}$ vs. $a^\star$), falling back to Qwen3-30B to judge semantic equivalence if exact parsing fails or for open-ended questions.

Rewards in OPSV Mode In OPSV mode, the prompt asks the model to verify the premises of the question against the image before answering. The model emits each check as

<check> $\hat{g}_i$ </check><verdict> $\hat{v}_i$ </verdict>, followed by one sentence of supporting visual evidence, and then reasons toward the final answer. This yields an on-policy checklist $\hat{\mathcal{G}} = \{(\hat{g}_i, \hat{v}_i)\}_{i=1}^{n}$ alongside the predicted answer $\hat{a}$. In the running example (Fig. 4), the model asks "Can I see a container?" "yes" and "Is it red?" "no", then concludes "The container is not red."

Verifying every emitted check against the image during training would be very expensive; we evaluate the checklist against the m golden verifications $\mathcal{G} = \{(g_j, \ell_j)\}_{j=1}^{m}$, $\ell_j \in \{\text{yes}, \text{no}\}$. We use a reward model μ (Qwen3-30B) to match each emitted check with the golden set, assigning $\mu(i) = j$ if $\hat{g}_i$ matches g_j and $\mu(i) = \varnothing$ otherwise. A gold check is considered *covered* if at least one matching emitted check produces the correct verdict and the coverage reward R_k is defined as:

$$R_k = \frac{|\mathcal{C}|}{m}, \qquad \text{where } \mathcal{C} = \left\{ j : \exists\, i,\ \mu(i) = j \ \wedge\ \hat{v}_i = \ell_j \right\}, \tag{4}$$

i.e., the ratio of golden verifications that were checked and correctly answered by the model. Each gold check can be covered at most once to prevent reward hacking through duplicate checks. Asking the right question with the wrong verdict ("Is the container red?" $\rightarrow$ "yes") earns no credit. The final reward is

$$R = (1 - w_f)\, R_c \left(\alpha + (1 - \alpha)\, R_k\right) + w_f\, R_f, \tag{5}$$

where format reward $R_f \in \{0, 1\}$ rewards structural compliance. Gating R_k by R_c prevents the model from gaming the reward by generating accurate intermediate checks while predicting an incorrect final answer. Notably, we never directly supervise which questions to ask, but reward the model for coming up with relevant questions by itself (on-policy). It may learn to ask additional questions it deems purposeful. Hyperparameter (α and w_f) choices are given in App. B.

4 Experiments

We present experiments of VERVE-TRAIN on three tasks, i.e., evaluation on VERVE benchmarks (Sec. 4.2), perception-hallucination benchmarks (Sec. 4.3), and cross-task transfer (Sec. 4.4). In addition, we ablate important training strategies and data selections (Sec. 4.5).

4.1 Experimental Setup

Evaluated Models On VERVE-MCQ, we evaluate frontier models (Gemini-3.7-Flash (Google DeepMind, 2026b), GPT-5.6-Sol (OpenAI, 2026), Qwen3.5-397B (Team, 2026), GLM-4.6V (Hong et al., 2025), Kimi-K3 (Team et al., 2026)), agentic visual search methods (Thyme (Zhang et al., 2026b), DeepEyes (Zheng et al., 2026), DeepEyes-v2 (Hong et al., 2026)), native ToolUse variants (Qwen3-VL-8B and Qwen3.5-9B with Qwen-Agent harness (Qwen Team, 2025)) and single-pass models (TreeVGR (Wang et al., 2026a), ZwZ (Wei et al., 2026a), Vision-OPD (Yuan et al., 2026)).

Training Setup To fairly compare with visual search methods above, we train VERVE-TRAIN on their base models: Qwen2.5-VL-7B-Instruct, Qwen3-VL-8B-Instruct, and Qwen3.5-9B. Each backbone is trained separately in plain and OPSV modes on our 57K dataset for 1 epoch (detailed in App. B).

4.2 Evaluation on Verve-MCQ

Tab. 1 reports $\text{Acc}_{\text{paired}}$ overall and broken down by hop (Target, 1-hop, 2-hop) and edit type (obj, attr, rel). In the appendix, we provide results for additional frontier models (Tab. 11), separate accuracy on positive and negative MCQs (Tab. 12), $\text{Acc}_{\text{paired}}$ by image source (Tab. 13), prompt engineering study (App. D.9.1) and a human evaluation on 100 randomly sampled MCQs (App. E).

Current models struggle on VERVE-MCQ As shown in Tab. 1, even the top frontier MLLM, Gemini-3.7-Flash, scores only 41.5% on our benchmark. With 3T parameters, Kimi-K3 scores an averaged 26.6%, while human baselines (Tab. 15) score 70.5% on averaged $\text{Acc}_{\text{paired}}$. Agentic models only score a single digit on average score. For example, Thyme achieves just 2.7% $\text{Acc}_{\text{paired}}$, even under the random guess at 4%. Meanwhile, the ToolUse variants show mixed effects on two base models, degrading the base performance of Qwen3.5-9B from 20.1% to 17.9%. One-pass SOTA visual search methods (TreeVGR, ZwZ and Vision-OPD) lag behind their respective base models, where Vision-OPD exhibits a 4.5% drop (compared with Qwen3.5-9B at 20.1%) in averaged $\text{Acc}_{\text{paired}}$. This exposes a core flaw in existing visual search approaches: a lack of robustness to hallucinated conditions. Furthermore, our attention analysis in App. D.7 reveals that model like ZwZ exhibits a rigid "tunnel vision" effect in their internal attention patterns, likely explaining such flaw.

Performance Drops on Fine-Grained Edits Tab. 1 reports $\text{Acc}_{\text{paired}}$ by edit slots and types, showing that models are most robust to negative edits at the "Target", followed by "1-Hop", and "2-Hop". For example, when an attribute in the "Target" position is edited to a negative, Qwen3.5-9B achieves 21.6%. This drops to 18.1% at "1-Hop" and further to 9.5% at "2-Hop". Even Gemini-3.7-Flash follows this trend, showing that current models struggle to spot errors in fine-grained queries and perform multi-hop reasoning. Edit type also impacts performance: models are generally weaker on relation edits than on other types. At "1-Hop", Qwen3.5-9B scores only 8.9% on relation edits, roughly half its score on attribute edits (18.1%).

Table 1: Paired accuracy ($\text{Acc}_{\text{paired}}$) on VERVE-MCQ, **Best** and second-best on each backbone.

		TARGET		1-HOP			2-HOP		
Model	**Avg.**	obj	attr	obj	attr	rel	obj	attr	rel
Frontier Models									
GLM-4.6V(107B)	15.5	24.9	18.1	17.1	6.6	6.9	9.5	6.8	4.6
Qwen3.5-397B	23.0	36.4	23.9	23.1	16.8	12.9	14.3	8.1	6.1
Kimi-K3(3T)	26.6	36.0	33.1	24.9	19.5	18.3	16.7	13.5	9.1
GPT-5.6 Sol	13.9	24.7	16.5	11.5	7.1	6.4	7.1	2.7	3.0
Gemini-3.7-Flash	41.5	47.0	48.9	46.8	35.8	32.7	26.2	27.0	**21.2**
Small Models									
Qwen2.5-VL-7B	11.4	21.6	8.7	11.5	5.8	6.4	8.3	4.1	4.6
+Thyme	2.7	6.5	1.5	3.0	1.3	1.0	0.0	0.0	0.0
+DeepEyes	3.9	8.5	3.6	3.4	2.2	0.5	2.4	0.0	0.0
+DeepEyes-V2	3.5	7.9	2.8	3.4	0.9	1.5	2.4	0.0	0.0
+TreeVGR	6.0	17.1	3.1	4.1	1.3	1.5	1.2	0.0	0.0
+VERVE$_{plain}$	33.5	44.5	38.4	34.9	30.5	14.4	33.3	24.3	4.6
+VERVE$_{opsv}$	**40.8**	**50.3**	**44.3**	**42.4**	**42.5**	**21.3**	**42.9**	**35.1**	**7.6**
Qwen3-VL-8B	11.2	19.8	12.0	12.6	5.3	5.5	4.8	0.0	1.5
+ToolUse	19.7	33.5	19.1	22.3	12.4	9.9	11.9	4.1	3.0
+ZwZ	7.6	18.2	7.6	3.4	2.2	1.0	6.0	0.0	1.5
+VERVE$_{plain}$	41.8	**51.9**	45.0	46.8	**40.7**	**24.3**	38.1	29.7	10.6
+VERVE$_{opsv}$	**42.2**	49.0	**48.1**	**47.6**	39.8	21.8	**50.0**	**31.1**	**12.1**
Qwen3.5-9B	20.1	29.0	21.6	22.7	18.1	8.9	13.1	9.5	3.0
+ToolUse	17.9	31.5	21.9	15.2	11.1	6.4	6.0	4.1	3.0
+Vision-OPD	15.6	27.6	19.6	13.0	9.7	5.5	7.1	0.0	1.5
+VERVE$_{plain}$	49.7	57.3	54.2	52.8	48.7	33.7	**52.4**	39.2	**21.2**
+VERVE$_{opsv}$	**55.3**	**61.8**	**60.3**	**58.7**	**54.0**	**47.5**	50.0	**40.5**	19.7

VERVE-TRAIN comprehensively boosts performance As shown in Tab. 1, fine-tuning three base models with VERVE-TRAIN yields massive $\text{Acc}_{\text{paired}}$ gains across question categories. On Qwen3.5-9B, VERVE$_{plain}$ already improves $\text{Acc}_{\text{paired}}$ from 20.1% to 49.7%, outperforming Gemini-3.7-Flash by 8.2%. VERVE$_{opsv}$ further boosts performance by an additional 5.6% to 55.3%, outperforming Gemini-3.7-Flash by 13.8%. Crucially, VERVE-TRAIN demonstrates notable resilience in questions with fine-grained negative edits: on the challenging 2-Hop edit slot, VERVE$_{opsv}$ reaches 50.0% on 2-Hop objects, dramatically outperforming Gemini-3.7-Flash (26.2%). Qualitative examples are shown in Fig. 2 and Fig. 8. These samples further confirm that VERVE-TRAIN successfully handles both positive and negative queries. The model reasoning traces in Fig. 12 confirm that our *OPSV* mode incentivizes self-verification behavior.

4.3 PERCEPTION VS. HALLUCINATION: BREAKING THE TRADE-OFF

To systematically investigate this dynamic, we evaluate three key model categories, standard base models, recent visual search models, and our models across eight perception and six hallucination benchmarks (details in App. C.1), with full results reported in Tab. 2. This targeted evaluation reveals a critical trade-off in existing paradigms: while recent visual search models boost performance on perception tasks, they suffer a sharp decline in hallucination resistance relative to their base models. On the Qwen2.5-VL-7B backbone, the agentic approach DeepEyes-V2 sees its average hallucination score drop from 72.3% to 40.6%. The one-pass method TreeVGR similarly drops by 14.7% (57.6% vs. 72.3%). On the Qwen3-VL backbone, ZwZ degrades by 9.2% on hallucination benchmarks. Likewise, on Qwen3.5-9B, Vision-OPD improves perception but noticeably degrades hallucination resistance by 4.9% (71.3% vs. 76.2%).

In contrast, VERVE breaks this perception-hallucination trade-off. On Qwen3.5-9B, our approach not only outperforms Vision-OPD in perception but simultaneously pushes the hallucination average to an unprecedented 84.1% (7.9% higher than its base model). On weaker backbones (Qwen3-VL and Qwen2.5-VL), VERVE achieves a Pareto improvement over the base model, matching the perception accuracy of prior methods while substantially improving hallucination resistance. Furthermore, both our plain and OPSV variants land on this expanded frontier, establishing a new standard for trustworthy visual search. Qualitative examples in Fig. 10 confirm that VERVE-TRAIN effectively reduces hallucinations across various benchmarks.

4.4 CROSS-TASK TRANSFER

Table 2: Perception and hallucination benchmark results, with **best** and second-best scores marked per backbone. Perception benchmarks include V* (Wu & Xie, 2024), HR-Bench-4K (Wang et al., 2025), HR-Bench-8K (Wang et al., 2025), CV-Bench (Tong et al., 2024), ColorBench (Liang et al., 2026), CountQA (Tamarapalli et al., 2025), BabyVision (Chen et al., 2026), and PerceptionBench (Lin et al., 2026). Hallucination benchmarks include FINER-CompreCap (Xiao et al., 2026), RePOPE (Neuhaus & Hein, 2025), DASH (Augustin et al., 2025), HallusionBench (Guan et al., 2024), AMBER Discriminative (Wang et al., 2023), and HaloQuest (Wang et al., 2024). Note that PerceptionBench is excluded from the perception average (same for Fig. 1).

