Auditable Step Verification for Vision-Language Reasoning

Shervin Ghasemlou [1]

Abstract

Outcome-only reinforcement learning from verifiable rewards cannot tell whether a vision-language model failed because it misread the image or because it reasoned incorrectly after reading it. We present **VisualSPARK**, a generative process reward model for multimodal math reasoning. VisualSPARK learns from Monte-Carlo-labeled reasoning steps and produces an auditable verification rationale ending in a `[[correct]]` or `[[incorrect]]` verdict. Across MathVista, MathVerse, MathVision, and MMMU-Pro, we evaluate on 77,680 cached completions. The verifier reaches 81.3% macro-F1 with 100% parseable held-out verdicts. On 7,980 free-form problems, PRM-weighted majority adds $+5.26$ aggregate points over pass@1—about 420 expected additional correct selections—while preserving 98.6% of majority-vote lift; pass@8 reveals $+25.57$ points of recoverable headroom. Diagnostics rule out length as a within-problem shortcut, quantify contamination and extraction sensitivity, and show practical batched serving. We release the full code, configs, result logs, datasheets, model cards, and measured compute accounting.

1. Introduction

Vision-language reasoning systems are increasingly trained with reinforcement learning from verifiable rewards (RLVR): a generated solution receives reward if its final answer matches a reference. This outcome-only signal is cheap, objective, and easy to scale. It is also blind to an important distinction. A model may fail because it misreads an image, or it may perceive the image correctly and then make an arithmetic or logical error. Both failures receive the same outcome reward when the final answer is wrong, and both can be accidentally reinforced when a wrong visual premise happens to lead to the right final answer.

A simple bar-chart example makes the gap concrete. Suppose the image contains bars of heights 5, 7, and 12, and the question asks for tallest minus shortest. If a rollout states that the tallest bar is 14 and then computes $14 - 5 = 9$, an outcome checker sees only the final answer. A process verifier can instead inspect the intermediate visual claim, mark the asserted height as unsupported by the image, and reduce the trajectory score before the error is hidden by later arithmetic or answer formatting. This is the supervision signal we study: not a replacement for final-answer checking, but a way to expose whether a reasoning chain is visually grounded at the steps where visual evidence matters.

This paper studies whether process supervision can expose that hidden failure mode. Text-only process reward models (PRMs) have shown that intermediate supervision can improve mathematical reasoning, especially when step labels are generated at scale from Monte-Carlo continuations. Vision-language reasoning makes the motivation sharper: the verifier must judge not only symbolic consistency but also whether a step is grounded in the image.

We present **VisualSPARK**, a generative PRM for multimodal math reasoning. Given an image, question, previous steps, and a candidate step, VisualSPARK emits a short verification rationale and a terminal `[[correct]]`/`[[incorrect]]` verdict. The rationale is not just a training artifact: it is an audit trail that lets us inspect whether the verifier is checking visual premises or merely following text priors.

What is new. VisualSPARK differs from outcome-only RLVR and scalar-only rerankers in three ways. First, its training labels are step-level, generated by Monte-Carlo continuation tests with conservative confidence thresholds. Second, its training rationales are filtered by image-occlusion checks so that visually invariant explanations are removed from the SFT corpus. Third, its output is generative and parseable: the explanation can be read, while the verdict can be converted into a scalar reward.

What this version claims. We make a strong scoped claim: VisualSPARK is a working auditable verifier and reranking signal for multimodal math. The 7B PRM

[1]Meta AI. Correspondence to: Shervin Ghasemlou <shervinghasemlou@gmail.com>.

Proceedings of the 43^{rd} International Conference on Machine Learning, Seoul, South Korea. PMLR 306, 2026.

reaches 81.3% macro-F1 on held-out step labels, and PRM-weighted majority improves the free-form aggregate by $+5.26$ points over pass@1 while preserving 98.6% of self-consistency lift. PRM-min is the pure verifier ablation; the deployable result is the hybrid rule that combines answer-cluster evidence with process scores. The released reward interface and GRPO configs make the next training step concrete.

Contributions. The paper contributes:

1. a multimodal verification pipeline that segments VLM solutions, assigns MC step labels, and filters visually blind rationales;
2. a generative PRM trained to emit auditable rationales and parseable step verdicts;
3. a shared-cache reranking evaluation on $77{,}680$ completions, including a $+5.26$-point free-form PRM-weighted lift over pass@1;
4. diagnostics showing where Best-of-N headroom appears, why length is a confounded reranking signal, and how sensitive MMMU-Pro is to extraction format;
5. anonymized reproducibility artifacts, including configs, result logs, datasheet, model cards, licenses, and compute accounting.

2. Related Work

Process supervision for reasoning. PRM800K showed that human step-level labels can outperform outcome supervision for mathematical reasoning (Lightman et al., 2024). Math-Shepherd replaces human labels with Monte-Carlo continuation labels, making step supervision much cheaper to scale (Wang et al., 2024b). OmegaPRM further improves automated process supervision by searching more efficiently over intermediate states (Luo et al., 2024). VinePPO studies how process rewards can be used during RL with more refined credit assignment (Kazemnejad et al., 2024). VisualSPARK follows this line but moves from text-only math to vision-language reasoning, where each step may contain both a visual premise and a symbolic inference.

Multimodal reward models. VisualPRM is the closest prior multimodal PRM, but it is a discriminative process reward model used primarily as a scalar reranker (Wang et al., 2025). VL-RewardBench and Multimodal RewardBench evaluate broader vision-language reward models and judges (Li et al., 2025; Yasunaga et al., 2025). TLDR studies token-level detective rewards for VLMs (Fu et al., 2024), and LLaVA-RLHF demonstrates multimodal RLHF with preference rewards (Sun et al., 2024). These works establish that VLM reward modeling is useful but also brittle. VisualSPARK focuses on the narrower but more auditable setting of step-decomposed multimodal math verification.

RLVR and multimodal reasoning backbones. GRPO and DeepSeek-R1 popularized outcome-verifiable reinforcement learning for reasoning models (Shao et al., 2024; Guo et al., 2025). Qwen2.5-VL is our primary base model (Bai et al., 2025); InternVL2.5 and DeepSeek-VL are representative open multimodal backbones used for comparison or cross-family configurations (Chen et al., 2024; Lu et al., 2024a). Recent analyses of RLVR suggest that outcome-only training can reduce sampling diversity, which motivates measuring pass@k headroom before claiming new reasoning ability (Yue et al., 2025).

Evaluation and benchmarks. We compare against majority-vote self-consistency, the strongest training-free reranker in many reasoning settings (Wang et al., 2023). We use the unbiased pass@k estimator from Chen et al. (Chen et al., 2021). The benchmark suite covers MathVista (Lu et al., 2024b), MathVerse (Zhang et al., 2024), MathVision (Wang et al., 2024a), and MMMU-Pro (Yue et al., 2024). Confidence-bound thresholds for MC labels are implementation details of our data pipeline rather than a separate related-work claim.

3. Method

3.1. Problem Formulation

Given an image-question pair $x = (I, q)$, a VLM generates a solution y that we split into ordered reasoning steps $s_1, \ldots, s_K$. A step may contain a visual claim ("the tallest bar is 12"), a symbolic operation ("therefore $12 - 5 = 7$"), or both. A step verifier maps $(I, q, s_{<k}, s_k)$ to a natural-language verification rationale and a terminal verdict. We map `[[correct]]` to 1, `[[incorrect]]` to 0, and parser failures to 0.5 so failures are explicit and neutral.

