MM-DATALOOP: CLOSING THE DATA-SIDE LOOP FOR RECURSIVE SELF-IMPROVEMENT IN MLLM POST-TRAINING

Anonymous authors
Paper under double-blind review

ABSTRACT

Recursive self-improvement (RSI) offers a path toward automated training loops for large models. Existing research focuses primarily on the model side, while data preparation remains a labor-intensive part of MLLM post-training. Data-side methods typically address individual operations, leaving the composition of heterogeneous interventions and the accumulation of cross-round experience less explicitly organized. We present MM-DataLoop, an end-to-end iteration loop for MLLM post-training data. It groups data operations into five families and runs an Observe $\rightarrow$ Diagnose $\rightarrow$ Plan $\rightarrow$ Execute $\rightarrow$ Verify $\rightarrow$ Reflect cycle. A global rule memory carries experience into subsequent plans. We hold the base-model initialization, training hyperparameters, and evaluation protocol fixed while varying the training data and its mixture. This contract makes data interventions comparable and auditable across rounds. We run 31 rounds on Qwen3.5-0.8B across MM-Bench, ChartQA, MMMU, MathVista, and RealWorldQA. The five-benchmark average improves from 57.72% zero-shot to 62.95% (+5.23 pp). The study also yields reusable empirical findings about filtering and data composition. For example, removing different subsets flagged by one groundedness filter produces opposite effects. Some subsets judged low-quality also provide useful visual diversity, and removing them reduces performance in the tested recipes. We will release the code, data, and empirical findings to support further data-side research.

1 INTRODUCTION

Recursive self-improvement (RSI) seeks successive improvements through evaluation and revision under a fixed protocol. It offers a route toward automated training with less human intervention. Most recent work targets the *model* side. One line trains on model-generated supervision through self-feedback, adversarial or preference-based self-play, self-rewarding critics, and repeated trajectory scoring. An agentic line uses research assistants to propose and evaluate experiments (Lu et al., 2024a). These approaches place model development at the center of iteration. Data preparation remains labor-intensive, but has received less attention as a closed loop. We study how to make training data itself the object of this iterative process.

Existing data-side methods address instruction tuning (Liu et al., 2023; 2024a), richer captioning (Chen et al., 2024a), and targeted synthesis (Shi et al., 2024b; Masry et al., 2022). Others select data for quality or diversity (Gadre et al., 2023; Penedo et al., 2024; Xie et al., 2023b), or optimize mixture weights (Xie et al., 2023a). Connecting these techniques requires a surrounding *loop*. A shared vocabulary organizes operations, and a cross-round protocol lets later plans use earlier evidence. A fixed evaluation protocol makes their effects comparable. Without these connections, local improvements remain difficult to audit or accumulate across rounds.

We propose MM-DataLoop, a data-side closed-loop pipeline for MLLM post-training. Five operation families—Acquire, Synthesize, Denoise, Compose, and Codify—organize the available interventions. A six-step cycle connects Observe, Diagnose, Plan, Execute, Verify, and Reflect. Plan reads codified rules from global memory, and Reflect updates them after both successful and failed candidates. A *frozen-vs-mutable contract* holds the base-model initialization, training hyperparameters, and evaluation protocol fixed while the training data and mixture change. This makes score changes comparable against recorded interventions and keeps planning decisions traceable to prior evidence. A five-tag diagnostic vocabulary structures the feedback into Plan (§3.4); residual untagged errors remain recorded as round-level anomalies.

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.