	Perception									*Hallucination*						
Method	*V**	*HR4K*	*HR8K*	*CVB.*	*Color*	*Count*	*BabyV.*	*Per.B.*	*Avg*	*FINER.*	*Re POPE*	*DASH*	*Hall.B.*	*AMB.*	*H.Q.*	*Avg*
Qwen2.5-VL-7B	79.1	72.6	66.4	74.7	76.8	**22.2**	10.5	24.2	57.5	54.5	92.4	81.7	43.3	86.9	75.0	72.3
+TreeVGR	86.3	76.6	72.7	74.3	75.2	**22.2**	12.3	24.1	59.9	43.0	90.0	60.3	29.7	84.3	38.5	57.6
+DeepEyes	82.2	73.5	68.6	78.2	77.6	19.1	12.8	–	58.9	44.6	86.9	54.4	33.2	82.5	46.8	58.1
+DeepEyes-V2	80.6	75.3	72.4	**79.7**	77.6	18.0	13.1	–	59.5	42.2	45.8	51.0	37.5	43.7	23.3	40.6
+Thyme	82.7	**77.5**	72.1	78.2	77.7	21.5	**14.1**	–	60.5	47.3	85.4	63.1	**47.9**	85.2	32.8	60.3
+VERVE$_{plain}$	**89.0**	75.3	72.6	76.2	**78.7**	20.4	13.1	25.8	**60.8**	**63.0**	93.5	**83.5**	43.9	87.4	**84.2**	**75.9**
+VERVE$_{opsv}$	85.9	73.4	**73.3**	77.7	77.6	20.3	13.2	**26.8**	60.2	62.8	**93.9**	80.4	44.2	**88.1**	81.5	75.1
Qwen3-VL-8B	83.2	79.6	76.2	85.2	82.5	28.5	15.7	29.2	64.4	67.5	94.0	75.1	57.5	87.3	75.0	76.1
+ZwZ	**91.1**	**84.3**	**82.0**	**86.9**	83.1	29.5	**16.7**	31.6	**67.7**	56.7	89.4	58.6	60.1	82.6	54.0	66.9
+VERVE$_{plain}$	85.8	83.0	80.0	85.9	83.5	29.8	12.3	**32.9**	65.8	70.5	94.7	84.2	62.1	87.5	**85.8**	**80.8**
+VERVE$_{opsv}$	89.0	84.0	81.6	85.7	**83.6**	**30.5**	15.7	32.0	67.2	**72.1**	**95.1**	**84.4**	**62.7**	**87.9**	82.3	**80.8**
Qwen3.5-9B	83.2	85.6	80.5	86.1	83.8	34.6	18.0	33.9	67.4	71.6	94.0	70.5	63.2	88.0	70.0	76.2
+Vision-OPD	90.1	87.0	84.3	**88.6**	85.2	38.3	18.0	35.5	70.2	68.6	92.5	57.1	58.6	85.4	65.3	71.3
+VERVE$_{plain}$	**90.6**	87.1	84.0	88.2	85.3	**43.5**	**21.7**	39.1	**71.5**	**82.3**	**95.7**	**84.8**	65.9	88.4	**87.7**	**84.1**
+VERVE$_{opsv}$	90.1	**88.8**	**84.5**	88.1	**85.4**	40.0	20.6	**39.9**	71.1	79.7	95.5	82.2	**66.8**	**89.0**	84.2	82.9

Beyond topping VERVE-MCQ and mitigating the perception-hallucination tradeoff, VERVE-TRAIN teaches models to distinguish positive from negative queries, enhancing cross-modal discriminativeness and strengthening vision-language alignment. This improved alignment yields cross-task benefits on benchmarks for which our model was not explicitly trained, including general understanding (MMStar), captioning (PerceptionRubrics), reward modeling (VL-RewardBench, MMRB2), grounding (GroundingME), medical hallucination (ProbMed-1K), and video hallucination (VidHalluc). Table 3 compares our results against ZwZ, Vision-OPD, and several frontier or expert models. Notably, VERVE-TRAIN improves Qwen3.5-9B by 14.1% on VL-RewardBench, 11.4% on GroundingME, and 5.4% on image captioning (PerceptionRubrics). On the first two, the gains extend beyond hallucination and rejection categories to reasoning and general categories (App. D.4). Despite receiving no medical or video training, it also gains on ProbMed-1K (+1.0%) and VideoHalluc (+3.3%). These results, together with the qualitative examples in Fig. 11, show that VERVE-TRAIN generalizes effectively.

Table 3: Cross-task transfer results on MMStar (Chen et al., 2024), PerceptionRubrics (Wei et al., 2026b), VL-RewardBench (Li et al., 2025c), MMRB2 (Hu et al., 2026), GroundingME (Li et al., 2026a), ProbMed-1K (Yan et al., 2025), VidHalluc (Li et al., 2025a). Reference models: [a]GPT-5.2, [b]GPT-5.5, [c]Proxy-GRM (Qiu et al., 2026), [d]GPT-5, [e]Qwen3-VL-235B, [f]MedGemma1.5 (Sellergren et al., 2026), [g]GPT-4o.

Model	*MMStar general*	*P.Rub. caption*	*VL-RW. reward*	*MMRB2 reward*	*G.ME grounding*	*P.Med medical*	*VidHal. video*
Qwen3-VL-8B	71.9	39.0	61.5	54.9	31.1	59.7	**76.5**
+ZwZ	73.1	43.0	63.8	55.4	**36.7**	54.2	75.5
+VERVE$_{plain}$	**74.1**	43.3	**68.9**	**56.9**	35.6	58.6	74.9
+VERVE$_{opsv}$	73.1	**44.0**	67.0	56.3	34.5	**60.6**	76.3
Qwen3.5-9B	74.3	50.8	59.8	56.5	38.8	61.7	76.7
+Vision-OPD	76.1	50.4	66.8	57.4	34.9	52.9	79.3
+VERVE$_{plain}$	**78.1**	**56.2**	73.2	**59.3**	46.4	62.5	**80.0**
+VERVE$_{opsv}$	76.9	53.5	**73.9**	59.1	**50.2**	**62.7**	77.8
Reference	77.1[a]	55.2[b]	72.4[c]	72.2[d]	49.8[e]	57.1[f]	82.4[g]

4.5 ABLATIONS

Ablation on Positive-Negative Ratio Using a 10K training subset, we study the optimal ratio between positive and negative samples (Tab. 4). A 1:1 ratio degrades performance on perception benchmarks, whereas 2:1 and 3:1 ratios substantially recover accuracy. Consequently, we select a 2:1 ratio when scaling to our full 57K training set.

Ablation on Reward Design On the full 57K training set, we ablate our reward design in Tab. 5 by progressively adding up components from base model: (1) adding R_c (our plain training); (2) adding

Table 4: Positive:negative ratio at a fixed budget of 10k, plain reward, Qwen3-VL-8B. **Best** and second-best.

ratio	VERVE MCQ	HR 8K	Re POPE
base	11.2	76.2	94.0
1:0	6.5	74.5	92.8
1:1	**26.8**	74.4	94.1
2:1	24.1	**76.8**	94.3
3:1	23.0	76.5	**94.4**

Table 5: Reward ablation on the 57k training set, Qwen3-VL-8B. SVP: Self-verification (SV) Prompt; *gating*: Eq. 5. **Best** and second-best.

Variant	VERVE MCQ	HR 8K	Re POPE
base	11.2	76.2	94.0
+ R_c	41.8	80.0	94.7
+ R_c + SVP	41.6	81.0	94.7
+ R_c + SVP + R_k	41.4	81.2	94.7
+ R_c + SVP + R_k + Gating	**42.2**	**81.6**	**95.1**

a SV prompt without R_k; (3) adding the coverage reward R_k without gating by R_c; and (4) applying full gated reward function in Eq. 5. As shown in Tab. 5, the full gated reward yields the best $\text{Acc}_{\text{Paired}}$ on VERVE-MCQ and balanced results on perception-hallucination benchmarks.

5 Related Work

Visual Search Benchmarks and Methods Standard visual search benchmarks (Wu & Xie, 2024; Wang et al., 2025) typically examine small-target search. While VERVE-MCQ also targets high-resolution, small-target scenarios, it uniquely evaluates perception and hallucination jointly. Early RL-based methods (Huang et al., 2026b; Shen et al., 2025a; Meng et al., 2025) leverage GRPO (Shao et al., 2024) or DAPO (Yu et al., 2026b) to enhance MLLM reasoning. However, recent work (Yu et al., 2026a; Wang et al., 2026b) identifies perception as the primary bottleneck in visual search tasks. To address this, training-free approaches (Shen et al., 2025b; Kim et al., 2026; Li et al., 2025b; 2026b) employ search strategies with high inference overhead, while agentic methods (Zhang et al., 2026b; Zheng et al., 2026; Hong et al., 2026; Lai et al., 2026) train models to dynamically crop image regions at the cost of high latency. Single-pass methods instead internalize visual search through grounding (Wang et al., 2026a; Su et al., 2026), region-to-global distillation (Wei et al., 2026a), or on-policy distillation (Yuan et al., 2026). Notably, VERVE-TRAIN incentivizes explicit verification during training, boosting perception while suppressing hallucinations. Unlike VAPO (Tian et al., 2026) and PEARL (Zhang et al., 2026a), which supervise reasoning via external, off-policy questions, VERVE-TRAIN generates its verification questions purely on-policy.

Discriminative Hallucination The failure mode highlighted in our work represents a form of *discriminative hallucination*. While cross-modal discriminativeness is foundational to vision-language pre-training (Radford et al., 2021; Zhai et al., 2023; Xiao et al., 2025; Huang et al., 2026c), in MLLMs it directly relates to hallucination robustness. Existing benchmarks evaluate this using binary queries (Li et al., 2023; Neuhaus & Hein, 2025; Augustin et al., 2025; Wang et al., 2023), false premises or unanswerable questions (Wang et al., 2024; Zhang et al., 2025; Miyai et al., 2025; Zhu et al., 2026), or fine-grained query-image mismatches (Xiao et al., 2026). VERVE-MCQ differs by focusing on high-resolution scenarios while jointly evaluating perception and hallucination. While recent work indicates that reasoning training can amplify hallucinations (Xu et al., 2026), we further show: even models trained for vision-centric visual search suffer degraded hallucination resistance.

6 Conclusion and Limitations

Conclusion In this work, we introduced VERVE, a framework designed to benchmark and enhance verified visual search. Through VERVE BENCHMARKS, we revealed that SOTA visual search methods show "loss of discriminativeness", uncovering a broader perception-hallucination trade-off. To address this, we introduced VERVE-TRAIN, which achieves top performance on VERVE-MCQ while mitigating this trade-off with strong gains across six independent hallucination benchmarks. Finally, we showed that VERVE-TRAIN transfers effectively across tasks.

Limitations Due to annotation complexity, VERVE-MCQ is limited up to two-hop scenarios, with two-hop queries being a minority. Evaluating higher-order chains will challenge future systems further. Additionally, while VERVE-TRAIN outperforms Vision-OPD in perception, it trails ZwZ by $\sim$0.5%. Closing this gap while maintaining hallucination robustness remains important.

AI USE STATEMENT

In this work, we used generative AI tools for coding assistance, polishing paper writing, and assisting in generating training datasets and is part of the benchmark generation pipeline. We have not used generative AI tools for coming up with ideas, drawing pipeline figures. We have reviewed all AI-assisted work (codes, data, writing). We take responsibility for the final content of this work, including text, claims or artifacts produced with the aid of generative AI.

REPRODUCIBILITY STATEMENT

We are committed to ensuring the reproducibility of the research presented in this paper. To this end, comprehensive implementation details for our models and experiments are provided in the appendix, including the training procedures and all hyperparameters used. Furthermore, we promise that upon acceptance of this paper, benchmark, all source code, training set, and pre-trained weights will be made publicly available.

REFERENCES

Maximilian Augustin, Yannic Neuhaus, and Matthias Hein. Dash: Detection and assessment of systematic hallucinations of vlms. In *ICCV*, 2025.

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

Shuai Bai, Keqin Chen, Xuejing Liu, Jialin Wang, Wenbin Ge, Sibo Song, Kai Dang, Peng Wang, Shijie Wang, Jun Tang, Humen Zhong, Yuanzhi Zhu, Mingkun Yang, Zhaohai Li, Jianqiang Wan, Pengfei Wang, Wei Ding, Zheren Fu, Yiheng Xu, Jiabo Ye, Xi Zhang, Tianbao Xie, Zesen Cheng, Hang Zhang, Zhibo Yang, Haiyang Xu, and Junyang Lin. Qwen2.5-vl technical report, 2025b. URL https://arxiv.org/abs/2502.13923.

Liang Chen, Weichu Xie, Yiyan Liang, Hongfeng He, Hans Zhao, Zhibo Yang, Zhiqi Huang, Haoning Wu, Haoyu Lu, Y. charles, Yiping Bao, Yuantao Fan, Guopeng Li, Haiyang Shen, Xuanzhong Chen, Wendong Xu, Shuzheng Si, Zefan Cai, Wenhao Chai, Ziqi Huang, Fangfu Liu, Tianyu Liu, Baobao Chang, Xiaobo Hu, Kaiyuan Chen, Yixin Ren, Yang Liu, Yuan Gong, and Kuan Li. Babyvision: Visual reasoning beyond language, 2026. URL https://arxiv.org/abs/2601.06521.

Lin Chen, Jinsong Li, Xiaoyi Dong, Pan Zhang, Yuhang Zang, Zehui Chen, Haodong Duan, Jiaqi Wang, Yu Qiao, Dahua Lin, et al. Are we on the right way for evaluating large vision-language models? In *NeurIPS*, 2024.

Google DeepMind. Gemini 3 Flash: frontier intelligence built for speed, December 2025. URL https://blog.google/products-and-platforms/products/gemini/gemini-3-flash/.

Google DeepMind. Gemini 3.5: frontier intelligence with action, May 2026a. URL https://blog.google/innovation-and-ai/models-and-research/gemini-models/gemini-3-5/.

Google DeepMind. Introducing Gemini 3.7 Flash, August 2026b. URL https://blog.google/innovation-and-ai/models-and-research/gemini-models/introducing-gemini-3-7-flash/.

Tianrui Guan, Fuxiao Liu, Xiyang Wu, Ruiqi Xian, Zongxia Li, Xiaoyu Liu, Xijun Wang, Lichang Chen, Furong Huang, Yaser Yacoob, et al. Hallusionbench: an advanced diagnostic suite for entangled language hallucination and visual illusion in large vision-language models. In *CVPR*, 2024.