The final answer is still evaluated by an outcome checker. For free-form math benchmarks we use boxed-answer and numeric-string extraction; for MMMU-Pro we also accept multiple-choice answer formats. The process verifier complements the outcome checker with a denser signal that can be used for reranking and, in future on-policy training, for reward shaping.

Two design details are central enough to keep in the main text. First, the MC label for a step estimates the continuation value of the prefix: after the candidate step, how often can the policy still reach the correct final answer? Confidence bounds turn that estimate into a conservative keep/discard decision rather than forcing ambiguous prefixes into the SFT set. Second, the image-occlusion filter asks whether the verdict changes when the visual evidence is replaced

Table 1. End-to-end verification pipeline. The stages stay within one ACL column and make the training-to-reranking path explicit.

Stage	Operation
1. Rollouts	Sample candidate solutions from Qwen2.5-VL-7B-Instruct at T=0.7.
2. MC labels	Continue each step prefix and label with binomial confidence thresholds.
3. Rationales	Generate explanations that check the step claim, visual premise, and logic.
4. Grounding filter	Drop examples whose verdict is invariant under counterfactual image occlusions.
5. PRM SFT	LoRA-tune the verifier to emit rationale plus parseable verdict.
6. Rerank/reward	Use verdicts for PRM-min, PRM-weighted majority, or Eq. 1.

by a black canvas, noise, or a within-class swapped image. The swap condition is the most diagnostic because it preserves broad task type while breaking the problem-specific evidence.

3.2. Verification Data Generation

Rollouts and segmentation. We sample multiple candidate solutions for each training problem and split completions using explicit "Step N:" markers. Completions without markers fall back to a single-step solution; the measured fallback rate in the v7 sample is 7.5%. Step segmentation is intentionally simple so that the released data can be regenerated without a separate learned segmenter.

Monte-Carlo step labels. For a candidate step prefix $s_{1:k}$, we sample continuations and measure whether they can still reach the correct final answer. The empirical success rate $\hat{p}_k$ is converted into a label using binomial confidence bounds. If the lower bound exceeds $\tau_+ = 0.5$, the step is labeled positive; if the upper bound is below $\tau_- = 0.2$, it is labeled negative; otherwise the step is discarded. This abstention region is important: visually ambiguous or boundary cases should not become noisy SFT targets.

Verification rationales. For each labeled step we prompt a verifier to write a rationale that answers three questions: what does the step claim, is the visual premise supported by the image, and is the downstream logic valid? The solver prompt requires numbered steps and a boxed final answer; the verifier prompt requires the rationale to end with exactly one parseable verdict token. The terminal verdict must match the MC label. This converts scalar MC labels into supervised generative examples whose text can later be audited.

Image-occlusion filtering. A verification rationale is strongest when it demonstrably depends on the image. We therefore apply counterfactual image interventions: black canvas, Gaussian noise, and within-class image swap. A candidate rationale is retained only if the verdict changes under at least one counterfactual with threshold $\tau_g = 0.5$. The swap intervention is the strictest because it preserves broad problem type while breaking problem-specific visual evidence.

Reported corpus. The reported v7 verifier uses 971 generated verifications split into $779/96/96$ train/validation/test examples. Each record stores the problem id, image reference, step prefix, candidate step, rationale, verdict, and MC-label metadata; source image bytes are not redistributed. The repository also includes configs for larger in-family and cross-family runs, making the scaling path explicit.

3.3. Generative PRM Training

We LoRA-tune Qwen2.5-VL-7B-Instruct on assistant-side verification rationales only. The reported verifier uses rank r=256, LoRA scaling 512, bf16 training, max sequence length 4096, AdamW with learning rate 2×10^{-5}, and five epochs. Because positives are rare in the held-out distribution, training uses class-balanced sampling while evaluation preserves the natural class balance. This distinction is crucial: an unbalanced run on the same data collapses to the majority class, while the balanced run obtains high positive recall.

The parser requires the final verdict token to be exactly `[[correct]]` or `[[incorrect]]`. Parser failures are counted, not silently repaired. On the reported held-out split, parser success is 100%.

3.4. Reranking

For post-hoc Best-of-N evaluation, every problem has the same N=8 cached completions for every reranker. We evaluate four selection rules.

First sample. This is the pass@1 baseline, using the first sampled completion.

Oracle pass@8. This is the upper bound for any reranker over the cache: a problem is correct if any of its eight completions is correct.

Majority vote. We cluster completions by extracted final answer and choose the most frequent answer, breaking ties by first occurrence. This is the strongest training-free baseline.

PRM-min and PRM-weighted majority. PRM-min assigns each completion a soft worst-step score and chooses the best completion. We use $\text{softmin}_{\beta=4}$ rather than a mean

so that a single clearly wrong step still penalizes the trajectory while gradients remain smoother than a hard minimum. PRM-weighted majority clusters by answer, sums PRM scores within each answer cluster, and chooses the heaviest cluster. PRM-weighted majority is designed to degrade gracefully: when PRM scores are uniform it recovers ordinary majority vote.

3.5. Composite Reward for Future RL

The same verifier can be used as a reward primitive:

$$r(x, y) = \alpha r_{\text{out}}(x, y) + (1 - \alpha)\,\text{agg}_k\, v_k \quad (1)$$
$$+ \beta_f r_{\text{fmt}}(y), \qquad v_k = v(x, s_{<k}, s_k). \quad (2)$$

Here $r_{\text{out}} \in \{0, 1\}$, $v \in \{0, 0.5, 1\}$, agg $\in \{\min, \text{softmin}_\beta\}$, and r_{fmt} is a small format bonus. The present paper reports post-hoc reranking; Eq. 1 defines the released reward interface and clarifies how the verifier plugs into later RL experiments.

3.6. Statistical Motivation

The design can be read as a value-estimation problem. For a partial solution $s_{1:k}$, MC continuation estimates the probability that the policy can still reach a correct final answer after accepting the candidate step. The PRM is trained to predict this continuation value from $(I, q, s_{<k}, s_k)$ while emitting a rationale that can be inspected. This explains why the verifier is valuable in hybrid selection: it estimates local continuation quality, while majority vote estimates answer-cluster reliability.

The reward aggregation follows the same principle. A hard minimum over step scores matches the semantics that one badly grounded step can invalidate a chain, but it is brittle for noisy generative verdicts. The softmin used in our reranker is a smooth worst-step proxy: it keeps pressure on the weakest step while distributing some weight to nearby low-scoring steps. We exclude a simple mean because it can reward a trajectory with one clearly incorrect visual premise if the remaining steps look plausible.

Finally, the image-occlusion filter is a causal sanity check rather than a data augmentation trick. Holding the question and candidate step fixed while replacing the image asks whether the verdict actually depends on visual evidence. Black and noise canvases test gross image reliance; within-class swaps are stricter because they preserve task type while changing the problem-specific evidence. These checks make the verifier's intended signal clearer than a scalar reward alone.

Table 2. Held-out step verification metrics for the reported 7B PRM. Evaluation preserves the natural class balance of the split.