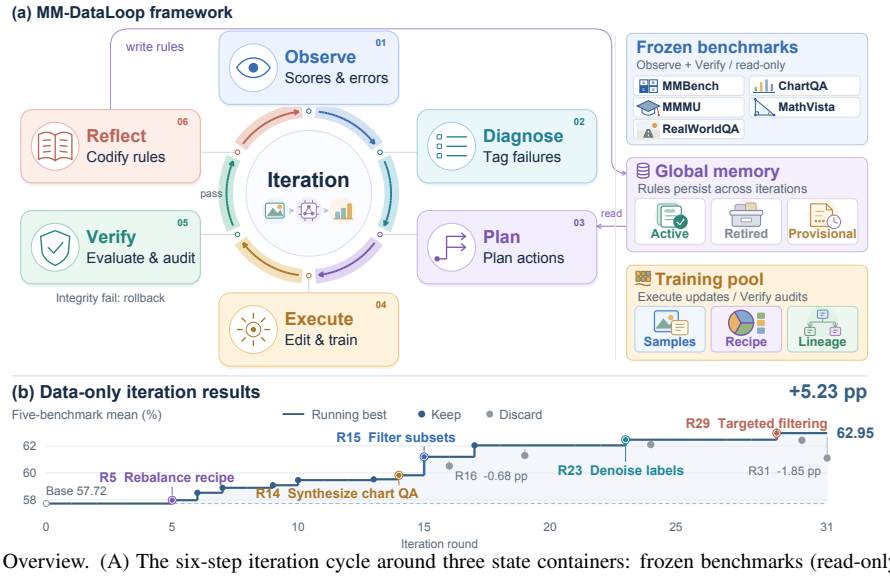

Figure 1: Overview. (A) The six-step iteration cycle around three state containers: frozen benchmarks (read-only), global memory (Reflect writes, Plan reads), and the training pool (Execute writes, Verify audits). A failed Verify rolls back the candidate model and training pool. Reflect retains the failure record in memory before the next round. (B) Five-benchmark mean over 31 rounds: 57.72% → 62.95%, with key rounds annotated.

We run a controlled 31-round study on Qwen3.5-0.8B across MMBench (Liu et al., 2024b), ChartQA (Masry et al., 2022), MMMU (Yue et al., 2024), MathVista (Lu et al., 2024b), and RealWorldQA (xAI, 2024). The five-benchmark average rises from 57.72% at R0 to 62.95% at R29 (+5.23 pp). Gains concentrate in MMBench, ChartQA, and RealWorldQA; MathVista returns to baseline, while MMMU declines. The trajectory also yields findings on teacher filtering, visual-diversity preservation, and answer-length trade-offs. The results section reports their benchmark-level effects and supporting rounds. Our contributions are threefold.

• **A data-side closed-loop framework.** MM-DataLoop unifies five data-operation families and a six-step iteration cycle with a global rule memory. A frozen-vs-mutable contract makes interventions comparable across rounds and provides a record for auditing their outcomes.

• **A controlled 31-round iteration study.** MM-DataLoop improves the average score across five multimodal benchmarks by 5.23 pp. The base-model initialization, training hyperparameters, and evaluation protocol remain fixed throughout the study. Each round links its measured score changes to a recorded training-data intervention.

• **Cross-round reusable empirical findings and supporting artifacts.** The findings span data-operation families and emphasize paired comparisons for attributing changes to individual interventions. Each finding identifies the rounds from R1–R31 that support or refute it. We will release MM-DataLoop, the data recipes, training and evaluation logs, and accumulated rules as public infrastructure for data-side RSI research.

2 Related Work

Model-Side Recursive Self-Improvement. Recent RSI research includes methods that iterate the model or policy using its own outputs. Examples include self-rewarding training (Yuan et al., 2024), self-play (Chen et al., 2024d), and reinforced self-training (Gülçehre et al., 2023; Singh et al., 2024). Rationale distillation (Zelikman et al., 2022; Yuan et al., 2023) and constitutional self-critique (Bai et al., 2022) also use generated feedback or supervision. Reflexion uses linguistic feedback and episodic memory to improve agent performance across trials (Shinn et al., 2023), while the AI Scientist organizes research into an autonomous experiment loop (Lu et al., 2024a). MM-DataLoop focuses this iteration on data, holding the base-model initialization, training hyperparameters, and evaluation protocol fixed.

MLLM Post-Training & Data Recipes. MLLM post-training uses instruction corpora alongside efficient adaptation methods such as LoRA (Hu et al., 2022). Data collections range from general

Table 1: Overview of Data Operations. In I/O, X = sample, Q/A = question/answer, T = teacher.