Jack Hong, Chenxiao Zhao, ChengLin Zhu, Weiheng Lu, and Guohai Xu. Deepeyesv2: Toward agentic multimodal model. In *ICLR*, 2026.

Wenyi Hong, Wenmeng Yu, Xiaotao Gu, Guo Wang, Guobing Gan, Haomiao Tang, Jiale Cheng, Ji Qi, Junhui Ji, Lihang Pan, et al. Glm-4.5 v and glm-4.1 v-thinking: Towards versatile multimodal reasoning with scalable reinforcement learning. *arXiv*, 2025.

Yushi Hu, Reyhane Askari-Hemmat, Melissa Hall, Emily Dinan, Luke Zettlemoyer, and Marjan Ghazvininejad. Multimodal rewardbench 2: Evaluating omni reward models for interleaved text and image. In *CVPR*, 2026.

Ailin Huang, Chengyuan Yao, Chunrui Han, Fanqi Wan, Hangyu Guo, Haoran Lv, Hongyu Zhou, Jia Wang, Jian Zhou, Jianjian Sun, et al. Step3-vl-10b technical report. *arXiv*, 2026a.

Wenxuan Huang, Bohan Jia, Shaosheng Cao, Zheyu Ye, Zhe Xu, Yao Hu, Shaohui Lin, et al. Vision-r1: Incentivizing reasoning capability in multimodal large language models. In *ICLR*, 2026b.

Yiran Huang, Karsten Roth, Quentin Bouniot, Wenjia Xu, and Zeynep Akata. Dissecting multimodal in-context learning: Modality asymmetries and circuit dynamics in modern transformers. In *ICML*, 2026c.

Sanghwan Kim, Rui Xiao, Stephan Alaniz, Yongqin Xian, and Zeynep Akata. Training-free uncertainty guidance for complex visual tasks with mllms. In *ECCV*, 2026.

Alexander Kirillov, Eric Mintun, Nikhila Ravi, Hanzi Mao, Chloe Rolland, Laura Gustafson, Tete Xiao, Spencer Whitehead, Alexander C Berg, Wan-Yen Lo, et al. Segment anything. In *ICCV*, 2023.

Woosuk Kwon, Zhuohan Li, Siyuan Zhuang, Ying Sheng, Lianmin Zheng, Cody Hao Yu, Joseph E. Gonzalez, Hao Zhang, and Ion Stoica. Efficient memory management for large language model serving with pagedattention. In *SOSP*, 2023.

Xin Lai, Junyi Li, Wei Li, Tao Liu, Tianjian Li, and Hengshuang Zhao. Mini-o3: Scaling up reasoning patterns and interaction turns for visual search. In *ICLR*, 2026.

LAION. Releasing re-laion-5b: transparent iteration on laion-5b with additional safety fixes. `https://laion.ai/blog/relaion-5b/`, 2024.

Chaoyu Li, Eun Woo Im, and Pooyan Fazli. Vidhalluc: Evaluating temporal hallucinations in multimodal large language models for video understanding. In *CVPR*, 2025a.

Geng Li, Jinglin Xu, Yunzhen Zhao, and Yuxin Peng. Dyfo: A training-free dynamic focus visual search for enhancing lmms in fine-grained visual understanding. In *CVPR*, 2025b.

Lei Li, Yuancheng Wei, Zhihui Xie, Xuqing Yang, Yifan Song, Peiyi Wang, Chenxin An, Tianyu Liu, Sujian Li, Bill Yuchen Lin, Lingpeng Kong, and Qi Liu. Vl-rewardbench: A challenging benchmark for vision-language generative reward models. In *CVPR*, 2025c.

Rang Li, Lei Li, Shuhuai Ren, Hao Tian, Shuhao Gu, Shicheng Li, Zihao Yue, Yudong Wang, Wenhan Ma, Zhe Yang, Jingyuan Ma, Zhifang Sui, and Fuli Luo. Groundingme: Exposing the visual grounding gap in mllms through multi-dimensional evaluation. In *CVPR*, 2026a.

Yangfu Li, Hongjian Zhan, Jiawei Chen, Yuning Gong, Qi Liu, and Yue Lu. Deepscan: A training-free framework for visually grounded reasoning in large vision-language models. In *CVPR*, 2026b.

Yifan Li, Yifan Du, Kun Zhou, Jinpeng Wang, Xin Zhao, and Ji-Rong Wen. Evaluating object hallucination in large vision-language models. In *EMNLP*, 2023.

Yijun Liang, Ming Li, Chenrui Fan, Ziyue Li, Dang Nguyen, Kwesi Cobbina, Shweta Bhardwaj, Jiuhai Chen, Fuxiao Liu, and Tianyi Zhou. Colorbench: Can vlms see and understand the colorful world? a comprehensive benchmark for color perception, reasoning, and robustness. In *NeurIPS*, 2026.

Zichao Lin, Yifeng Xie, Bowen Qu, Haiming Wang, Jia Li, Haoning Wu, Yuhao Dong, Zuhao Yang, Jinguo Zhu, Haoyu Lu, Zijia Zhao, Tongtian Yue, Zhangyang Qi, Junwei Yang, Mengfan Dong, Peizhou Cao, Chenzhuang Du, Zaida Zhou, Haotian Yao, Hao Yang, Hongcheng Gao, Lin Sui, Weihong Li, Xinxing Zu, Jia Chen, Yao Wang, Xiaoxue Wu, Yalin Wang, Y. Charles, Yiping Bao, Yangyang Liu, Zhiqi Huang, and Xinyu Zhou. Perceptionbench: Evaluating atomic visual perception in multimodal large language models, 2026. URL `https://arxiv.org/abs/2607.24957`.

Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. In *ICLR*, 2019.

Fanqing Meng, Lingxiao Du, Zongkai Liu, Zhixiang Zhou, Quanfeng Lu, Daocheng Fu, Tiancheng Han, Botian Shi, Wenhai Wang, Junjun He, et al. Mm-eureka: Exploring the frontiers of multimodal reasoning with rule-based reinforcement learning. *arXiv*, 2025.

Atsuyuki Miyai, Jingkang Yang, Jingyang Zhang, Yifei Ming, Qing Yu, Go Irie, Yixuan Li, Hai Li, Ziwei Liu, and Kiyoharu Aizawa. Unsolvable problem detection: Robust understanding evaluation for large multimodal models. In *ACL*, 2025.

Yannic Neuhaus and Matthias Hein. Repope: Impact of annotation errors on the pope benchmark. *arXiv*, 2025.

OpenAI. Introducing GPT-5.1 for developers, November 2025. URL https://openai.com/index/gpt-5-1-for-developers/.

OpenAI. GPT-5.6: Frontier intelligence that scales with your ambition, July 2026. URL https://openai.com/index/gpt-5-6/.

Weijie Qiu, Dai Guan, Junxin Wang, Zhihang Li, Yongbo Gai, Mengyu Zhou, Erchao Zhao, Xiaoxi Jiang, and Guanjun Jiang. Rationale matters: Learning transferable rubrics via proxy-guided critique for vlm reward models. *arXiv*, 2026.

Qwen Team. Qwen-agent: Agent framework and applications built upon qwen, 2025. URL https://github.com/QwenLM/Qwen-Agent.

Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language supervision. In *ICML*, 2021.

Andrew Sellergren, Chufan Gao, Fereshteh Mahvar, Timo Kohlberger, Fayaz Jamil, Madeleine Traverse, Alberto Tono, Bashir Sadjad, Lin Yang, Charles Lau, et al. Medgemma 1.5 technical report. *arXiv*, 2026.

Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Xiao Bi, Haowei Zhang, Mingchuan Zhang, YK Li, Yang Wu, et al. Deepseekmath: Pushing the limits of mathematical reasoning in open language models. *arXiv*, 2024.

Haozhan Shen, Peng Liu, Jingcheng Li, Chunxin Fang, Yibo Ma, Jiajia Liao, Qiaoli Shen, Zilun Zhang, Kangjia Zhao, Qianqian Zhang, et al. Vlm-r1: A stable and generalizable r1-style large vision-language model. *arXiv*, 2025a.

Haozhan Shen, Kangjia Zhao, Tiancheng Zhao, Ruochen Xu, Zilun Zhang, Mingwei Zhu, and Jianwei Yin. Zoomeye: Enhancing multimodal llms with human-like zooming capabilities through tree-based image exploration. In *EMNLP*, 2025b.

Alex Su, Haozhe Wang, Weiming Ren, Fangzhen Lin, and Wenhu Chen. Pixel reasoner: Incentivizing pixel space reasoning via curiosity-driven reinforcement learning. In *NeurIPS*, 2026.

Jayant Sravan Tamarapalli, Rynaa Grover, Nilay Pande, and Sahiti Yerramilli. Countqa: How well do mllms count in the wild? *arXiv*, 2025.

Kimi Team, Tongtong Bai, Yifan Bai, Yiping Bao, Jianfeng Cai, Xinyuan Cai, Peizhou Cao, Yuxuan Cao, Ziwei Chai, Y Charles, et al. Kimi k3: Open frontier intelligence. *arXiv*, 2026.

Qwen Team. Qwen3.5: Towards native multimodal agents, February 2026. URL https://qwen.ai/blog?id=qwen3.5.

Xinyu Tian, Shu Zou, Zhaoyuan Yang, Mengqi He, Fabian Waschkowski, Lukas Wesemann, Peter Tu, and Jing Zhang. More thought, less accuracy? on the dual nature of reasoning in vision-language models. In *ICLR*, 2026.

Shengbang Tong, Ellis Brown, Penghao Wu, Sanghyun Woo, Manoj Middepogu, Sai C Akula, Jihan Yang, Shusheng Yang, Adithya Iyer, Xichen Pan, et al. Cambrian-1: A fully open, vision-centric exploration of multimodal llms. In *NeurIPS*, 2024.

Haochen Wang, Xiangtai Li, Zilong Huang, Anran Wang, Jiacong Wang, Tao Zhang, Sule Bai, Zijian Kang, Jiashi Feng, Zhuochen Wang, et al. Traceable evidence enhanced visual grounded reasoning: Evaluation and method. In *ICLR*, 2026a.

Junyang Wang, Yuhang Wang, Guohai Xu, Jing Zhang, Yukai Gu, Haitao Jia, Jiaqi Wang, Haiyang Xu, Ming Yan, Ji Zhang, et al. Amber: An llm-free multi-dimensional benchmark for mllms hallucination evaluation. *arXiv*, 2023.

Wenbin Wang, Liang Ding, Minyan Zeng, Xiabin Zhou, Li Shen, Yong Luo, Wei Yu, and Dacheng Tao. Divide, conquer and combine: A training-free framework for high-resolution image perception in multimodal large language models. In *AAAI*, 2025.

Zhecan Wang, Garrett Bingham, Adams Wei Yu, Quoc V. Le, Thang Luong, and Golnaz Ghiasi. Haloquest: A visual hallucination dataset for advancing multimodal reasoning. In *ECCV*, 2024.

Zhenhailong Wang, Xuehang Guo, Sofia Stoica, Haiyang Xu, Hongru Wang, Hyeonjeong Ha, Xiusi Chen, Yangyi Chen, Ming Yan, Fei Huang, et al. Perception-aware policy optimization for multimodal reasoning. In *ICLR*, 2026b.

Lai Wei, Liangbo He, Jun Lan, Lingzhong Dong, Yutong Cai, Siyuan Li, Huijia Zhu, Weiqiang Wang, Linghe Kong, Yue Wang, Zhuosheng Zhang, and Weiran Huang. Zooming without zooming: Region-to-image distillation for fine-grained multimodal perception. In *ICML*, 2026a.

Yana Wei, Hongbo Peng, Yanlin Lai, Liang Zhao, Kangheng Lin, En Yu, Keyu Lv, Han Zhou, Yin Tang, Haodong Li, et al. Perceptionrubrics: Calibrating multimodal evaluation to human perception. In *ICML*, 2026b.

Penghao Wu and Saining Xie. V*: Guided visual search as a core mechanism in multimodal llms. In *CVPR*, 2024.

Rui Xiao, Sanghwan Kim, Mariana-Iuliana Georgescu, Zeynep Akata, and Stephan Alaniz. Flair: Vlm with fine-grained language-informed image representations. In *CVPR*, 2025.

Rui Xiao, Sanghwan Kim, Yongqin Xian, Zeynep Akata, and Stephan Alaniz. Finer: Mllms hallucinate under fine-grained negative queries. In *CVPR*, 2026.

Zhongxing Xu, Chengzhi Liu, Qingyue Wei, Juncheng Wu, James Zou, Xin Wang, Yuyin Zhou, and Sheng Liu. More thinking, less seeing? assessing amplified hallucination in multimodal reasoning models. In *NeurIPS*, 2026.

Qianqi Yan, Xuehai He, Xiang Yue, and Xin Eric Wang. Worse than random? an embarrassingly simple probing evaluation of large multimodal models in medical vqa. In *Findings of ACL*, 2025.

En Yu, Kangheng Lin, Liang Zhao, Yana Wei, Yuang Peng, Haoran Wei, Jianjian Sun, Chunrui Han, Zheng Ge, Xiangyu Zhang, et al. Perception-r1: Pioneering perception policy with reinforcement learning. In *NeurIPS*, 2026a.

Qiying Yu, Zheng Zhang, Ruofei Zhu, Yufeng Yuan, Xiaochen Zuo, Yu Yue, Weinan Dai, Tiantian Fan, Gaohong Liu, Lingjun Liu, et al. Dapo: An open-source llm reinforcement learning system at scale. In *NeurIPS*, 2026b.