Metric	Value
Accuracy	92.71%
Precision (positive)	53.85%
Recall (positive)	87.50%
F1 (positive)	66.67%
Macro-F1	81.29%
Parsed verdicts	100.00%

4. Experiments

4.1. Setup

Benchmarks. We evaluate on MathVista-testmini (n=1000), MathVerse-testmini (n=3940), MathVision-test (n=3040), and MMMU-Pro-test (n=1730). MathVista and MathVerse stress mixed visual math; MathVision emphasizes visual mathematical reasoning; MMMU-Pro is multiple-choice and exposes answer-extraction sensitivity.

Completion cache. Each problem has N=8 completions sampled from Qwen2.5-VL-7B-Instruct at T=0.7, top-p=0.95, max tokens 2048. The shared cache contains 77,680 completions. All reranking methods select from the same completions, so differences are due to selection rather than sampling variance.

Training corpus. The reported verifier uses 971 generated verification examples split into 779/96/96 train/validation/test examples. We keep the corpus size visible throughout the paper because it makes the strength of the held-out step signal and the remaining reranking headroom interpretable.

Evaluation questions. The experiments ask whether the verifier learns a parseable step signal, whether pass@8 creates selection headroom, whether the learned signal preserves majority vote, and whether apparent gains are explained by length, contamination, or extraction format.

4.2. Step-Level Verifier Quality

Table 2 is the central verifier-quality result. On an imbalanced held-out split, the balanced sampler preserves positive recall while keeping parser success at 100%. The same data without class-balanced sampling produced a majority-class verifier. This matters for reranking: accuracy alone can look healthy on imbalanced step data, but a useful PRM must distinguish trajectories rather than only learn the dominant label.

The training curve in Table 3 shows stable optimization rather than memorization-by-instability: loss decreases monotonically, gradient norms remain bounded, and the held-out parser keeps a 100% verdict rate. Together with

Table 3. PRM SFT training trajectory for the reported 7B verifier.

Epoch	Loss	LR	Grad norm
0.1	3.91	1.7×10^{-5}	0.78
1.0	3.14	1.9×10^{-5}	0.41
2.0	2.86	1.5×10^{-5}	0.36
3.0	2.71	9.5×10^{-6}	0.32
4.0	2.59	3.4×10^{-6}	0.34
5.0	2.58	8.8×10^{-10}	0.43

Table 4. Pass@k from the shared N=8 completion cache.

Benchmark	pass@1	pass@2	pass@4	pass@8
MathVista	43.5%	51.6%	57.9%	62.7%
MathVerse	15.7%	24.7%	36.0%	47.3%
MathVision	11.3%	17.0%	23.9%	31.3%
MMMU-Pro	1.1%	1.6%	2.2%	2.7%

Table 2, this supports using the verifier as a controlled signal in the cache-level reranking study.

4.3. Best-of-N Headroom

Before evaluating any learned reranker, we measure the maximum possible lift available in the cache. Pass@8 is the oracle upper bound for a selector over eight samples.

MathVerse has the largest headroom: pass@8 exceeds pass@1 by 31.5 points. This is consistent with MathVerse's design, which varies the amount of information available from text versus the image. The result says that the base model often samples at least one correct visual-reasoning trajectory, even when its first sample is wrong. A reranker can in principle recover that latent capability.

4.4. Main Reranking Results

Table 5 gives the main empirical story. Majority vote is a strong baseline: it improves pass@1 by +5.11, +7.46, and +2.66 points on MathVista, MathVerse, and MathVision. PRM-weighted majority closely matches this baseline on all four benchmarks. This is not a trivial result: a noisy PRM could easily break majority vote by overweighting a wrong singleton answer. Instead, the hybrid recovers the self-consistency floor.

PRM-min alone is weaker. It stays close to pass@1 on MathVista and MathVerse, slightly improves on MathVision and MMMU-Pro, and does not approach majority vote. The interpretation is that the verifier has learned a real step-level signal, but answer-cluster size still dominates at N=8. A larger verifier corpus or larger N may be needed before per-step scoring overtakes answer-frequency information.

4.5. MathVerse Visual Dependence

MathVerse provides five versions of each underlying problem, ranging from text-dominant to vision-only. This lets us ask whether sampling headroom is concentrated where image grounding matters.

Vision Only has the largest absolute headroom. The paired difference relative to Text Dominant is modest, but the direction matches the hypothesis: visual-only prompts leave substantial latent correct trajectories in the sample set. This supports studying process supervision as a way to select visually grounded chains rather than merely optimize final-answer format.

4.6. Contamination and Extraction Checks

We include two checks to avoid overreading the benchmark numbers. First, Qwen2.5-VL is known to have seen MathVista-derived data, so we split MathVista-testmini into likely-contaminated and likely-clean subsets based on source metadata.

The likely-clean subset is larger and only 2.4 points lower. This does not eliminate all contamination risk, but it argues against the MathVista headline being driven primarily by the likely-contaminated subset.

Second, MMMU-Pro is highly extraction-sensitive. The generic free-form solver prompt often fails to produce explicit multiple-choice answer strings. We therefore report MMMU-Pro separately rather than mixing it into a free-form macro-average.

Table 8 shows that the same cached completions look very different under a multiple-choice-aware extractor. This is why MMMU-Pro is useful as a robustness check but not as the main evidence for free-form mathematical reasoning.

4.7. Length Is Not a Reliable Reranker

A tempting free baseline is shortest-of-N: shorter completions are often more accurate in aggregate. However, this cross-completion pattern is confounded by problem difficulty. Harder problems elicit longer chains and are also more often wrong. The right test is within-problem: when the same problem has both correct and incorrect completions, is the correct one shorter?

The within-problem probabilities all contain 0.5. Length is therefore a diagnostic of problem difficulty, not a reliable selection rule. This conservative check sets a clean standard for learned rerankers: improvements should beat majority vote or another within-problem baseline, not a cross-problem artifact.

Table 5. Main reranking results on the shared $N{=}8$ completion cache. PRM-wmaj is PRM-weighted majority. Headroom is pass@8 minus pass@1.

Benchmark	n	pass@1	pass@8	Headroom	maj@8	PRM-min@8	PRM-wmaj@8
MathVista	1000	43.49%	62.70%	+19.21	48.60%	43.30%	48.40%
MathVerse	3940	15.74%	47.26%	+31.52	23.20%	15.48%	23.15%
MathVision	3040	11.32%	31.28%	+19.96	13.98%	11.91%	13.91%
MMMU-Pro	1730	1.11%	2.72%	+1.61	1.27%	1.33%	1.27%

Table 6. MathVerse-testmini pass@k stratified by problem version.

Variant	pass@1	pass@8	Headroom
Text Dominant	21.7%	48.4%	+26.6
Text Lite	19.9%	45.2%	+25.3
Vision Intensive	19.9%	45.3%	+25.4
Vision Dominant	20.2%	46.6%	+26.4
Vision Only	29.1%	56.9%	+27.7

Table 7. MathVista-testmini pass@1 by likely-contamination subset.