Operation	Family	Input → Output	Main Use
Dataset Ingestion	Acquire	URL → Asset	Source Selection, License, Versioning
Source Segmentation	Acquire	Doc → Units	Segmentation, OCR, Frame Extraction
Teacher QA Rewriting	Synthesize	$(Q, A) \rightarrow (Q', A')$	Form / Style / Length Alignment
CoT Distillation	Synthesize	$(Q, T) \rightarrow (Q, \text{CoT} + A)$	Rejection-Sampled CoT Distillation
MCQ Conversion	Synthesize	$(Q, A) \rightarrow (Q, \text{opts})$	Free-Form → MCQ
Instruction Evolution	Synthesize	$Q \rightarrow Q'$	Difficulty / Diversity Expansion
Rule-based Filtering	Denoise	$X \rightarrow$ Keep/Drop	Length / Language / Corrupt-Image
Structural Repair	Denoise	$X \rightarrow X'$	`<image>` Repair, Format Align
Verifier Screening	Denoise	$(X, T) \rightarrow$ Keep/Drop	Teacher-as-Verifier
Contamination Scan	Denoise	Pool × Bench → Hits	Byte-Level Leakage Scan
Mixture Weighting	Compose	Subsets → Recipe	Static Subset Weighting
Curriculum Design	Compose	Pool → Ordered Recipe	Staged / Leveled Schedule
Rule Codification	Codify	Obs. → Rule Entry	Experience → Memory

instruction corpora (Liu et al., 2023; 2024a; Chen et al., 2024a) to domain-specific pools (Shi et al., 2024b; Masry et al., 2022). More recent systems combine broader collections with stronger backbones (Chen et al., 2024c; Wang et al., 2024b; Bai et al., 2025; Tong et al., 2024). Recipe studies compare data mixtures and their effects on model performance (McKinzie et al., 2024; Laurençon et al., 2024; Deitke et al., 2024; Li et al., 2025; Shi et al., 2024a). MM-DataLoop adds a record of how each result informs subsequent data revisions, so the trajectory can support continued search.

Data Curation, Agent Memory, & Evaluation Integrity. Data curation studies selection (Gadre et al., 2023; Penedo et al., 2024; Xie et al., 2023b), mixture optimization (Xie et al., 2023a; Ye et al., 2024), and processing pipelines (Chen et al., 2023). Instruction and reasoning synthesis provide additional supervision (Wang et al., 2023; Xu et al., 2024a; Wei et al., 2022; Yuan et al., 2023; Xu et al., 2024b). Agent-memory research stores experience across trajectories (Wang et al., 2024a; Park et al., 2023) or formalizes persistent memory and agent roles (Packer et al., 2023; Hong et al., 2024). Evaluation work addresses integrity and standardized protocols (Chen et al., 2024b; Duan et al., 2024), while MLAgentBench studies agents executing research tasks (Huang et al., 2024). MM-DataLoop connects data operations to persistent rules and verification records within one iteration loop.

3 MM-DataLoop

3.1 Overview

MM-DataLoop combines a shared operation vocabulary (§3.2) with a feedback path into planning (§3.3). Diagnostic tags turn observed failures into planning inputs, while global memory carries supported experience across rounds. Verify gates candidate acceptance on evaluation results and data integrity. Figure 1 shows these connections.

3.2 Data Operation Families

Data operations fall into five families according to what they change. The classification applies to the operation rather than the tool: generating samples is Synthesize, while judging or removing existing samples is Denoise. Each supported operation has one family assignment.

Acquire brings external assets into the pipeline through ingestion, extraction, or OCR, recording their source, license, and preprocessing version. ***Synthesize*** creates new examples from existing assets, including rewritten QA, CoT demonstrations, and MCQs. Each output records its source model, prompt, and verification trace. ***Denoise*** repairs, normalizes, or removes existing samples through rules or model judgments.

Compose changes collection-level properties such as subset ratios, length distribution, difficulty, and curriculum to form candidate recipes. ***Codify*** turns experience into structured rules and provides the write operation for global memory. Table 1 lists the supported operations. Contamination Scan belongs to Denoise but runs at Verify, to record audit metadata without modifying samples.