Qianhao Yuan, Jie Lou, Xing Yu, Hongyu Lin, Le Sun, Xianpei Han, and Yaojie Lu. Vision-opd: Learning to see fine details for multimodal llms via on-policy self-distillation. *arXiv*, 2026.

Xiaohua Zhai, Basil Mustafa, Alexander Kolesnikov, and Lucas Beyer. Sigmoid loss for language image pre-training. In *ICCV*, 2023.

Chi Zhang, Haibo Qiu, Qiming Zhang, Yufei Xu, Zhixiong Zeng, Siqi Yang, Peng Shi, Lin Ma, and Jing Zhang. Perceptual-evidence anchored reinforced learning for multimodal reasoning. In *CVPR*, 2026a.

Yi-Fan Zhang, Xingyu Lu, Shukang Yin, Chaoyou Fu, Wei Chen, Xiao Hu, Bin Wen, Kaiyu Jiang, Changyi Liu, Tianke Zhang, et al. Thyme: Think beyond images. In *ICLR*, 2026b.

Zongmeng Zhang, Wengang Zhou, Jie Zhao, and Houqiang Li. Robust multimodal large language models against modality conflict. In *ICML*, 2025.

Ziwei Zheng, Minghao Yang, Jack Hong, Chenxiao Zhao, Guohai Xu, Le Yang, and Chao Shen. Deepeyes: Incentivizing" thinking with images" via reinforcement learning. In *ICLR*, 2026.

Yanxu Zhu, Shitong Duan, Xiangxu Zhang, Jitao Sang, Peng Zhang, Tun Lu, Xiao Zhou, Jing Yao, Xiaoyuan Yi, and Xing Xie. Mohobench: Assessing honesty of multimodal large language models via unanswerable visual questions. In *AAAI*, 2026.

A VERVE BENCHMARK CONSTRUCTION DETAILS

This section details the construction pipeline and rigorous filtering applied to create the VERVE benchmark. We detail the methodology in two primary components: the generation of the multi-choice split (VERVE-MCQ) and the derivation of the binary diagnostic split (VERVE-BINARY).

A.1 VERVE-MCQ

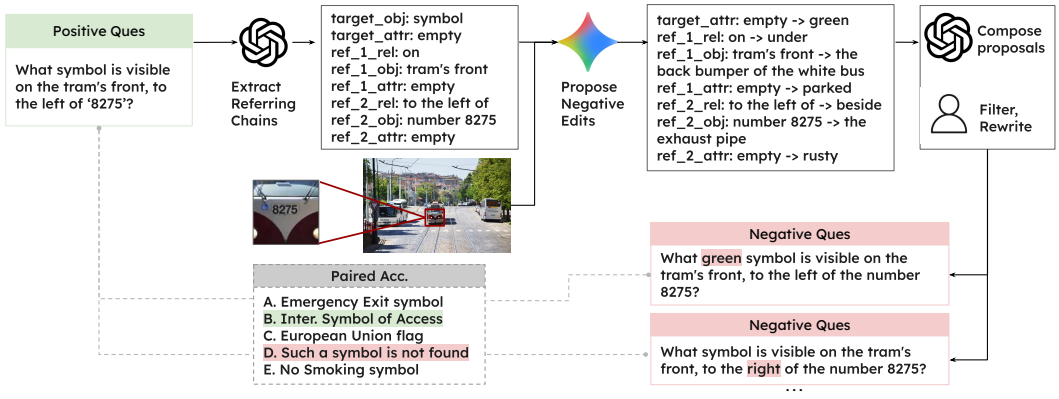

Figure 5: VERVE-MCQ construction pipeline.

As shown in Fig. 5, we construct VERVE-MCQ by converting positive visual search questions into one or more minimally edited negative twins.

Positive Question Selection We collect original samples from ZoomBench (Wei et al., 2026a) and TreeBench (Wang et al., 2026a), yielding an initial pool of 1,250 images paired with one question each. Human annotators filter out 53.3% of these questions, leaving 584 images with verified positive questions. Fig. 5 illustrates an example: an image showing a tram in a street view is selected with the positive question "What symbol is visible on the tram's front, to the left of '8275'?"

Extracting Referring Chains We parse each positive question into the object-anchored chain defined in Eq. 1. Each chain contains a target object and one or more reference anchors, with editable slots for objects, attributes, and relations. Using GPT-5.1 with the prompt in App. H, we generate referring chains for all positive queries. This structure lets us edit one visual condition at a time while keeping the rest of the question intact. For example, in Fig. 5, "symbol" is identified as the target, and "tram's front" serves as the first reference anchor (`ref_1_obj`).

Proposing Negative Edits For each filled slot, we prompt Gemini-3-Flash with the image, original question, and parsed chain to propose a plausible replacement not supported by the image. A valid edit modifies a single object, attribute, or relation, making the question unanswerable under the original choices. For instance, changing "to the left of '8275"' to "to the right of '8275"' is valid only if the image lacks the queried symbol at the edited position. If no valid edit can be found for a slot, it remains unchanged. Overall, this pipeline generates 2,736 negative edit proposals, averaging 4.68 per positive question.

Human Review Automated proposals can still produce invalid edits. In Fig. 5, for example, Gemini proposed changing "symbol to the left of 8275" to "symbol beside 8275," which fails as a negative edit because both descriptions fit the image. To address this, two human annotators manually review all proposals to ensure that each edit modifies only the intended condition, is not satisfied elsewhere in the image, and correctly maps to the premise-denial answer option. Annotators first discard 977 invalid proposals, leaving 1,759 (64.2% of the initial pool), and then refine or rewrite 617 of the remaining edits (35.1%). Annotators cross-check each other's work and resolve disagreements through further edits, yielding an inter-annotator agreement rate of 95.8%.

Pairing with Positive Twins Accepted negative edits are reassembled into MCQs using GPT-5. A negative twin shares the same image and answer choices as its positive counterpart, but its correct

answer changes to the premise-denial option (e.g., "Such a symbol is not found" in Fig. 5). This option is included in positive questions as well, ensuring its presence does not reveal whether a query is positive or negative. As a result, each positive and negative pair is distinguished solely by the edited visual premise.

Benchmark Distribution We measure the distribution of question pairs across edit locations ("Target", "1-Hop", and "2-Hop") and edit types within each location ("object", "attribute", and "relation"). Fig. 6 visualizes this breakdown. The distribution shows that the majority of edits in VERVE-MCQ remain within the "Target" and "1-Hop" categories, demonstrating that the benchmark does not artificially inflate difficulty by relying heavily on multi-hop samples. Instead, our goal is to show that frontier models can fail even on simple, coarse-grained negative edits.

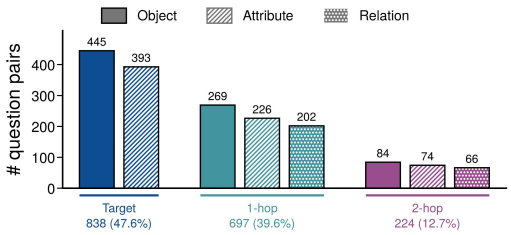

Figure 6: Number of samples over question category on VERVE-MCQ

A.2 VERVE-BINARY

VERVE-BINARY is derived from the edit history of VERVE-MCQ. For each accepted negative edit, we construct a pair of binary verification questions: one for the original visual premise and one for the edited (negative) visual premise. The positive binary question asks whether the original phrase can be seen in the image, while the negative binary question asks whether the edited phrase can be seen.

Using the example in Fig. 5, the MCQ edit from "to the left of the number 8275" to "to the right of the number 8275" produces a binary pair that directly tests this relation. The positive binary question should be asking "Can you see the symbol to the left of number 8275" while the negative binary question should be asking "Can you see the symbol to the right of number 8275". In this way, VERVE-BINARY removes the multiple-choice answer selection step, isolates whether the model can verify the edited visual condition itself.

Since VERVE-BINARY is derived from the human-reviewed VERVE-MCQ, its quality is inherently high. To refine it further, we used Gemini-3.7-Flash to inspect all binary questions, identifying 201 samples where the model disagreed with the answer key. Human annotators manually reviewed these cases, removed some of them and updated 70 of them, yielding a final set of 1,680 paired binary questions for VERVE-BINARY. All binary questions were verified to ensure they preserve the intended edit from VERVE-MCQ, can be answered strictly as yes/no questions, and introduce no extra ambiguity. We use this diagnostic split to test whether a model fails on VERVE-MCQ because it overlooks an edit within a complex query, or because it fails to recognize the edited visual condition in isolation.

B TRAINING DETAILS

Training Configuration We train VERVE-TRAIN on our curated 57K training set for one epoch. For Qwen2.5-VL-7B, Qwen3-VL-8B, Qwen3.5-9B, we train these three backbones under the same configurations. The detailed configuration is documented in Tab. 6.

Reward Hyperparameters Regarding the reward hyperparameters (α, w_f) in Eq. 5, we fix $w_f = 0.1$ across all three backbones (Qwen2.5-VL-7B, Qwen3-VL-8B, and Qwen3.5-9B). We set $\alpha = 0.5$ when training Qwen2.5-VL-7B and Qwen3-VL-8B. However, when training Qwen3.5-9B, we observed early saturation of R_k due to the stronger inherent capabilities of the backbone. Because such saturation can hinder learning, we found that applying an annealing schedule to α significantly boosts performance. Details are provided in App. D.10.

Examples of Training Data Figure 7 visualizes representative examples from our training data construction pipeline (Sec. 3.1). Specifically, we show one positive query sample alongside three distinct negative query types corresponding to object, attribute, and relation edits proposed by GPT-

Config	Qwen2.5-VL-7B	Qwen3-VL-8B	Qwen3.5-9B
Training data	57,447		
Tuned parameters	all (vision tower unfrozen)		
RL algorithm	GRPO (Shao et al., 2024)		
Rollouts per prompt G	8		
Rollout temperature / top-p	1.0 / 1.0		
Prompts per step	384		
Global BS	96 prompts (768 responses)		
Total epochs	1 (150 steps)		
Optimizer	AdamW (Loshchilov & Hutter, 2019), β=$(0.9, 0.999)$		
Learning rate	1×10^{-6}		
LR scheduler	constant		
Warm up ratio	0		
Weight decay	1×10^{-2}		
Gradient clipping	1.0		
KL coefficient	0		
Clip ratio (low / high)	0.2 / 0.3		
Max prompt / response length	25,000 / 2,048 tokens		
Image pixel budget (min / max)	512^2 / 4096^2		
Precision	bf16		

Table 6: Hyper-parameters for VERVE-TRAIN

OSS-120B. For self-verification training (Sec. 3.1), we also construct ground-truth verification checks used to compute the coverage reward R_k; these reference checks are shown in the rightmost column.

B.1 DISCUSSION ON TRAIN-TEST DISTRIBUTION

In this paper, we took three steps to rule out risks for any potential in-distribution leakage between our training set and test set.

Image Deduplication Our 57K training samples are drawn from the ZwZ-RL training set (Wei et al., 2026a). The benchmark images come from ZoomBench (Wei et al., 2026a) and TreeBench (Wang et al., 2026a), where ZwZ-RL was already deduplicated against ZoomBench by its authors.

Separate Generation Pipelines The training data and the benchmark are built by completely different pipelines with different models and prompts. For training, GPT-OSS-120B proposes negative questions and Qwen3.5-27B verifies them against the image (Sec. 3.1). For the benchmark, GPT-5 parses the referring chain, Gemini-3-Flash proposes the edits, and GPT-5 assembles the final questions (App. A.1). The benchmark also passes through heavy human review: annotators discard 977 of the 2,736 proposed edits and rewrite 617 of the remaining ones, which moves the benchmark distribution well away from any automatic pipeline (see App. A.1 for details).

Gains on the Held-out Source Finally, we validate this on detailed breakdowns of our VERVE-MCQ shown in Tab. 13. VERVE-TRAIN significantly improves p $\text{Acc}_{\text{paired}}$ on the subset whose image source is TreeBench. For example from 11.0% to 36.1% on Qwen3.5-9B. If the improvement came from in-distribution exposure, it would not appear on this subset (at least not with this scale).

C EVALUATION DETAILS

C.1 EVALUATED BENCHMARKS

In addition to our VERVE-MCQ benchmark and the diagnostic VERVE-BINARY suite, we evaluate models across three broad benchmark categories: eight perception benchmarks, six hallucination benchmarks, and seven cross-task transfer benchmarks. Here, we briefly introduce each benchmark



Figure 7: Visualization of training data. We visualize one positive training samples and three negative samples, each negative sample represents one edit type (Obj, Attr or Rel). The correct option is underlined. Note that self-verifications only serves as extra labels for reward modeling in the OPSV training and are not used in plain training.

evaluated in Sec. 4.3 and Sec. 4.4. Regarding the metrics reported for each benchmark in Table 2 and Table 3, we use micro-averaged accuracy unless otherwise specified.

Perception Benchmarks To evaluate fine-grained visual perception capabilities, we benchmark performance across eight distinct visual understanding tasks:

- **V* (Wu & Xie, 2024):** Evaluates small-detail localization within cluttered scenes using 191 high-resolution (2K) images with multiple-choice questions (MCQs) targeting spatial relations and fine object attributes.
- **HR-Bench-4K and HR-Bench-8K (Wang et al., 2025):** Pushes visual limits with ultra-high-resolution images (4K and 8K) over four-way MCQs covering single-object perception (attributes, OCR) and multi-object spatial reasoning.
- **CV-Bench (Tong et al., 2024):** Comprises 2,638 MCQs assessing fundamental 2D/3D visual cues, including object counting, spatial relations, depth ordering, and relative distance estimation.
- **ColorBench (Liang et al., 2026):** Isolates color perception across varied tasks, including color recognition, extraction, comparison, and specialized optical illusion tests.
- **CountQA (Tamarapalli et al., 2025):** Evaluates exact object counting (ranging from 0 to 400 instances) in real-world photographs using verified exact-match numerical ground truth.