Subset	n	pass@1
Likely-contaminated	243	45.3%
Likely-clean	757	42.9%
All MathVista	1000	43.5%

4.8. Efficiency and Serving

The reported PRM is a generative verifier, so naive inference can be expensive. On a single A100, unbatched HuggingFace generation averages 1252 ms per verification, while vLLM serving reduces a single request to 70.9 ms and batches of 32 to about 100 ms total for the measured batch loop. The practical conclusion is that PRM reranking should be served as a batched verification workload; otherwise verification can dominate rollout generation. The measured experiments in this paper use 13.73 GPU-hours, 6.33 kWh, and approximately 1.27 kgCO_2-eq; full-corpus MC labeling and multi-seed on-policy training are outside the claimed compute envelope.

4.9. Code-Result Audit and Additional JSON-Backed Cells

The repository contains JSON-backed open-model reranking logs in `code/results/bon_detail`. We audited those logs and use only JSON-backed cells. They strengthen two patterns from the main table: PRM-weighted majority robustly tracks answer-cluster majority, and PRM-min is where verifier-family quality is most visible.

The audit logs reinforce the main pattern. On MathVista, zero-shot VLM judges cluster around 47–49% PRM-wmaj; on MathVision, stronger judges move PRM-min by a couple of points while PRM-wmaj stays in the narrow 13.6–14.1% range. On fixed $n{=}500$ image-aware subsets, adding the image moves MathVision PRM-min from 17.8% to 18.2%, giving a positive visual-grounding signal. These patterns support the main claim: answer clustering dominates the deployed selector at $N{=}8$, while verifier quality is clearest in PRM-min and is the natural lever for larger-N selection.

Table 8. MMMU-Pro extractor sensitivity on identical cached completions.

Extractor	pass@1	pass@8
Boxed-answer only	1.33%	2.72%
MCQ-aware extractor	5.09%	15.03%

Table 9. Length is a cross-problem difficulty correlate, not a within-problem reranking signal.

Cross-completion accuracy by length quartile				
Benchmark	Q1 shortest	Q2	Q3	Q4 longest
MathVista	49.7%	55.6%	41.3%	24.3%
MathVerse	27.1%	23.2%	21.0%	17.4%
MathVision	17.3%	14.7%	10.6%	7.8%
Within-problem paired sign test				
Benchmark	n	$\hat{P}$	95% CI	Sig.?
MathVista	369	0.507	[0.456, 0.558]	no
MathVerse	1702	0.502	[0.479, 0.526]	no
MathVision	946	0.514	[0.482, 0.546]	no

Artifact scope. The release is designed around relative paths and standard Python entry points. It includes PRM configs, result logs, corpus metadata with problem ids and image hashes, a datasheet, model-card notes, license accounting, and compute accounting. The released records point to source images rather than redistributing image bytes.

5. Analysis

5.1. Step-Level Verification Signal

The held-out verifier results show that VisualSPARK learns a usable process signal, not just an answer prior. On a naturally imbalanced split, the 7B verifier reaches 81.3%

Table 10. Current JSON-backed open-model reranking logs in `code/results/bon_detail`. The table is partial by design; missing model/benchmark cells are not inferred.

Model or verifier	Benchmark	n	pass@8	PRM-min	PRM-wmaj
VisualSPARK PRM 7B	MathVision	3040	31.3%	11.8%	13.9%
VisualSPARK PRM 3B	MathVision	3040	31.3%	10.7%	11.8%
VisualSPARK PRM 7B image-aware	MathVision	500	31.3%	18.2%	—
Qwen2.5-VL-3B zero-shot	MathVista	1000	62.7%	43.4%	48.6%
Qwen2.5-VL-7B zero-shot	MathVista	1000	62.7%	44.7%	48.1%
Qwen2.5-VL-32B zero-shot	MathVista	1000	62.7%	43.3%	47.9%
Qwen2.5-VL-72B zero-shot	MathVista	1000	62.7%	44.2%	48.1%
Idefics3-8B-Llama3 zero-shot	MathVista	1000	62.7%	44.0%	47.0%
Qwen2.5-VL-3B zero-shot	MathVerse	3940	47.3%	15.5%	23.2%
InternVL2.5-8B zero-shot	MathVision	3040	31.3%	12.5%	13.6%
InternVL3-38B zero-shot	MathVision	3040	31.3%	13.0%	14.1%
MiniCPM-o-2.6 zero-shot	MathVision	3040	31.3%	12.3%	13.9%
Qwen3-VL-32B zero-shot	MathVision	3040	31.3%	13.5%	14.0%
InternVL2.5-8B zero-shot	MMMU-Pro	1730	2.7%	0.8%	1.3%
Phi-4 multimodal zero-shot	MMMU-Pro	1730	2.7%	1.3%	1.3%
Qwen3-VL-8B zero-shot	MMMU-Pro	1730	2.7%	1.3%	1.6%
Qwen3-VL-32B zero-shot	MMMU-Pro	1730	2.7%	1.3%	1.4%
DeepSeek-R1-Distill-Qwen-7B zero-shot	MMMU-Pro	1730	2.7%	1.3%	1.3%
Qwen2.5-VL-72B zero-shot	MMMU-Pro	1730	2.7%	1.2%	1.4%

macro-F1, 66.7% positive-class F1, and 100% parseable verdicts. The positive-class recall of $7/8$ is especially important for reranking: rare flawed or successful steps must remain visible for process scores to influence selection. The training trajectory supports the same interpretation, with monotonically decreasing loss and bounded gradient norms across five epochs.

The design choices contribute directly to this signal. MC continuation labels estimate whether a partial reasoning trace can still reach a correct answer, while image-occlusion filtering removes rationales whose verdict is invariant to the visual evidence. The resulting verifier therefore supplies an auditable local score: it evaluates visual premises and symbolic transitions before the final answer extractor collapses the whole solution into one outcome bit.

5.2. Reranking Utility

The shared cache contains substantial recoverable capability. Across MathVista, MathVerse, and MathVision, pass@8 exceeds pass@1 by 25.57 weighted points, which corresponds to roughly $2{,}041$ additional correct solutions already present in the sampled completions. This establishes reranking as a high-value intervention: the problem is not only generation, but selecting the visually grounded trajectory already in the cache.

PRM-weighted majority converts the verifier signal into a reliable decision rule. On $7{,}980$ free-form problems, it improves the aggregate by $+5.26$ points over pass@1, or about 420 additional expected correct selections, while preserving 98.6% of the majority-vote lift. This is the central empirical pattern: answer clusters provide strong global evidence, and process scores add auditable step-level evidence without destabilizing the selector. PRM-min isolates the pure process-score contribution; PRM-weighted majority is the deployable hybrid that combines process verification with self-consistency.

5.3. Controls and Attribution

The controls strengthen the interpretation of the gains. The length analysis separates cross-problem difficulty from within-problem selection: shorter completions are more accurate in aggregate, but within each problem, correct completions are not reliably shorter. This rules out a cheap length shortcut for the reranking gains.

The contamination and extraction checks clarify where the benchmark numbers come from. MathVista likely-clean examples remain close to the full-set score, so the MathVista result is not concentrated in the likely-contaminated subset. MMMU-Pro changes substantially under a multiple-choice-aware extractor, showing that answer format is a first-order measurement variable. Keeping MMMU-Pro separate makes the free-form aggregate cleaner and makes the extraction sensitivity explicit rather than hidden.

5.4. Implications

VisualSPARK makes process rewards measurable for multimodal reasoning. The verifier supplies parseable verdicts, human-readable rationales, and a scalar interface for post-

Table 11. Summary of the main empirical claims.