3.3 DATA ITERATION CYCLE

The cycle follows Observe → Diagnose → Plan → Execute → Verify → Reflect. Round t produces candidate model $\hat{F}_t$ and pool $\hat{P}_t$. The retained state is (F_t, P_t, M_t), comprising the accepted model, pool and recipe, and memory. R0 denotes the original base checkpoint F_0; B is the fixed benchmark suite. A rejected candidate leaves the accepted model and pool unchanged. Its results remain in the experiment history, and Reflect retains the failure in memory.

Observe evaluates the retained model and records scores and errors. ***Diagnose*** assigns multi-label failure tags D_t (§3.4). ***Plan*** reads these tags, retrieved rules R_t, current scores, and experiment history to produce hypotheses H_t and operations Π_t. ***Execute*** writes the candidate pool and trains $\hat{F}_t$ from F_0 with a fresh LoRA adapter. ***Verify*** records candidate scores and data-integrity results in J_t (§3.6). ***Reflect*** updates memory after accepted and rejected candidates alike. Valid runs can support positive or negative findings; invalid runs support pipeline-repair lessons, not claims about recipe effectiveness.

Each step uses the same execution mechanism across rounds. Observe, Verify, Reflect, and Diagnose's per-sample comparison use deterministic scripts. Execute invokes scripted data operations or the fixed Qwen3.8-27B synthesis teacher with hardcoded prompts and decoding parameters. A chat-mode LLM performs Diagnose's five-tag classification and generates Plan's hypotheses and operation sequences. Its operating protocol fixes the failure vocabulary, operation menu (Table 1), and maximum of three operations per round. Table 6 specifies the fixed and mutable components.

Algorithm 1: One Round of MM-DataLoop

Input: original base F_0; retained state $(F_{t-1}, P_{t-1}, M_{t-1})$; benchmarks B; failure vocabulary V; experiment history $\mathcal{H}_{t-1}$.
Output: retained state (F_t, P_t, M_t) and history $\mathcal{H}_t$.
Notation: S_t^{obs}, E_t = observed scores and errors; S_t = candidate scores; D_t = tags; R_t = retrieved rules; H_t, Π_t = hypotheses and operations; J_t = verification record.

1. $(S_t^{\text{obs}}, E_t) \leftarrow \textsc{Evaluate}(F_{t-1}, B)$ ▷ Observe
2. $D_t \leftarrow \textsc{Diagnose}(E_t; V)$ ▷ Diagnose
3. $R_t \leftarrow \textsc{Retrieve}(M_{t-1}; D_t, B)$ ▷ Active rules in scope
4. $(H_t, \Pi_t) \leftarrow \textsc{Plan}(D_t, R_t, S_t^{\text{obs}}, \mathcal{H}_{t-1})$ ▷ Plan
5. $\hat{P}_t \leftarrow \textsc{Execute}(P_{t-1}, \Pi_t)$; $\hat{F}_t \leftarrow \textsc{Train}(F_0, \hat{P}_t)$ ▷ Fresh adapter
6. $J_t \leftarrow \textsc{Verify}(\hat{F}_t, \hat{P}_t, B)$; $S_t \leftarrow J_t.\text{scores}$ ▷ Candidate results
7. $(F_t, P_t) \leftarrow (F_{t-1}, P_{t-1})$ ▷ Retain previous state
8. **if** $\neg\textsc{Discard}(J_t) \wedge \bigwedge_{k=1}^{3} g_k(J_t)$ **then** ▷ Accept if gates pass
9. $\quad (F_t, P_t) \leftarrow (\hat{F}_t, \hat{P}_t)$
10. **end if**
11. $\mathcal{H}_t \leftarrow \mathcal{H}_{t-1} \cup \{(t, H_t, \Pi_t, \hat{P}_t, J_t)\}$ ▷ Archive every candidate
12. $M_t \leftarrow \textsc{Codify}(M_{t-1}, H_t, J_t)$ ▷ Reflect on every round

3.4 FAILURE DIAGNOSTIC VOCABULARY

Diagnose uses three sample-level tags and two aggregate tags. *Format-violation* marks outputs that violate the required answer format. *Hallucination* marks content unsupported by visual or factual evidence. *Reasoning-skip* marks a missing intermediate step when an explicit reasoning trace is required. At the aggregate level, *length-drift* marks a shift in answer-length distribution, and *task-mix-mismatch* marks task-level changes associated with the data mixture. Tags may co-occur. Errors not covered by the vocabulary remain recorded as anomalies.