- **BabyVision (Chen et al., 2026):** Probes foundational visual reasoning skills-such as visual tracking, pattern recognition, and spatial perception-across 388 mixed-format evaluation items.
- **PerceptionBench (Lin et al., 2026):** Isolates atomic visual perception from high-level reasoning using 3,000 short-answer questions constructed from failure cases of frontier models across ten perceptual dimensions.

Hallucination Benchmarks To measure robustness against visual hallucinations, we evaluate models across six established evaluation suites:

- **FINER-CompreCap** (Xiao et al., 2026): Assesses fine-grained negative queries across four distinct settings: multi-object, multi-attribute, multi-relation, and open-ended "What" queries. In Tab. 2, we report the micro-averaged paired accuracy ($\text{Acc}_{\text{paired}}$) across all subsets.
- **RePOPE** (Neuhaus & Hein, 2025): A cleaned, re-annotated variant of POPE (Li et al., 2023) evaluating binary object existence on COCO images, eliminating benchmark label noise and ambiguity.
- **DASH-B** (Augustin et al., 2025): Extends binary object-existence evaluation to web-scale scenes from ReLAION-5B (LAION, 2024), querying present versus absent objects verified by human annotators.
- **HallusionBench** (Guan et al., 2024): Disentangles visual illusions from language prior hallucinations using paired original and human-edited images evaluated under identical binary questions; we report (fAcc) in Tab. 2.
- **AMBER** (Wang et al., 2023): Covers existence, attribute, and relation hallucinations across 1,004 images; we utilize its discriminative subset comprising 14,216 template-generated binary questions.
- **HaloQuest** (Wang et al., 2024): Evaluates open-ended queries designed to elicit hallucinations via false premises, missing context, or high visual ambiguity, graded via an LLM judge.

Cross-Task Transfer Benchmarks To evaluate cross-task transferability, we conduct experiments across seven distinct downstream benchmarks:

- **MMStar** (Chen et al., 2024): A multi-modal benchmark comprising 1,500 human-selected multiple-choice questions designed to prevent data leakage, spanning six core capabilities from coarse visual perception to mathematical reasoning.
- **PerceptionRubrics** (Wei et al., 2026b): Evaluates fine-grained image captioning across 1,038 images, using human-designed rubrics that pair essential visual facts with easily missed details. We follow the benchmark's protocol and report mean gated rubric score in Tab. 3, the fraction of easy-wrong rubric items covered by the caption, counted only if all must-right items are covered.
- **VL-RewardBench** (Li et al., 2025c): Assesses model capability as a multimodal reward model over 1,250 human-verified pairwise response comparisons spanning general, hallucination, and reasoning categories.
- **MMRB2** (Hu et al., 2026): Evaluates multimodal reward modeling across 4,000 expert-annotated pairwise judgments covering text-to-image generation, image editing, interleaved generation, and complex reasoning.
- **GroundingME** (Li et al., 2026a): Tests visual grounding performance on high-resolution images from SA-1B (Kirillov et al., 2023) and HR-Bench under complex instructions, featuring hard instances with spatial occlusion, tiny objects, and non-existent target rejection. We report grounding accuracy at IoU > 0.5 in Tab. 3, where rejection items are correct only if no box is returned.
- **ProbMed-1K** (Yan et al., 2025): Probes medical VQA robustness across X-ray, CT, and MRI modalities using paired positive and adversarial negative questions. To maintain statistical significance while reducing compute overhead, we construct a balanced, stratified subset of 1,000 question pairs sampled across all diagnostic categories.

Table 7: Generation parameters during inference.

Parameter	Value
Inference engine	vLLM 0.11 (Kwon et al., 2023)
Temperature	0.7
Top-p / top-k	0.8 / 20
Presence penalty	1.5
repetition penalty	1.0
Max. output tokens	8,192
Context length	32,768 tokens
Image budget	65,536-16,777,216 pixels
Random seed	42
LLM judge	GPT-OSS-120B (temperature 0)

- **VidHalluc** (Li et al., 2025a): Measures temporal hallucination in video domain tasks by pairing visually distinct yet semantically similar video sequences to evaluate action ordering and scene transition queries.

C.2 Generation Parameters

To ensure a fair evaluation, we adopt the standard generation parameters recommended for the Qwen-VL family and evaluate both our VERVE models and baseline systems using identical configurations and random seeds. For agentic architectures (e.g., Thyme, DeepEyes, and DeepEyes-v2), we preserve each model's native inference settings, as empirical observation indicates that altering these parameters can lead to suboptimal performance. A complete overview of the evaluation parameters is detailed in Table 7.

D Additional Experiments

D.1 Full results on VERVE-Binary

In Sec. 2.2, we reported only the overall average $\text{Acc}_{\text{paired}}$ for VERVE-BINARY, demonstrating that SOTA open-source methods such as TreeVGR, ZwZ, and Vision-OPD lag behind their base models. For a detailed breakdown, the full evaluation is provided in Tab. 8. Beyond the overall $\text{Acc}_{\text{paired}}$ across all samples, because VERVE-BINARY derives object, attribute, and relation edits from VERVE-MCQ, we also compute fine-grained accuracy breakdowns across these three edit types. We also report model's accuracy when the ground-truth of the question is "yes" or the ground-truth of the question is "no", as well as (macro) average of the two. Our VERVE variants excel not only in paired accuracy but also in terms of averaged accuracy split by ground-truth type, proving that we are enhancing discriminativeness of the models instead of constraining to negative or positive bias.

D.2 Qualitative Results for Verve Benchmarks

Fig. 8 shows paired examples from VERVE-MCQ. On the positive queries, all three models agree with the correct answer. The negative queries change a single word in the query, such as "sign" to "digital display", "side" to "roof", or "car" to "truck". Gemini-3.7-Flash and Vision-OPD-9B fail in the negative cases and still return the answer that matches the visible text or object, for example "TAKEAWAY ONLY" when asked about a digital display that does not exist. VERVE-9B checks the query against the image and selects the "not found" option in all three cases. The failures of the baselines are not perception errors, since they read the text correctly in the positive case. They come from answering without verifying the premise of the question.

Fig. 9 shows the same pattern on VERVE-BINARY. The negative queries replace one attribute, such as "under" with "left of", "leaf" with "animal", or "bell tower" with "clock tower". Both baselines answer "yes" to every negative query, even though they answer the positive version correctly. VERVE-9B answers "No" on all three negatives while keeping the positive answers correct.

D.3 Qualitative Results on Hallucination Benchmarks

Fig. 10 shows one example from each of the six hallucination benchmarks, comparing VERVE-9B with its base model Qwen3.5-9B and with Vision-OPD-9B. Most of these questions mention an object

Table 8: Detailed breakdown results on VERVE-BINARY. Macro is the mean of GT=yes and GT=no accuracy. **Best** and second-best within each backbone are marked for paired accuracy and Macro; GT=yes and GT=no are left unmarked because a strongly biased model can score high on one of them alone.

	Paired accuracy				*Accuracy by GT*		
Method	**Avg**	obj	attr	rel	GT=yes	GT=no	**Avg.**
Gemini-3.7-Flash	85.65	90.34	87.74	64.26	94.54	90.09	92.32
Qwen2.5-VL-7B	70.65	77.32	74.59	37.45	77.77	92.50	85.14
+TreeVGR	57.08	72.42	51.27	22.98	97.54	53.85	75.70
+DeepEyes	44.94	59.66	39.31	12.34	95.34	40.32	67.83
+DeepEyes-V2	20.24	25.64	18.39	7.66	99.89	10.25	55.07
+Thyme	66.55	74.74	69.66	30.64	94.75	68.25	81.50
+Ours *plain*	74.94	78.35	79.37	**51.06**	79.06	96.18	87.62
+Ours *verify*	**79.52**	**84.41**	**86.10**	44.68	91.38	86.67	**89.03**
Qwen3VL-8B	77.92	83.51	81.32	49.79	92.02	83.59	87.81
+ZwZ	55.48	63.79	56.65	24.68	99.14	50.50	74.82
+Ours *plain*	**83.93**	**87.89**	**87.00**	62.13	91.43	91.63	**91.53**
+Ours *verify*	81.96	84.66	85.20	**63.83**	90.14	90.76	90.45
Qwen3.5-9B	73.15	82.09	75.93	35.74	95.50	74.61	85.06
+Vision-OPD	59.46	72.16	59.04	18.72	99.25	55.12	77.19
+Ours *plain*	86.55	88.66	89.24	71.91	90.57	95.85	93.21
+Ours *verify*	**87.44**	**89.18**	**90.28**	**73.62**	93.09	93.90	**93.50**



Figure 8: VERVE-MCQ qualitative results.

or attribute that is not in the image, such as an oven in a dining room, a freight car in a top-down driving scene, a white glove on a referee holding a yellow one, or the torch that the Statue of Liberty is not holding. Vision-OPD-9B accepts the false premise in every case, and Qwen3.5-9B accepts it in four of the six. VERVE-9B rejects the false premise in all six and, on FINER-CompreCap, also picks the correct replacement (window frame instead of mirror frame). The HallusionBench example is the reverse situation: the red shape is indeed a square, and both baselines say no while VERVE-9B says yes. This shows that the model is not simply biased toward negative answers but is checking the claim against the image in both directions.

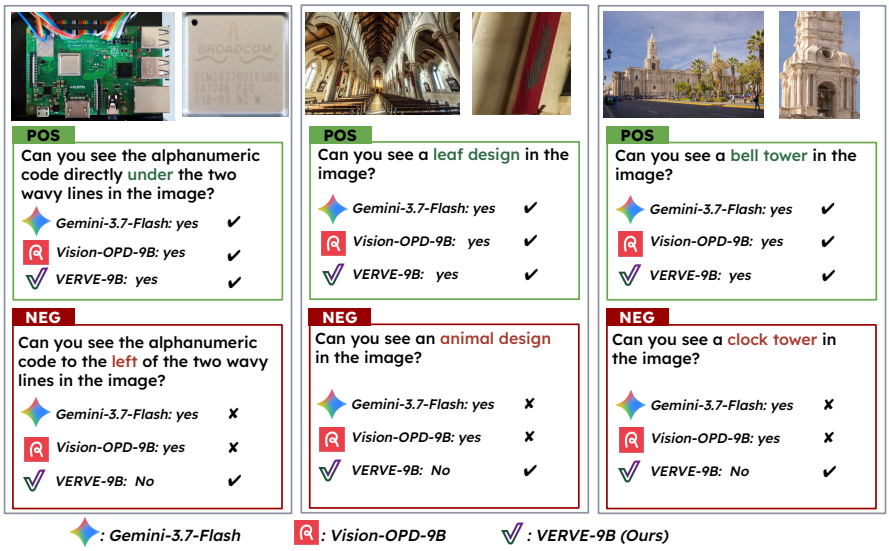

Figure 9: VERVE-Binary qualitative results.



Figure 10: Qualitative results on hallucination benchmarks. Only the final prediction is visualized, while intermediate reasoning traces are omitted.

D.4 Per-Category Breakdown of Cross-Task Transfer

GroundingME and VL-RewardBench each have a category that directly matches our training objective: GroundingME has a *Rejection* subset of non-existent targets, and VL-RewardBench has a *Hallucination* category. We therefore break both benchmarks down by category in Tab. 9 for Qwen3.5-9B family. As expected, VERVE-Train improves these two categories: Rejection rises from 0.0 to 13.9 (plain) and 27.4 (OPSV), and Hallucination improves by 16.0 and 15.6 points. Notably, the gains also extend to general capabilities. On GroundingME, Discriminative accuracy improves by 10.8 and 7.9 points and Spatial accuracy by 3.3 and 14.7 points. On VL-RewardBench, Reasoning improves by 11.4 and 14.5 points and General by 12.7 and 12.2 points. Overall, VERVE-Train does more than teach the model to reject false premises; it strengthens the model's visual understanding more broadly.

D.5 Qualitative Results on Cross-task Transfer

Fig. 11 shows examples from the cross-task benchmarks, none of which were specifically optimized in our training. On MMStar, VERVE-9B reads the ruler correctly and answers 7 cm, while Qwen3.5-9B and Vision-OPD-9B both answer 6. On ProbMed-1K, all three models detect pneumonia, but only

Table 9: Per-category breakdown of GroundingME (Li et al., 2026a) and VL-RewardBench (Li et al., 2025c) for the Qwen3.5-9B family.