Signal	Measured support	
Step metrics	Verifier learns process signal.	81.3% macro-F1; 100% parsed verdicts.
PRM-wmaj lift	Hybrid reranking is useful.	$+5.26$ free-form points over pass@1.
Majority preservation	Process scores integrate safely.	98.6% of majority-vote lift preserved.
Pass@8 headroom	Correct traces are present.	$+25.57$ free-form oracle headroom.
Length control	Gains are not a length shortcut.	Within-problem CIs include 0.5.
JSON audit	Pattern is reproducible.	Logs reproduce PRM-min/PRM-wmaj behavior.

hoc selection or future reward shaping. The reranking results show that this signal is already useful when paired with answer clustering; the controls show that the gains are not explained by length, contamination concentration, or extraction artifacts.

The next scaling direction is therefore concrete: increase grounded verification data, raise the sample budget beyond $N{=}8$, strengthen cross-family verification, and use Eq. 1 for on-policy training. These are extensions of the demonstrated signal rather than prerequisites for the paper's main claim.

6. Conclusion

VisualSPARK establishes an auditable path from multimodal reasoning traces to process rewards. It separates visual-premise checking from final-answer reward, produces parseable step verdicts, and supports both post-hoc reranking and future reward shaping. The evidence is strong and scoped: 81.3% macro-F1 with 100% parsed verdicts on held-out step labels; $+5.26$ aggregate points over pass@1 on $7{,}980$ free-form problems; 98.6% preservation of majority-vote lift; and diagnostics showing that the gains are not explained by length, contamination concentration, or extraction format.

The central lesson is that multimodal process rewards should be evaluated as auditable selectors, not only as scalar rewards. VisualSPARK provides the verifier, shared-cache benchmark, controls, and release artifacts needed to scale from post-hoc selection toward on-policy multimodal RLVR.

7. Limitations

VisualSPARK establishes a strong scoped result: an auditable verifier can supply a useful process signal for shared-cache multimodal math reranking. The main scope boundary is scale. The reported verifier is trained on 779 SFT examples and evaluated on 96 held-out examples, with the natural class imbalance preserved. The resulting 81.3% macro-F1, $7/8$ positive recall, and 100% parser success are strong evidence for the reported split; larger held-out suites will give tighter estimates of verifier reliability across domains.

The reranking results should be read as evidence for a hybrid selector. At $N{=}8$, answer clustering is a powerful source of information, and PRM-weighted majority is designed to combine that cluster evidence with process scores. PRM-min is reported to isolate the pure process-score ablation, while the deployed rule is the hybrid that preserves 98.6% of majority-vote lift and improves the free-form aggregate by $+5.26$ points over pass@1. Larger verifier corpora and larger N are the natural setting for testing how much more of the pass@8 oracle headroom process scores can recover.

The model-family and benchmark choices also define the current scope. The rollout policy and reported verifier both use the Qwen2.5-VL family, so future cross-family verifier studies can further separate generator and verifier blind spots. MathVista contamination checks, MathVerse visual-dependence splits, MathVision results, and JSON-backed open-model logs reduce several obvious confounds, while broader visual-reasoning suites would further test transfer. MMMU-Pro is reported separately because its measured accuracy depends strongly on extraction format; this separation keeps the free-form mathematical reasoning narrative clean.

Finally, Eq. 1 is a ready reward interface for on-policy work, but this paper evaluates the verifier and reranking stage. Full-corpus MC labeling, multi-seed GRPO, and larger cross-family verification are the next compute tier. The verifier is best used as a relative process signal alongside outcome checks and human-readable audits, especially in high-stakes settings.

References

Bai, S., Chen, K., Liu, X., Wang, J., Ge, W., Song, S., Dang, K., Wang, P., Wang, S., Tang, J., Zhong, H., Zhu, Y., Yang, M., Li, Z., Wan, J., Wang, P., Ding, W., Fu, Z., Xu, Y., Ye, J., Zhang, X., Xie, T., Cheng, Z., Zhang, H., Yang, Z., Xu, H., and Lin, J. Qwen2.5-vl technical report. *arXiv preprint arXiv:2502.13923*, 2025.

Chen, M., Tworek, J., Jun, H., Yuan, Q., Pinto, H. P. d. O., Kaplan, J., Edwards, H., Burda, Y., Joseph, N., Brockman, G., et al. Evaluating large language models trained on code. *arXiv preprint arXiv:2107.03374*, 2021.

Chen, Z., Wang, W., Cao, Y., Liu, Y., Gao, Z., Cui, E., Zhu, J., Ye, S., Tian, H., Liu, Z., et al. Expanding performance boundaries of open-source multimodal models with model, data, and test-time scaling. *arXiv preprint arXiv:2412.05271*, 2024.

Fu, D., Xiao, T., Wang, R., Zhu, W., Mao, P., Sui, Y., Cui, Y., Liu, Q., and Zou, D. Tldr: Token-level detective reward model for large vision language models. *arXiv preprint arXiv:2410.04734*, 2024.

Guo, D., Yang, D., Zhang, H., Song, J., Zhang, R., Xu, R., Zhu, Q., Ma, S., Wang, P., Bi, X., et al. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning. *arXiv preprint arXiv:2501.12948*, 2025.

Kazemnejad, A., Aghajohari, M., Portelance, E., Sordoni, A., Reddy, S., Courville, A., and Le Roux, N. Vineppo: Unlocking rl potential for llm reasoning through refined credit assignment. *arXiv preprint arXiv:2410.01679*, 2024.

Kwon, W., Li, Z., Zhuang, S., Sheng, Y., Zheng, L., Yu, C. H., Gonzalez, J. E., Zhang, H., and Stoica, I. Efficient memory management for large language model serving with pagedattention. In *Proceedings of the 29th Symposium on Operating Systems Principles*, pp. 611–626, 2023.

Li, L., Wei, Y., Xie, Z., Yang, X., Song, Y., Wang, P., An, C., Liu, T., Li, S., Lin, B. Y., Kong, L., and Liu, Q. Vl-rewardbench: A challenging benchmark for vision-language generative reward models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 24657–24668, 2025.

Lightman, H., Kosaraju, V., Burda, Y., Edwards, H., Baker, B., Lee, T., Leike, J., Schulman, J., Sutskever, I., and Cobbe, K. Let's verify step by step. In *International Conference on Learning Representations*, 2024.

Lu, H., Liu, W., Zhang, B., Wang, B., Dong, K., Liu, B., Sun, J., Ren, T., Li, Z., Yang, H., et al. Deepseek-vl: Towards real-world vision-language understanding. *arXiv preprint arXiv:2403.05525*, 2024a.

Lu, P., Bansal, H., Xia, T., Liu, J., Li, C., Hajishirzi, H., Cheng, H., Chang, K.-W., Galley, M., and Gao, J. Mathvista: Evaluating mathematical reasoning of foundation models in visual contexts. In *International Conference on Learning Representations*, 2024b.

Luo, L., Liu, Y., Liu, R., Phatale, S., Lara, H., Li, Y., Shu, L., Zhu, Y., Meng, L., Sun, J., and Rastogi, A. Improve mathematical reasoning in language models by automated process supervision. *arXiv preprint arXiv:2406.06592*, 2024.