3.5 GLOBAL MEMORY SCHEMA

Global memory stores rules and retains failure records across rounds. Each rule carries the fields in Table 2. Reflect writes rules through Codify; Plan retrieves entries whose scope matches the current task and whose status is *active*. Codify can create, update, or retire a rule. Retired entries remain available for audit but are excluded from default planning reads.

For example, experience from a failed rewriting attempt can motivate a rule favoring verification-based filtering. The entry records its supporting rounds and applicable data settings. A later plan can

Table 2: Global Memory Fields. Entries link operational recommendations to their evidence and scope.

Field	Semantics
text	natural-language rule with an explicit applicability condition
evidence	rounds supporting or refuting the rule; positive and negative evidence share this set
scope	applicability tags: benchmark, task form, data-operation family, or model configuration
status	*active* / *retired* / *provisional*; only *active* rules participate in default planning reads
hint	operational hint to next-round planning, mappable to operations or families of Table 1
lineage	creation/update context: round index, triggering hypothesis, verification-snapshot pointer

retrieve that rule when considering label-denoise. Contrary evidence can narrow its scope or retire it. Retirement preserves the evidence record without continuing to recommend the rule.

3.6 Built-In Verification Gates

Verify separates the evaluation verdict from data-integrity checks. The evaluation compares candidate and reference scores using MMBench and ChartQA as primary benchmarks. An improvement on either is *keep*, or *partial* if a secondary benchmark regresses beyond its threshold; otherwise the verdict is *discard*. Appendix B specifies the score thresholds. This verdict differs from the five-benchmark mean used to rank reported configurations.

Three integrity checks yield pass bits g_1, g_2, g_3. *Contamination Scan* records benchmark matches and their disposition. *Verifier Spot-Check* re-judges new or edited samples, stratified by source and operation. *Targeted Regression Check* checks each operation's interface behavior. Acceptance requires a non-discard verdict and $\bigwedge_{k=1}^{3} g_k = 1$. Failure retains the previous model and pool, while Reflect still records the outcome. Appendix B describes the checks.

Let $v_t \in \{\text{keep}, \text{partial}, \text{discard}\}$ denote the evaluation verdict. The acceptance indicator is

$$a_t = \mathbf{1}[v_t \in \{\text{keep}, \text{partial}\}] \prod_{k=1}^{3} g_k(J_t). \tag{1}$$

Thus, a favorable score comparison cannot override a failed integrity check. Conversely, passing all integrity checks does not establish that a candidate improves evaluation performance. The checks assess different properties of the same candidate, so each outcome remains separately available in the round's verification record J_t.

Acceptance changes the retained model and pool together:

$$(F_t, P_t) = \begin{cases} (\hat{F}_t, \hat{P}_t), & a_t = 1, \\ (F_{t-1}, P_{t-1}), & a_t = 0. \end{cases} \tag{2}$$

This state update does not carry trained weights into the next training run. Execute always initializes that run from F_0; the retained pool and recipe provide the starting point for the next data intervention. The candidate's verification record remains available even when its model and pool are rejected.

The checks inspect different properties. Verifier Spot-Check records disagreement with teacher decisions on changed samples. Targeted Regression Check tests whether operations produce the specified outputs, such as reproducible filtering decisions or the planned mixture. Training and evaluation then assess the candidate's downstream performance.

For example, a candidate with improved scores but a failed operation check is rejected under Equation 1. Its record can inform a repair to the data-processing pipeline. A candidate that passes the integrity checks but receives a discard verdict instead provides evidence about the tested recipe. Both outcomes remain available to subsequent planning, with their different failure contexts preserved in the experiment history.

4 Iteration Study Setup

We run a controlled 31-round iteration study on five multimodal benchmarks. The base-model initialization, training hyperparameters, and evaluation protocol remain fixed. The training data and its mixture vary across rounds. This paper reports the first 31 rounds of the search. R31 is the reporting cutoff, not a convergence criterion; further rounds can continue from the retained data recipes and memory.