	GroundingME					VL-RewardBench				
Model	Discr.	Spatial	Limited	Reject.	All	Halluc.	Reason.	General	Overall	Macro
Qwen3.5-9B	58.3	36.0	54.3	0.0	38.8	67.0	67.2	45.3	63.9	59.8
+Vision-OPD	59.3	34.0	42.7	0.0	34.9	71.7	74.1	54.7	69.9	66.8
+VERVE$_{plain}$	**69.1**	39.3	**59.7**	13.9	46.4	**83.0**	78.6	**58.0**	78.3	73.2
+VERVE$_{opsv}$	66.2	**50.7**	54.0	**27.4**	**50.2**	82.6	**81.7**	57.5	**78.8**	**73.9**

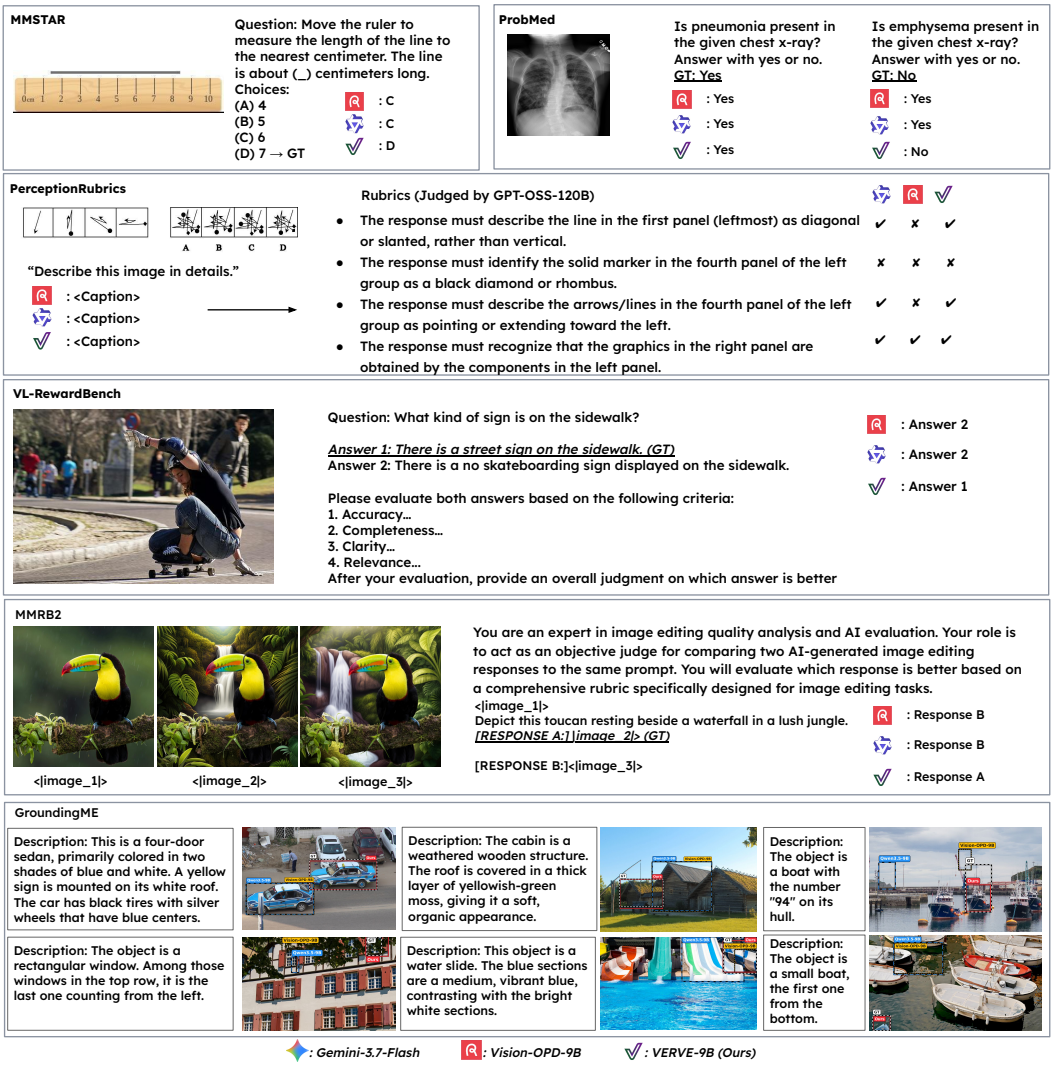

Figure 11: Qualitative results on cross-task transfer.

VERVE-9B correctly rejects emphysema, which is the same premise-checking behavior we see on the hallucination benchmarks. On PerceptionRubrics, VERVE-9B satisfies three of the four rubric items, matching the base model and beating Vision-OPD-9B, which fails on the two items about line direction. On VL-RewardBench and MMRB2, both baselines prefer the wrong candidate. In the VL-RewardBench case, they are drawn to the more specific but incorrect "no skateboarding sign", whereas VERVE-9B selects the answer that is actually supported by the image. On GroundingME, the descriptions are long and rely on attributes and ordinal positions such as "the last window from

the left" or "the first boat from the bottom". The baselines often localize a plausible object of the right category but the wrong instance, while VERVE-9B's boxes overlap the ground truth more closely. Across these tasks, the gain comes from the same habit: the model verifies each part of the query against the image instead of committing to the first plausible match.

D.6 Example Reasoning Trace

We show in Fig. 12 an illustrative example of the reasoning traces generated by our models compared to other state-of-the-art open-source baselines on a VERVE-MCQ negative question. As demonstrated, the visual prompt asks the model to identify a symbol on a "bench," introducing a false premise since no such bench is present in the image. While baseline models (including Qwen3.5-9B, DeepEyesV2, and Visio-nOPD) fail to detect this deception and incorrectly focus on a yellow square located on an overhead sign, both our plain and verify models successfully identify the missing object and correctly reject the premise.

D.7 Attention Analysis

Having analyzed the primary failure modes on VERVE-MCQ, we investigated an intriguing empirical anomaly: perception-focused methods such as ZwZ (Wei et al., 2026a) underperform their backbone model, Qwen3-VL-8B (Bai et al., 2025a). Further error analysis reveals that ZwZ's performance drop stems predominantly from negative questions, where it exhibits severe discriminative hallucination.

To uncover the internal mechanisms behind this behavior, we analyze and compare the attention allocation patterns of both models during reasoning. Specifically, we evaluate inference traces on two benchmark subsets: positive questions and TARGET-OBJ negative questions. Using the annotated target bounding boxes-which serve strictly as post-hoc ground truth coordinates and are never provided to the model during inference-we measure the attention weight assigned by each newly generated token to either the entire image token sequence or specifically to the target bounding box tokens.

Quantifying Attention Allocation in Generation For a given instance, let $I \subset P$ represent the image tokens within the prompt sequence P, $B \subset I$ denote the subset of image tokens whose spatial coordinates fall within the target bounding box, and $S = \{0\}$ designate the initial token serving as the attention sink. For layer ℓ, head h, and generation step t, let $a_t^{\ell,h}$ be the query attention row corresponding to the generated output token y_t. The head-averaged attention mass directed to a spatial region R is defined as:

$$m_t^{\ell}(R) \;=\; \frac{1}{H}\sum_{h=1}^{H}\sum_{j\in R} a_{t,j}^{\ell,h}. \tag{6}$$

Step $t = 0$ corresponds to the final prompt token emitting the first generated token (the *prefill* phase), while steps $t \geq 1$ represent the generated reasoning trace.

Given a temporal window of steps W and a subset of network layers L, we report two main metrics:

$$\textbf{Whole-Image Attention:} \quad A_{\text{img}}(W) = \frac{1}{|L|}\sum_{\ell\in L}\frac{\sum_{t\in W} m_t^{\ell}(I)}{\sum_{t\in W}\left(m_t^{\ell}(P) - m_t^{\ell}(S)\right)}, \tag{7}$$

$$\textbf{Bounding-Box Concentration:} \quad A_{\text{bbox}}(W) = \frac{1}{|L|}\sum_{\ell\in L}\frac{\sum_{t\in W} m_t^{\ell}(B)}{\sum_{t\in W} m_t^{\ell}(I)}. \tag{8}$$

Here, A_{img} quantifies the proportion of non-sink prompt attention allocated to image tokens. The initial sink token S is excluded from the denominator to prevent skewing, as over half of total prompt attention routinely settles on S. Conversely, A_{bbox} measures the relative fraction of image-bound attention concentrated specifically within the target box, making it invariant to prompt length and sink dynamics. Attention masses are aggregated across window W prior to division to prevent steps with negligible visual attention from distorting the ratio. In practice, we evaluate middle-layer representations across $L = \{13, \ldots, 24\}$ (out of 36 layers) and compute mean metric trajectories across deciles of the generated reasoning trace.

Table 10: Results on hallucination benchmarks across two backbones, comparing our method against existing hallucination-reduction approaches, with **best** and second-best scores marked per column within each backbone. Benchmarks include FINER-CompreCap (Xiao et al., 2026), RePOPE (Neuhaus & Hein, 2025), DASH (Augustin et al., 2025), HallusionBench (Guan et al., 2024), AMBER Discriminative (Wang et al., 2023), and HaloQuest (Wang et al., 2024).

Method	FINER.	RePOPE	DASH	HallBench	AMBER	HaloQuest
Qwen2.5-VL-7B	54.5	92.4	81.7	43.3	86.9	75.0
+FINER-Tuning	59.2	92.8	82.3	**46.5**	87.5	80.8
+Ours *plain*	**63.0**	93.5	**83.5**	43.9	87.4	**84.2**
+Ours *verify*	62.8	**93.9**	80.4	44.2	**88.1**	81.5
Qwen3-VL-8B	67.5	94.0	75.1	57.5	87.3	75.0
+FINER-Tuning	69.9	94.4	77.4	57.5	86.5	77.8
+Ours *plain*	70.5	94.7	84.2	62.1	87.5	**85.8**
+Ours *verify*	**72.1**	**95.1**	**84.4**	**62.7**	**87.9**	82.3

Findings: Fine-Grained Focus vs. Rigid Tunnel Vision Figure 13 summarizes the attention profiles across models and question types. ZwZ exhibits significantly higher bounding-box concentration (A_{bbox}) overall. This localized focus explains its strong performance on positive questions, which inherently demand fine-grained visual grounding.

However, this localized bias becomes problematic for negative questions. When presented with a premise involving an absent target object, ZwZ stubbornly concentrates its attention within the designated box area ($A_{\text{bbox}} = 0.187$ on negatives vs. 0.152 for the base model). Moreover, ZwZ fails to modulate its overall visual engagement when facing negative queries, maintaining nearly identical whole-image attention ($A_{\text{img}} = 0.195$ on negatives vs. 0.201 on positives).

In contrast, the baseline Qwen3-VL-8B dynamically expands its visual search when a premise is false. It allocates a larger share of non-sink attention to the broader image ($A_{\text{img}} = 0.238$ on negatives vs. 0.218 on positives) and distributes its attention more broadly across non-target regions. Because verifying target absence requires exhaustive visual scanning over the whole canvas rather than zooming into a single region, the base model's flexible search behavior leads to far superior verification accuracy.

Discussion These observations reveal a critical trade-off in visual reasoning design: while training models to crop or concentrate heavily on localized visual details effectively boosts positive search performance, it risks inducing a rigid *"tunnel vision"* effect. When presented with hallucinated or absent targets, overly localized models struggle to disengage from candidate regions and fail to execute global visual verification. Developing training objectives that dynamically balance localized feature extraction with global image scanning remains a vital direction for robust visual search architectures.

D.8 Comparison with Dedicated Hallucination Reduction Method

We compare VERVE with FINER-Tuning (Xiao et al., 2026), a state-of-the-art method specifically designed for hallucination reduction, across six hallucination benchmarks. As shown in Table 10, while FINER-Tuning yields consistent gains over the baseline models, VERVE achieves further performance improvements, outperforming FINER-Tuning by up to 7% on the DASH benchmark using the Qwen3-VL-8B backbone. These results indicate that beyond mitigating the perception-hallucination trade-off, VERVE-TRAIN provides robust hallucination suppression that compares favorably with dedicated alignment techniques.

D.9 More Results on Verve-MCQ

We provide more results on VERVE-MCQ, including an analysis of whether prompt engineering alone can close the gap (App. D.9.1), the evaluation of more frontier and open models (App. D.9.2), a breakdown into positive and negative accuracy that motivates our choice of $\text{Acc}_{\text{paired}}$ (App. D.9.3), and paired accuracy split by image source (App. D.9.4).

Table 11: Additional results on VERVE-MCQ (Acc_{paired}).

		TARGET		1-HOP			2-HOP		
Model	**Avg.**	obj	attr	obj	attr	rel	obj	attr	rel
Frontier Models									
Gemini-3.5-Flash	30.1	42.9	36.6	25.7	23.0	17.8	21.4	13.5	15.2
Gemini-3-Flash	23.2	37.3	25.7	20.1	16.4	10.9	16.7	9.5	10.6
Open Models									
Qwen3-VL-235B	20.4	31.2	21.9	21.2	15.9	9.9	15.5	5.4	6.1
Qwen3-VL-32B	13.6	23.8	14.0	14.9	9.3	4.0	8.3	1.4	3.0
Qwen3.5-27B	23.7	36.2	29.3	21.6	16.8	10.9	14.3	9.5	6.1
Step3-VL-10B	18.2	26.3	23.9	15.6	11.5	10.9	7.1	9.5	9.1

Table 12: Averaged positive and negative accuracy on VERVE-MCQ. **Best** and second-best per column within each backbone.

Model	**Paired**	**Pos.**	**Neg.**
Proprietary (API)			
Gemini-3.7-Flash	41.5	64.2	69.5
Qwen2.5-VL-7B	11.4	46.1	36.6
+TreeVGR	6.0	53.8	11.4
+Ours (plain)	33.5	50.7	78.0
+Ours (OPSV)	**40.8**	**54.8**	**80.6**
Qwen3-VL-8B	11.2	44.7	31.0
+ZwZ	7.6	**60.9**	12.2
+Ours (plain)	41.8	54.5	80.3
+Ours (OPSV)	**42.2**	53.9	**81.1**
Qwen3.5-9B	20.1	56.0	40.5
+Vision-OPD	15.6	**64.2**	27.3
+Ours (plain)	49.7	59.8	88.6
+Ours (OPSV)	**55.3**	63.9	**90.5**

Table 13: Paired accuracy on VERVE-MCQ, overall and split by image source: ZoomBench (Zoom.) and TreeBench (Tree.).