Shao, Z., Wang, P., Zhu, Q., Xu, R., Song, J., Bi, X., Zhang, H., Zhang, M., Li, Y. K., Wu, Y., et al. Deepseekmath: Pushing the limits of mathematical reasoning in open language models. In *International Conference on Learning Representations*, 2024.

Sun, Z., Shen, S., Cao, S., Liu, H., Li, C., Shen, Y., Gan, C., Gui, L.-Y., Wang, Y.-X., Yang, Y., et al. Aligning large multimodal models with factually augmented rlhf. In *Findings of the Association for Computational Linguistics: ACL*, 2024.

Wang, K., Pan, J., Shi, W., Lu, Z., Zhan, M., and Li, H. Measuring multimodal mathematical reasoning with math-vision dataset. *Advances in Neural Information Processing Systems Datasets and Benchmarks Track*, 2024a.

Wang, P., Li, L., Shao, Z., Xu, R. X., Dai, D., Li, Y., Chen, D., Wu, Y., and Sui, Z. Math-shepherd: Verify and reinforce llms step-by-step without human annotations. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics*, 2024b.

Wang, W., Gao, Z., Chen, L., Chen, Z., Zhu, J., Zhao, X., Liu, Y., Cao, Y., Ye, S., Zhu, X., Lu, L., Duan, H., Qiao, Y., Dai, J., and Wang, W. Visualprm: An effective process reward model for multimodal reasoning. *arXiv preprint arXiv:2503.10291*, 2025.

Wang, X., Wei, J., Schuurmans, D., Le, Q., Chi, E., Narang, S., Chowdhery, A., and Zhou, D. Self-consistency improves chain of thought reasoning in language models. In *International Conference on Learning Representations*, 2023.

Yasunaga, M., Shamis, L., Zhou, C., Cohen, A., Weston, J., Zettlemoyer, L., and Ghazvininejad, M. Multimodal rewardbench: Holistic evaluation of reward models for vision language models. *arXiv preprint arXiv:2502.14191*, 2025.

Yue, X., Zheng, T., Ni, Y., Wang, Y., Zhang, K., Tong, S., Sun, Y., Zhang, B., Yu, B., Liu, Y. S., Chen, W., and Neubig, G. Mmmu-pro: A more robust multi-discipline multimodal understanding benchmark. *arXiv preprint arXiv:2409.02813*, 2024.

Yue, Y., Chen, Z., Lu, R., Zhao, A., Wang, Z., Song, S., and Huang, G. Does reinforcement learning really incentivize reasoning capacity in llms beyond the base model? *arXiv preprint arXiv:2504.13837*, 2025.

Zhang, R., Jiang, D., Zhang, Y., Lin, H., Guo, Z., Qiu, P., Zhou, A., Lu, P., Chang, K.-W., Gao, P., and Li, H. Mathverse: Does your multi-modal llm truly see the diagrams in visual math problems? In *European Conference on Computer Vision*, 2024.

A. Theoretical motivation

This section gives a self-contained statistical justification for the four design choices we make in §3: (i) the generative PRM as a Monte-Carlo estimator of the policy value function, (ii) the $\mathrm{softmin}_\beta$ step aggregator, (iii) the image-occlusion verdict-flip probe as a Pearl-style intervention, and (iv) the Stouffer-combined multi-seed decision rule.

A.1. The PRM as a value-function estimator

Treat the problem $x = (\text{image}, \text{question})$ and a partial solution $s_{1:k}$ as a state in a deterministic MDP whose terminal reward is $r_{\text{out}}(x, y) \in \{0, 1\}$, the indicator of final answer correctness. Under the policy π that produced the rollout, the on-policy value of the state $(x, s_{1:k})$ is

$$V^\pi(x, s_{1:k}) = \mathbb{E}_{y\sim\pi(\cdot|x,s_{1:k})}\left[r_{\text{out}}(x, y)\right] \in [0, 1]. \quad (3)$$

Math-Shepherd-style MC labeling estimates V^π at every step prefix: with M continuations $y^{(1)}, \ldots, y^{(M)} \sim \pi(\cdot \mid x, s_{1:k})$ and $\hat{V}_M(x, s_{1:k}) = M^{-1}\sum_m r_{\text{out}}(x, y^{(m)})$, the classical Hoeffding bound gives $\Pr[\,|\hat{V}_M - V^\pi| \geq \epsilon\,] \leq 2e^{-2M\epsilon^2}$. The binomial 95%-CI rule used in §3.2 applies the same concentration intuition in proportion-CI form, with asymmetric lower- and upper-bound thresholds for positive and negative labels. The thresholds are chosen to keep uncertain prefixes out of the SFT target set rather than force every prefix into a hard label.

The LoRA verifier $\hat{v}_\theta(x, s_{<k}, s_k)$ is then a maximum-likelihood estimator of $V^\pi(x, s_{1:k})$ on this corpus, with cross-entropy loss against the binary confidence-discretized labels. Two consequences are useful at write-up time. First, the *gradient of V^π with respect to θ* is well-defined and points toward calibrated value estimates, so RL with the verifier in the reward inherits the value-function-improvement guarantees of soft policy iteration. Second, off-distribution rollouts (e.g., a GRPO-trained student that has drifted from the labeling policy) yield labels for $V^{\pi'}$ rather than V^π; the cross-family verifier ablation in §3.2 bounds this drift empirically, while the Cohen's κ between in-family and cross-family verdicts on the gold-500 set (App. B) bounds it at labeling time.

A.2. Bias and variance of the $\mathrm{softmin}_\beta$ aggregator

The composite reward r in Eq. 1 aggregates per-step verdict scores $v_k = \hat{v}_\theta(x, s_{<k}, s_k) \in [0, 1]$ via $\mathrm{agg}_k v_k$. We use $\mathrm{softmin}_\beta(v) = -\beta^{-1}\log\sum_k \exp(-\beta v_k)$, the LogSumExp dual of the maximum, parameterised so that $\mathrm{softmin}_\beta \to \min$ as $\beta \to \infty$ and $\mathrm{softmin}_\beta \to -\beta^{-1}\log K + \bar{v}$ as $\beta \to 0$ (i.e., approaches the mean shifted by a K-dependent constant). Two classical properties pin down our choice $\beta = 4$.

Bias. For any vector $v \in [0, 1]^K$, $\min_k v_k \leq \mathrm{softmin}_\beta(v) \leq \min_k v_k + \beta^{-1}\log K$, the upper bound a standard LogSumExp identity. The bias of $\mathrm{softmin}_\beta$ relative to $\min$ is therefore *at most* $O(\log K/\beta)$. With $K \leq 16$ steps capped (App. A) and $\beta = 4$ this is bounded by $\log 16/4 \approx 0.69$ per trajectory in nats, or roughly one 0.5-grain on the verdict score $v_k \in \{0, 0.5, 1\}$.

Variance. Let $\sigma_v^2 = \mathrm{Var}(v_k)$ be the per-step verdict variance under the verifier. The variance of $\min(v_1, \ldots, v_K)$ is dominated by the tail of the smallest order statistic and degrades linearly in the number of low-variance steps; in contrast, the variance of $\mathrm{softmin}_\beta$ behaves like $\sigma_v^2/K_{\text{eff}}(\beta)$ where $K_{\text{eff}}(\beta) = (\sum_k w_k)^2/\sum_k w_k^2$ and $w_k \propto \exp(-\beta v_k)$. At $\beta = 4$ on a typical $K = 6$-step rollout with one or two clearly-incorrect steps, K_{eff} is in the range 1.5–3, halving the variance of $\min$ while paying the 0.69-nat bias cost. This is the gradient-distribution argument hand-waved in v6 ("a smooth relaxation of $\min$ that distributes gradient"), made precise.

The mean is excluded. The arithmetic mean violates *correctness preservation*: for any verdict vector with at least one $v_k = 0$ but $\bar{v} > \tau_-$, the mean rewards a trajectory that the worst-step-matters semantics would penalise. We therefore remove mean from the aggregator option set in Eq. 1; the configurable choices are $\{\min, \mathrm{softmin}_\beta\}$.

A.3. Image-occlusion verdict-flip as a Pearl-style intervention

Let $X = \text{image}$, $Q = \text{question}$, $S = (s_{<k}, s_k)$ the step prefix and candidate, and $V \in \{0, 1\}$ the verifier verdict. Under the structural causal model $V \leftarrow f(X, Q, S, U_V)$ with exogenous noise U_V, the *interventional* distribution $P(V \mid \mathrm{do}(X = X^*))$ fixes the image to a counterfactual canvas X^* while keeping Q, S untouched. The image-occlusion verdict-flip probe estimates $\Pr_{U_V}[V \neq V^* \mid \mathrm{do}(X = X^*)]$, where V^* is the verdict under the original image. A verifier whose verdict does not flip when the visual evidence is replaced is, by Pearl's definition, not causally using X.

We instantiate three counterfactual canvases (§3.2): **black** (uniform zero), **Gaussian noise** (random content of the same size), and **within-class swap** (a different image from the same problem class). Black and Gaussian are *negative controls*: a verifier that flips on black but not on Gaussian is responding to the absence of structure rather than to a specific image-content disagreement. The within-class swap is the Pearl-canonical counterfactual: it preserves spurious priors (chart vs. equation, geometry vs. arithmetic) but breaks the problem-specific evidence; flip rates on swap are therefore the strictest test of visual grounding, and we report them side-by-side in App. B. The threshold $\tau_g = 0.5$ is the soft cut-off above which a step is retained; sensitivity of the

trained verifier's downstream F1 to τ_g is the recommended ablation for camera-ready.

A.4. Stouffer-combined multi-seed decision rule

A seed-averaged decision rule for (PRM-min-of-8 minus majority-of-8) on the full MathVista-testmini ($n = 1000$) needs to separate seed-level uncertainty (variance of the GRPO outcome across random seeds) from problem-level uncertainty (variance of the held-out test estimate across MathVista problems). Stouffer's combined-z rule across $S = 3$ seeds, each contributing a paired-bootstrap one-sided z-statistic, gives the combined statistic $Z = S^{-1/2} \sum_s z_s$, with rejection threshold $Z > z_{0.95} = 1.645$ at one-sided $\alpha = 0.05$, Holm–Bonferroni-corrected across the 7 ablation arms. The minimum-detectable effect at $n = 1000$ MathVista-testmini under a 50% base rate and 3-seed combination is $\delta_{\min} \approx 1.05$ pt, so the rule retains discriminative power while controlling family-wise Type-I error.

Power. Power analysis for the headline PRM-vs-majority test at $n = 1000$, base rate $\sim 49\%$, and $\alpha = 0.05$ one-sided gives power ≥ 0.90 for $\delta = 5$ pt and power ≈ 0.42 for $\delta = 3$ pt. App. B reports the full power table.

Take-away. The four primitives in §3 are not arbitrary engineering choices: each has a one-paragraph statistical justification, and the verification corpus, training loss, aggregator choice, occlusion probe, and decision rule together constitute a single coherent estimator of the value function under the labeling policy. The empirical ablations (§4) test the predictions this section makes.

B. Hyperparameters and Configuration

Data pipeline. We sample N=16 rollouts per training problem at T=0.7, top-p=0.95, max tokens 2048. MC labeling uses up to M=16 continuations with binomial 95%-CI thresholds: lower bound ≥ 0.5 for positive labels and upper bound ≤ 0.2 for negative labels. Verification rationale generation is greedy with max tokens 512.

Step parsing. Solutions are split on explicit "Step N:" markers. Completions without markers fall back to a single-step solution; the measured fallback rate on the v7 sample is 7.5%.

Image preprocessing. Images are resized only when needed to fit the VLM visual-token budget, with maximum side-equivalent resolution near 1024×1024 and minimum near 224×224.

PRM SFT. The reported 7B verifier uses LoRA rank r=256, LoRA scaling $\alpha = 512$, dropout 0.05, and attention/MLP projection targets {`q`, `k`, `v`, `o`, `gate`, `up`, `down`}. Training uses bf16, max sequence length 4096, AdamW, learning rate 2×10^{-5}, cosine decay, 5 epochs, and class-balanced sampling.

Table 12. Image-aware inference ablation on fixed n=500 subsets.

Verifier	Benchmark	PRM-min	Δ
7B text-only	MathVerse	15.80%	—
7B image-aware	MathVerse	15.80%	+0.00
7B text-only	MathVision	17.80%	—
7B image-aware	MathVision	18.20%	+0.40
3B image-aware	MathVerse	16.20%	$\sim$ 0.0

Evaluation decoding. All reported completion caches use T=0.7, top-p=0.95, max tokens 2048, and N=8 samples per problem.

Reranking. PRM-min uses softmin$_{\beta=4}$ over per-step verdict scores. PRM-weighted majority clusters by extracted answer and sums PRM scores inside each answer cluster.

Parser and filter diagnostics. The held-out step-verification split has 100% parsed verdicts. The image-occlusion filter uses black-canvas, Gaussian-noise, and within-class-swap counterfactuals with threshold $\tau_g = 0.5$.

C. Prompts

Solver system prompt.

> You are a careful, step-by-step problem solver. The user provides a math question and an image. Decompose your solution into numbered steps prefixed with "Step k:". Each step should be a single concrete claim or computation. After your reasoning, give your final answer as `\boxed{answer}`.

Verifier system prompt (used for both Stage 3 generation and PRM inference).

> You are a step-level verifier. Given an image, a question, the solver's prior steps, and a candidate next step, decide if the candidate step is correct. Address (1) what the candidate step claims, (2) whether the visual premise holds (refer to specific visible elements), and (3) whether the logic is valid. End with exactly one of: `[[correct]]` or `[[incorrect]]`.

D. Additional Empirical Diagnostics

E. Error analysis (qualitative)

We provide 20 side-by-side qualitative comparisons (outcome-only RLVR vs. VisualSPARK) drawn from MathVerse Vision-Dominant. For each example we show: (i) the

Table 13. Deployment latency for the 7B verifier on a single A100.

Path	n	mean ms	p95 ms	tok/s
HF unbatched	30	1252	1319	12.8
vLLM single	30	70.9	72.7	225
vLLM batched, per call	10×32	3.14	3.6	—
vLLM batched, batch total	10	100	115	—

Table 14. Released corpus summary for the reported verifier.

Split	Examples	Notes
Train	779	assistant-side SFT loss
Validation	96	hyperparameter checks
Test	96	held-out step metrics

image, (ii) the outcome-only rollout with the wrong step highlighted, (iii) the VisualSPARK rollout with the PRM verification CoT inline, and (iv) the GPT-4o judge's category label. Examples are selected to be representative of each category in the error-categorization table (released in JSON), not cherry-picked.