Model	**All**	**Zoom.**	**Tree.**
Proprietary (API)			
Gemini-3.7-Flash	41.5	43.9	30.3
Qwen2.5-VL-7B	11.4	11.2	11.9
+TreeVGR	6.0	6.4	4.2
+Ours (plain)	33.5	35.5	24.5
+Ours (OPSV)	**40.8**	**42.7**	**32.3**
Qwen3-VL-8B	11.2	11.9	8.1
+ZwZ	7.6	8.2	4.5
+Ours (plain)	41.8	**44.4**	29.7
+Ours (OPSV)	**42.2**	42.9	**39.0**
Qwen3.5-9B	20.1	22.1	11.0
+Vision-OPD	15.6	17.3	7.7
+Ours (plain)	49.7	52.8	35.5
+Ours (OPSV)	**55.3**	**59.4**	**36.1**

D.9.1 PROMPT ENGINEERING OFFERS LIMITED GAINS

Can models perform well on VERVE-MCQ using better prompts? To test this, we evaluate a "Verify" prompt (instructing caution regarding query premises) and a "Self-Decompose" prompt (decomposing the query into sub-questions, similar to our OPSV training but test-time only). We provide the prompts in App. I. As shown in Fig. 14, left, prompt steering generally improves paired accuracy. However, a closer inspection reveals that most of these gains originate from improved performance on negative queries (Fig. 14, right), which actively results in a decrease in accuracy on positive queries (Fig. 14, middle). For example, while prompting improves Qwen3.5-9B's overall paired accuracy by 5.2%, this is primarily driven by the model becoming overly skeptical and correctly rejecting negative queries, at the direct expense of answering valid positive queries correctly. This fundamental trade-off highlights the need for our VERVE-TRAIN, which robustly improves both positive and negative performance simultaneously, confirming that prompting alone cannot resolve the core failure mode.

D.9.2 EVALUATION OF MORE FRONTIER MODELS

Apart from the frontier models in Tab. 1, we also evaluated a series of other frontier models. They either include some older generations of proprietary models like Gemini-3-Flash (Google DeepMind, 2025), Gemini-3.5-Flash (Google DeepMind, 2026a); Also we evaluate other open-sourced models like Step3-VL-10B (Huang et al., 2026a), Qwen3.5-27B (Team, 2026), Qwen3-VL-235B and Qwen3-VL-32B (Bai et al., 2025a). The results are shown in Tab. 11.

Table 14: Effect of the annealing phase on the Qwen3.5-based VERVE *OPSV* training mode on full 57k training set. **Best** and second-best.

Variant	*Both*		*Percep.*		*Halluc.*	
	VERVE MCQ	VERVE Binary	V*	HR 8K	Re POPE	DASH
Qwen3.5-9B	20.1	73.2	83.2	80.5	94.0	70.5
+Ours *w/o anneal*	51.3	85.8	89.5	**85.2**	**95.7**	79.0
+Ours *anneal*	**55.3**	**87.4**	**90.1**	84.5	95.5	**82.2**

D.9.3 Positive and Negative Accuracy

We select $\text{Acc}_{\text{paired}}$ as our primary metric for VERVE-MCQ, following the definition of verified visual search: the model must search for the target when the query is valid, while correctly indicating its absence when the query does not match the image.

For a detailed breakdown, we also report average positive and negative accuracy on VERVE-MCQ in Tab. 12. As shown in the table, previous state-of-the-art methods like Vision-OPD perform well on positive questions but lag behind their base model on negative questions, dropping by 13.2%. In contrast, our model improves negative question accuracy over its base model by 50%. This gain does not stem from a negation bias: on positive questions, our approach still improves Qwen3.5-9B from 56.0% up to 63.9%.

These observations reinforce our choice of $\text{Acc}_{\text{paired}}$ as the main metric. While positive and negative accuracies can fluctuate wildly, simple averages fail to reflect true performance stability. Paired accuracy offers a stricter, more reliable benchmark for this setting.

D.9.4 Paired Accuracy Across Image Sources

As described in App. A.1, VERVE-MCQ comprises images sourced from ZoomBench (Wei et al., 2026a) and TreeBench (Wang et al., 2026a). We report average $\text{Acc}_{\text{paired}}$ across both subsets in Tab. 13. On both sources, VERVE-TRAIN yields substantial improvements. For instance, applying our OPSV training mode improves Qwen3-VL-8B by 31.0% on ZoomBench samples and 30.9% on TreeBench samples.

D.10 Additional Ablations

Annealing the coverage reward for Qwen3.5-9B When training Qwen3.5-9B in *OPSV* mode, we noticed that due to the strong competence of the base model, it picks up the self-verification format very quickly: its coverage reward R_k reaches about 0.86 within the first 40 steps and stays flat afterwards. Once R_k is nearly the same for all K samples of a prompt, it no longer separates good from bad responses in the GRPO advantage, so the remaining signal comes from R_c alone, but the $(1-\alpha)$ share of the reward is still spent on restating checks. We therefore treat the verification terms as a scaffold and retire them once they are learned. Let s be the training step and $[s_0, s_1]$ an annealing window. We define a ramp

$$\tau(s) = \min\big(1, \max(0, \tfrac{s-s_0}{s_1-s_0})\big), \tag{9}$$

and replace the constants in Eq. (5) by step-dependent values

$$\alpha(s) = \alpha + (1-\alpha)\,\tau(s), \qquad w_f(s) = w_f\,\big(1-\tau(s)\big). \tag{10}$$

Before s_0 the reward is exactly Eq. (5). Between s_0 and s_1 the coverage bonus and the format reward fade out linearly. After s_1 we have $\alpha(s)=1$ and $w_f(s)=0$, so $R = R_c$ and the model is trained on correctness only, while still prompted and answering in the verify format. Nothing else changes: the matcher, the gating on R_c and the format check are the same throughout. For Qwen3.5-9B we use $s_0 = 40$ and $s_1 = 80$ out of 150 steps, leaving about 70 correctness-only steps. The Qwen2.5-VL and Qwen3-VL verify models are trained without annealing ($s_0 = s_1$, which reduces to Eq. (5)), because they never exhibit early saturation on coverage reward during training. Tab. 14 compares the two Qwen3.5-9B training (with and without annealing). Annealing helps most on the paired metrics of VERVE-MCQ and VERVE-BINARY, as well as on DASH, and is at the same level as the non-annealing baseline on the remaining columns.

Table 15: Human performance on VERVE-MCQ (50 twin pairs, 4 evaluators). *Average*: mean over all four cross-evaluator pairings. *Upper bound*: union of two evaluators per test half.

	Paired Acc.	Pos. Acc.	Neg. Acc.
Average	70.5	81.0	86.0
Upper bound	94.0	96.0	98.0

E HUMAN STUDY

Given the difficulty of the benchmark (high-res images, small target, negative questions), one human is not able to answer all our benchmark questions. To establish a human reference point, we conduct a controlled study on VERVE-MCQ. We randomly sample 50 pairs of MCQs (100 questions total), stratified proportionally to the benchmark's category distribution. Each pair consists of a positive (answerable) question and its false-premise negative counterpart.

Since a participant who sees both halves of a pair could use their memory of one to infer the answer to the other, we split the 50 pairs across two non-overlapping tests. For each pair, one half is randomly assigned to Test A and the other to Test B, yielding two balanced sets of 50 questions (25 positive, 25 negative each). Four evaluators participate: two independently take Test A and two take Test B. Participants see the full-resolution photograph and five answer options-the same information available to models. No bounding boxes, object categories, or indications of whether a question contains a false premise are shown.

Because no single participant sees both halves of any pair, we compute paired accuracy by combining each Test A evaluator with each Test B evaluator, yielding four cross-evaluator combinations. We report their mean as the primary human paired accuracy metric for model comparison. As an upper bound, we also compute paired accuracy on the union of the two evaluators per test, counting a question as correct if *either* evaluator answered it correctly. Unlike models, human performance varies across individuals due to oversight or task difficulty; this upper bound reveals the proportion of questions that are human-solvable. Tab. 15 summarizes the results, and Fig. 15 shows the User Interface (UI) for our human evaluation. As shown in Tab. 15, humans achieve an average paired accuracy of 70.5%. In practice, we notice that human participants usually make mistakes due to oversight. We therefore anticipate that given ample time and effort, human performance could be further improved.

F PROMPT FOR TRAINING

Default Prompt - plain mode

A conversation between User and Assistant. The user asks a question, and the Assistant solves it. The assistant first thinks about the reasoning process in the mind and then provides the user with the answer. The answer is enclosed within <answer> </answer> tags, respectively, i.e., reasoning process here <answer> answer here </answer>

Self-verification Prompt - OPSV mode

First verify every claim the question body relies on against the image. Write each check and its verdict as <check> a short yes/no question here </check><verdict> yes or no </verdict> followed by one sentence of visual evidence. Then think about the reasoning process step by step in your mind, and give your final answer from the given choices, i.e. <answer> answer here </answer>

G PROMPT FOR INFERENCE

For baseline methods and our VERVE models trained in plain mode, we adopt the standard default system prompt following ZwZ (Wei et al., 2026a). For VERVE models trained in OPSV mode, the architecture supports dual-mode inference using either the default system prompt or the dedicated

self-verification prompt. We evaluate the self-verification prompt on our VERVE-MCQ and VERVE-BINARY benchmarks.

Default Prompt

You are a helpful assistant. The assistant first thinks about the reasoning process in the mind and then provides the user with the answer. The answer are enclosed within <answer> </answer> tags, respectively, i.e., reasoning process here <answer> answer here </answer>

Self-verification Prompt

First verify every claim the question body relies on against the image. Write each check and its verdict as <check> a short yes/no question here </check><verdict> yes or no </verdict> followed by one sentence of visual evidence. Then think about the reasoning process step by step in your mind, and give your final answer, i.e. <answer> answer here </answer>

H PROMPT FOR BENCHMARK CONSTRUCTION

Prompt for Extracting Referring Chains

You are a careful parser for visual grounding WHAT/WHICH questions.
Return two fields:
1. `num_ref_obj`: number of reference objects used to locate/describe the needle object.
2. `pos_dict`: `queried_aspect`, `pos_needle_obj`, `pos_needle_attr`, and references.
Task definition:
- The question asks about a target/needle object or entity in an image.
- It may ask about color, shape, material, surface finish, text, name, object identity, item type, activity, action, or state.
- First identify the queried aspect/property.
- Then identify the needle object/entity whose property, identity, or action is being asked.
- Then identify explicit spatial/action relations from the needle object to distinct reference objects.
- `num_ref_obj`equals the number of such reference objects.
Extraction rules:
- Copy original words/characters. Do not paraphrase, rewrite, singularize, pluralize, or invent.
- Object fields should keep compound nouns together, e.g. 'traffic light', 'license plate', 'belt buckle'.
- Ordinary modifiers go into attr.
- Possessive/part-whole phrases using "of" or "'s" stay inside the object phrase when they define the object, e.g. "person's jacket", "window of the building", "door of the boat".
- Do NOT count generic 'of' or possessive "'s" as relations.
- Do NOT split ownership/part-whole phrases into reference objects.
- Count only explicit spatial/action relations connecting distinct objects, e.g. on, in, under, behind, in front of, attached to, holding, wearing, worn by, placed on, located in, positioned in, next to, beside, above, beneath.
- Ignore generic references to the image itself, e.g. 'in the picture', 'in the image', 'on the left side of the picture', unless no other grounding information exists.
Critical distinction:
- If a relation only connects the queried aspect/property to the needle object, ignore it.
Example: in 'What text is written on the man's blue cap?', 'on' only connects text to cap, so `num_ref_obj` $= 0$.
- If a relation is part of locating/describing the needle object, count it.
Example: in 'What color is the cup on the table?', 'on the table' locates the cup, so `num_ref_obj` $= 1$.
Examples:
Question: What text is written on the man's blue cap?
Output:
`num_ref_obj` $= 0$
`pos_dict`= { `queried_aspect`: 'text', `pos_needle_obj`: 'cap', `pos_needle_attr`: "man's

blue", references: [] }
Return only the structured fields.

Prompt for Proposing Negative Edits

You are helping construct false-premise visual questions.
You will see an image, the original positive WHAT/WHICH question, and its parsed positive dictionary.
Your job is to rewrite the dictionary so that the new dictionary describes something false or clearly unsupported by the image content.

Input fields:
- `question_body`: the original positive question.
- `pos_dict`: parsed positive dictionary.
- `source_options` and `correct_answer` are provided only for visual context.

Output field:
- `neg_dict`: same structure as `pos_dict`, but with false-premise content.

Core rules:
- Keep `queried_aspect` unchanged in `neg_dict`.
- Modify `pos_needle_obj`, `pos_needle_attr`, and all fields inside `references`.
- Construct strong, unambiguous, visually challenging negatives. Do not copy or preserve the positive target content.
- If a field in `pos_dict` is `'-1'`, you are allowed to create an active negative object, attribute, or relation for `neg_dict`.
- The proposed negatives must NOT be visible or supported anywhere in the image content.
- If a field is hard to negate safely without accidental matches, output `'-1'` for that specific field.
- Prefer clear, concrete visual flips (e.g., blue → red, roof → ground, on → under, boat → car).
- Avoid weak, trivial, or ambiguous variations.
- Keep phrasing short, concise, and natural.

Structural rules:
- Preserve the exact reference order from `pos_dict`.
- In `neg_dict`, `references` must be a list in the identical order as the positive references.
- If `pos_dict` has no references, `neg_dict` must also have `references: []`.
- Field name compatibility note: `pos_needle_obj` and `pos_needle_attr` are retained as keys in `neg_dict`, but must contain the NEGATIVE object and NEGATIVE attribute values respectively.