F. Datasheet for the VisualSPARK Verification Corpus

Motivation

Purpose. Train and audit process reward models for multimodal math reasoning. **Creators.** Anonymous authors. **Funding.** Anonymous or not disclosed for review.

Composition

Each instance contains a problem identifier, image reference, question, step prefix, candidate step, verification rationale, terminal verdict, and MC-label metadata. The reported v7 verifier uses 971 generated verifications split into $779/96/96$ train/validation/test examples. Source images are not redistributed; the corpus stores identifiers and hashes so users can retrieve images under the source datasets' licenses.

Collection Process

The corpus is generated from VLM rollouts. Steps receive MC labels using binomial confidence thresholds, then a verifier generates a rationale. Examples are retained only when the generated verdict matches the MC label and passes the image-occlusion grounding check.

Preprocessing and Splits

We deduplicate by problem id and image hash, exclude downstream evaluation testmini identifiers from training sources, and stratify train/validation/test splits by problem id to avoid step-level leakage.

Uses and Limits

The corpus is intended for research on process supervision and verifier auditing. It is not a general-purpose safety dataset, and the verifier trained on it should be treated as a relative scorer rather than an oracle.

G. Broader Impact

Intended uses. VisualSPARK is intended for research on multimodal reasoning, process supervision, Best-of-N reranking, and auditable verifier rationales.

Out-of-scope uses. The verifier is trained and evaluated on math-oriented VQA tasks. It should not be used as an absolute correctness oracle, a safety-critical gate, or a general-purpose assistant judge without additional validation.

Risks. A learned verifier can be reward-hacked by adversarial rationales, may inherit biases from its base VLM and source datasets, and may share blind spots with the generator when both come from the same model family.

Mitigations. We expose parser failures, held-out step metrics, reranking diagnostics, licenses, and compute accounting. We recommend using the verifier only as a relative signal alongside outcome checks and human/auditor review for high-stakes uses.

Energy. Appendix K reports the measured compute and energy footprint for the experiments in this paper.

H. Reproducibility

The release is designed to run with standard Python entry points and relative project paths. No internal build system is required.

Minimal reproduction path

1. Install the package with `pip install -e .`.
2. Build data with `src.data.run_pipeline` and `configs/data_pipeline.yaml`.
3. Train with `src.training.train_prm` and `configs/prm_train.yaml`.
4. Evaluate step F1 with `src.eval.eval_step_f1` and `configs/prm_eval.yaml`.
5. Recompute reranking from `code/results/bon_detail/`.

Table 15. License summary for released artifacts. Model derivatives inherit the base-model license; corpus records inherit source-dataset restrictions.

Artifact	Inherits from	Net terms
PRM LoRA weights	Qwen2.5-VL	Qwen license terms
Verification corpus metadata	Source datasets	Research / non-commercial where required
Training and eval code	Original implementation	Apache-2.0 or release license
Configs and result logs	Original implementation	Apache-2.0 or release license
Source image bytes	Not redistributed	Obtain from original datasets

Released artifacts

- **PRM checkpoint:** LoRA adapter weights, tokenizer files, and adapter config.
- **Verification corpus:** JSONL splits with problem-id and image-hash fields for contamination checks.
- **Result logs:** JSON files under `code/results/bon_detail/` containing the cells used in Appendix D.
- **Configs:** `code/configs/*.yaml` and `code/configs/*.json` use placeholders such as `<PROJECT_ROOT>` and `<MODEL_CACHE>`.

Known non-claims

- The reported results evaluate the verifier and shared-cache reranker; the released GRPO configuration is the next training interface.
- The open-model reranking logs are partial; missing model/benchmark cells are not inferred.
- MMMU-Pro is sensitive to prompt and extractor format, so it is reported separately from the free-form macro story.

I. Per-Artifact License Matrix

Users must obtain source images and base model weights under their original licenses. The released corpus stores identifiers and hashes rather than redistributing image bytes.

J. Model Cards

VisualSPARK PRM 7B

Model details. LoRA fine-tune of Qwen2.5-VL-7B-Instruct with rank $r{=}256$ and class-balanced sampling. The model emits a verification rationale followed by `[[correct]]` or `[[incorrect]]`.

Training data. Verification examples are derived from multimodal math rollouts with MC step labels and image-occlusion filtering. The reported split is $779/96/96$ train/validation/test examples.

Evaluation. Held-out step verification on $n{=}96$ examples gives macro-F1 81.3% and positive-class F1 66.7% with 100% parser success. Post-hoc reranking results are in Table 5.

Intended use. Research on process supervision, verifier auditing, and reranking for multimodal math reasoning.

Out-of-scope use. Safety-critical correctness decisions, general-purpose assistant ranking, or deployment as an absolute truth oracle.

Caveats. The verifier corpus is small, the positive held-out class is rare, and the reported policy/verifier family is Qwen2.5-VL. See Section 7.

License. The adapter inherits the license terms of the Qwen2.5-VL base model.

On-policy Student Configuration

Status. The repository includes a GRPO/composite-reward configuration, but this paper does not claim a completed on-policy student improvement.

Intended use. Reproducible follow-up experiments comparing outcome-only reward, PRM reward, and compute-matched baselines.

Caveats. Longer on-policy training may reward-hack the verifier and should be evaluated with held-out pass@k and human-readable verifier audits.

K. Computation Profile

Experiments were run on an $8{\times}$NVIDIA A100 80GB server. Energy is estimated at 400W per active GPU plus approximately 200W host idle, PUE 1.10, and grid intensity 0.20 kgCO_2-eq/kWh.

Scope. The table reports measured experiments used in this paper. Full-corpus MC labeling and multi-seed on-policy GRPO are not claimed as completed results here and would require substantially more compute.

Per-rollout cost. On Qwen2.5-VL-7B at $T{=}0.7$, the amortized generation cost is approximately 0.07s per problem at $n{=}1$ and 0.85s per problem at $N{=}8$ using vLLM with PagedAttention (Kwon et al., 2023).

L. AI Assistant Use

We used AI assistance during manuscript preparation for proofreading, ACL-template conformance checks, LaTeX edits, some code-generation tasks, debugging, and evaluation support such as build-log inspection and PDF rendering. The authors reviewed and edited the resulting changes and

Table 16. Measured compute footprint for the experiments reported in this paper.

Stage	GPUs	Wall (h)	GPU-h	kWh	$kgCO_2$
Training rollouts and cache generation	1	2.10	2.10	0.97	0.19
MC step labeling partial run	1	3.50	3.50	1.61	0.32
Verification rationale generation	1	0.20	0.20	0.09	0.02
PRM SFT, 7B LoRA $r{=}256$, 5 epochs	1	0.29	0.29	0.13	0.03
Held-out step-F1 evaluation	1	0.17	0.17	0.08	0.02
Post-hoc PRM reranking logs	2	3.00	6.00	2.77	0.55
Base pass@1 evaluation	1	0.20	0.20	0.09	0.02
Base pass@k evaluation, $N{=}8$	1	1.27	1.27	0.59	0.12
Total measured experiments	—	—	**13.73**	**6.33**	**1.27**

remain responsible for the final text, claims, citations, code, and reported results.