Examples:
Question: What is the primary material used for the white bench next to the tree?
Input pos_dict:
{ queried_aspect: 'material', pos_needle_obj: 'bench', pos_needle_attr: 'white', references: [{rel: 'next to', obj: 'tree', attr: '-1'}] }
Output neg_dict:
{ queried_aspect: 'material', pos_needle_obj: 'metal table', pos_needle_attr: 'black', references: [{rel: 'under', obj: 'umbrella', attr: 'red'}] }

Now process this sample.
question_body: {question_body}
source_options: {source_options}
correct_answer: {correct_answer}
positive_reference_order: {pos_ref_keys}
pos_dict: {pos_dict}
Return only the structured neg_dict.

I PROMPT FOR TEST-TIME STEERING

Prompt steering - verify

You are a careful visual question answering assistant.
Given one image and one question asking about a target in the image. You must carefully follow the question, search for the target, and inspect all visual conditions mentioned in the question.
When the question mentions a target object, attribute, relation, location, number, text, or other visual condition, explicitly verify that visual condition against the image before concluding.
The assistant first thinks about the reasoning process in the mind and then provides the user with the answer. The answer is enclosed within <answer> </answer> tags, i.e., reasoning process here <answer> answer here </answer>.

Prompts steering - self-decompose

You will be given an image and a multiple-choice question about that image. Before choosing an answer, you must perform the following steps:
Step 1. Decompose the question stem into a list of simple yes/no verification sub-questions. Write them under the heading "Sub-questions:".
Step 2. Inspect the image and answer each sub-question in order. Write them under the heading "Sub-answers:". For each item, write "yes", "no", or "unclear", followed by a short justification grounded in what you actually see in the image.
Step 3. Use your sub-answers to reason about which multiple-choice option best fits the image and the question.
Step 4. Output the chosen option letter inside <answer> </answer> tags, e.g. <answer> A </answer>.
Rules for the Sub-questions list:

- Cover every object the question stem mentions with an existence check, e.g. "Is there a X?".
- Cover every descriptive attribute the question stem attaches to an object, with an attribute check, e.g. "Is the X red?". Descriptive attributes include color, size, material, shape, count, specific text content, visual pattern, age or state, etc.
- Cover every spatial or relational phrase the question stem uses, with a relation check, e.g. "Is the X behind the Y?". Relations include positional ones, interactional ones, etc.
- Decompose only the question stem itself. Ignore the multiple-choice options when writing sub-questions.

The following are examples that illustrate ONLY how to break a question stem into sub-questions. They are NOT real questions about the image you are given - they intentionally omit sub-answers, reasoning, and final answers, because those must come from inspecting your actual image.
Example A
Question stem: "What is the color of the small dog next to the bicycle?"
Sub-questions:

1. Is there a dog?
2. Is the dog small?
3. Is there a bicycle?
4. Is the dog next to the bicycle?

Example B
Question stem: "What symbol is visible on the green tram's front, to the left of the number 8275?"
Sub-questions:

1. Is there a tram?
2. Is the tram green?
3. Is there a number 8275 on the tram's front?
4. Is there a symbol to the left of the number 8275?

Example C
Question stem: “How many red apples are on the wooden table?”
Sub-questions:

1. Is there a wooden table?
2. Are there apples on the table?
3. Are the apples red?

Now look closely at the image and answer the following question by following Steps 1-4 above.
Question: {}





What symbol is shown inside the yellow square on the bench?
A. a checkmark
B. a person walking
C. an airplane landing
D. a suitcase
E. there is no such symbol

Qwen3.5-9B:
Looking at the image, there is a yellow square on the black sign that says "Domestic Arrivals" in English and Thai. Inside that yellow square is a symbol of an airplane with wheels down, indicating landing.

<answer> C </answer>

The model didn't mention the "bench" at all.

VisionOPD:
The image shows a sign with the text "Domestic Arrivals" and Thai/Chinese translations. To the left of the text, there is a yellow square containing a black silhouette of an airplane on a runway, which clearly depicts an airplane landing.
This matches option C: **an airplane landing**.

<answer>C</answer>

The model didn't mention the "bench" at all.

DeepEyesV2:
<think>The image shows a sign with a yellow square containing an icon. Let's examine the options:
- A. a checkmark: Not present.
- B. a person walking: Not present.
- C. an airplane landing: The icon looks like an airplane, which would represent an airplane landing or flight.
- D. a suitcase: Not present.
- E. there is no such symbol: Clearly, there is a symbol.</think>
<code>```python
import matplotlib.pyplot as plt
# Define the crop box for the yellow square and its icon (approximate coordinates)
# x, y, width, height
crop_box = (1380, 440, 1450, 510)
# Crop the image to focus on the yellow square and its icon
cropped_image = image_1.crop(crop_box)
# Display the cropped image
plt.imshow(cropped_image)```</code><think>The cropped image focuses on the yellow square and its icon. From the original image and the crop, the icon inside the yellow square is clearly an airplane. Therefore, the correct answer is C.</think>
<answer>C. an airplane landing</answer>

The model didn't mention the "bench" at all.

Ours (Plain):
Looking at the image:
- There is no "bench" in the question context — the yellow square appears on a sign that says "Domestic Arrivals" with an airplane symbol inside it.
- The question asks about a symbol inside a yellow square on the bench — but there's no bench visible in the image; instead, there are chairs or seating areas along the wall, and a prominent yellow square is part of the overhead sign for domestic arrivals.
- Furthermore, looking closely at the image: the yellow square is actually located on the black hanging signboard labeled "Domestic Arrivals", not on any bench. Inside the yellow square is an icon of an airplane (landing or taking off).

However, the user asked: What symbol is shown inside the yellow square on the bench? There is no bench visible in this image. Therefore, the premise of the question is incorrect — there's no yellow square on a bench. So option E ("there is no such symbol") fits best because the location mentioned ("on the bench") does not exist in the image.

The only yellow square present is on the overhead sign — containing an airplane symbol. But since the question specifically mentions "on the bench", which doesn't exist → answer should be E.

<answer> E </answer>

Our plain model instantly recognized there's no bench.

Ours (OPSV):
<check> Is there a bench in the image? </check>
<verdict> no </verdict>
There are chairs and seating areas, but no specific bench is visible.

<check> Is there a yellow square in the image? </check>
<verdict> yes </verdict>
There is a yellow square with an airplane symbol on the hanging sign for "Domestic Arrivals".

<check> Is there a symbol inside the yellow square? </check>
<verdict> yes </verdict>
The symbol is an icon of an airplane.

<check> Is the symbol inside the yellow square on the bench? </check>
<verdict> no </verdict>
The yellow square is located on a directional sign above the floor, not on a bench.

<check> Can the symbol be identified as an airplane landing? </check>
<verdict> yes </verdict>
The symbol depicts an airplane, consistent with airport signage.

Since there is no bench in the image, the premise that there is a symbol *on* a bench within a yellow square is false. Therefore, "there is no such symbol" (in the context of being on a bench) is the correct choice based on the visual facts.

<answer> E </answer>

Our verify model recognized there's no bench by asking self-verification questions and answer them correctly.

Figure 12: Thinking traces of our models versus the other open-source models on a VERVE-MCQ negative question. Our models, both plain and OPSV, successfully rejected the false premise, while the other SOTA models fail to detect it.

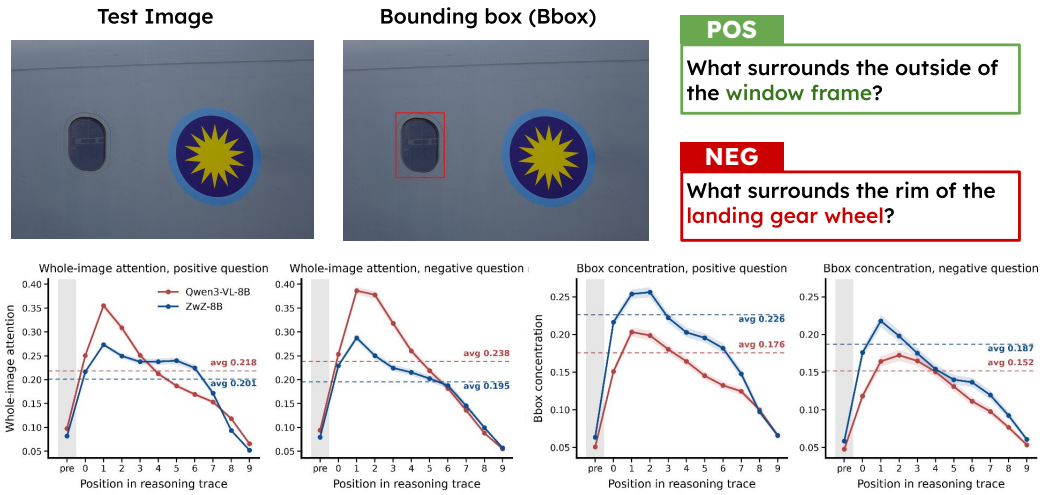

Figure 13: Attention dynamics on positive and negative questions in VERVE-MCQ. **Top:** A representative test image alongside paired positive and negative queries, where bounding boxes delineate ground-truth region B. **Bottom:** Temporal progression of whole-image attention $A_{\text{img}}(W)$ and bounding-box concentration $A_{\text{bbox}}(W)$ from prefill across reasoning deciles for Qwen3-VL-8B (Bai et al., 2025a) and ZwZ (Wei et al., 2026a).

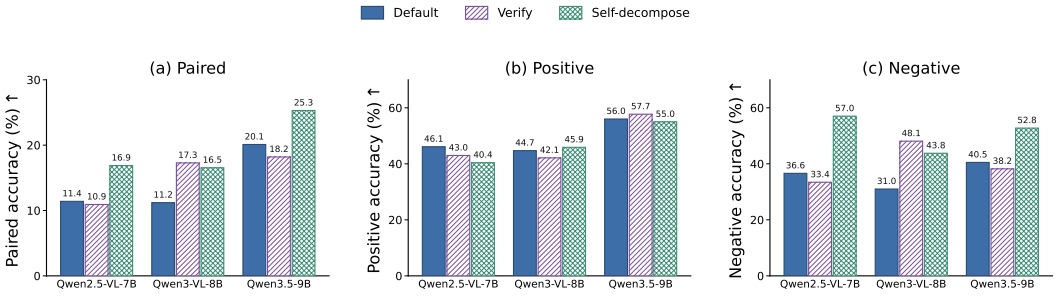

Figure 14: Effect of different prompts on VERVE-MCQ for Qwen2.5-VL-7B, Qwen3-VL-8B and Qwen3.5-9B. Left: paired accuracy. Middle: accuracy on positive queries. Right: accuracy on negative queries.

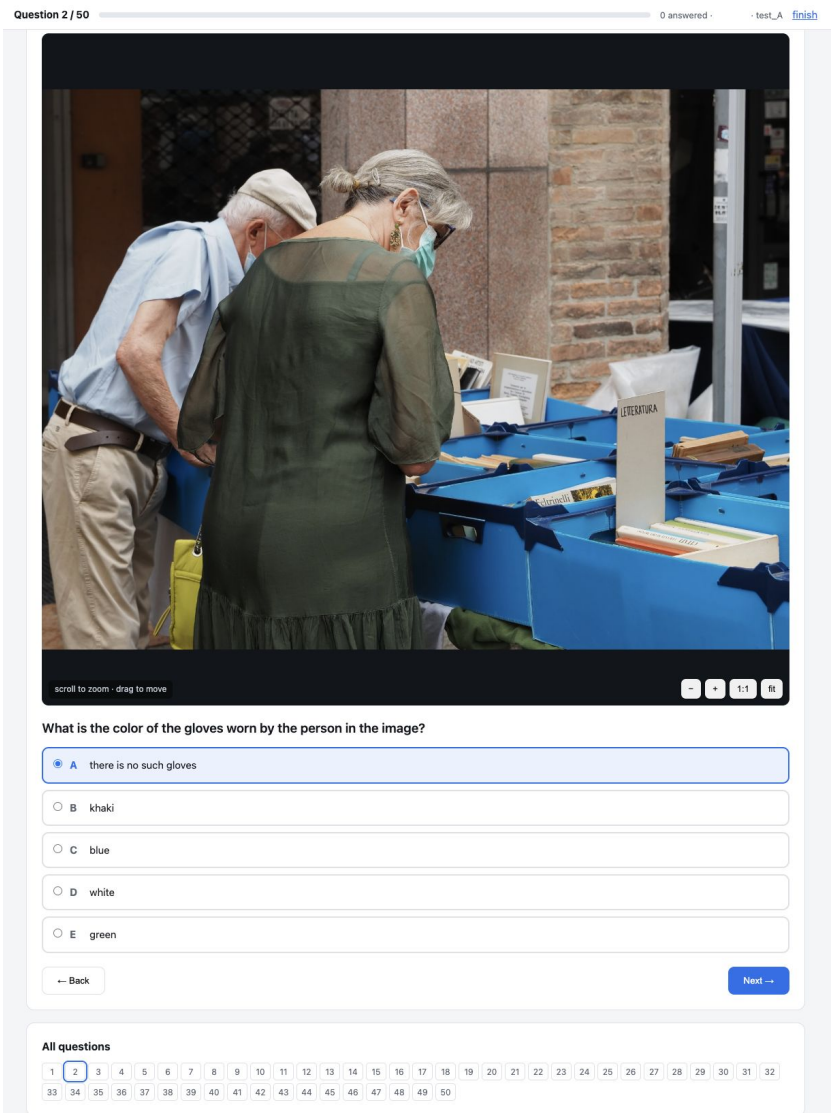

Figure 15: Example for human evaluation UI.