Base Model & Training Protocol. The base model is Qwen3.5-0.8B (instruct), with its weight version locked to a fixed fingerprint before R0. Each round starts from the same original base checkpoint with a newly initialized LoRA adapter; no trained weights or adapters carry over from earlier rounds. We use LoRA (r=32, α=64) to limit per-round cost. Training uses AdamW with a cosine schedule, an effective batch size of 128, three epochs, and a cutoff length of 2,048. The implementation stack is described in Appendix D. The synthesis and verification teacher is Qwen3.8-27B, with its user-facing version fixed. This version lock does not guarantee identical outputs across repeated calls. The initial training pool draws from widely-used open instruction corpora—LLaVA (Liu et al., 2023; 2024a), ShareGPT4V (Chen et al., 2024a), MathV360K (Shi et al., 2024b), and ChartQA (Masry et al., 2022)—to ensure a reproducible and representative starting point; Acquire runs during R1–R19, with no external data added thereafter.

Frozen Evaluation. Evaluation covers the five benchmarks in Table 3, including MCQ and free-form tasks. Their different answer formats motivate the length-drift diagnostic. Inference, prompts, post-processing, and answer matching remain fixed across rounds. The released four-benchmark evaluation package uses a SHA-256 fingerprint; MMMU uses its separate fixed evaluation kit. Final scores use rule-based matching rather than an LLM judge, although teacher ability can still affect the training data. Absolute scores may differ from official model reports, so comparisons use the same evaluation protocol throughout.

Scoring & Seed Protocol. $S_{t,b}$ is candidate round t's accuracy on benchmark b, in percent; $\bar{S}_t$ is the five-benchmark mean. Tables retain rejected candidates' scores. Δ R0 compares with the base, while Δ Prev compares with the preceding numbered round. Operation-level comparisons name their reference recipe explicitly. The four-benchmark suite averages seeds 42 and 43; MMMU uses two evaluation runs. Incomplete evaluations are excluded. R2, R3, and R23 have additional same-checkpoint evaluations, as detailed in Appendix E.

Compute Budget. A coarse budget estimate assumes about 2 h of training on 8 A100s and 1 h of evaluation on 2 A100s per round. Across 31 rounds, this gives approximately 580 A100-hours: 496 for training, 62 for evaluation, and about 20 for synthesis. These are planning estimates rather than a sum of measured job durations.

Variance Disclosure. Table 6 distinguishes fixed settings from evolving recipes and rules. The synthesis teacher and planning LLM can produce different outputs across calls, even under a fixed operating protocol. Repeated evaluation of one checkpoint measures evaluation variability, not variation in data generation, planning, or training. Candidate comparisons nevertheless use the same scoring procedure.

5 Study Results & Trajectory

5.1 Overview

R29 achieves the highest average in the reported 31 rounds: 62.95, compared with 57.72 at R0 (+5.23 pp). Table 3 gives the benchmark-level decomposition. MMBench, ChartQA, and RealWorldQA improve; MathVista effectively matches baseline, while MMMU declines. The positive results span CircularEval consistency across option permutations, free-form chart answers, and real-world scene MCQ. Thus, the gains cover several evaluation formats, with gains of different sizes.

MMMU Residual. Two factors may help explain the MMMU results. The training pool emphasizes visual math, charts, and short-answer VQA, whereas MMMU covers college-level reasoning across disciplines. This distribution difference may limit transfer. The fixed evaluation kit also represents some open-ended items as pseudo-MCQ questions (Appendix F). This affects how answers are assessed, but is not an immutable performance ceiling and does not by itself explain changes relative to R0.

R23 matches the MMMU baseline at 42.67, but the later synthesis, filtering, and subset-removal variants score lower. R27–R29 all remain near 41 despite their different removal sets. Under the equal-weight average, R29 improves on R23 by 0.49 pp, with MathVista +1.83 and RealWorldQA

Table 3: The five frozen benchmarks with capability, answer format, and per-benchmark absolute scores at R0 (baseline) and R29 (best-avg); Δ = R29 $-$ R0 (pp). Splits: MMBench en/dev (group-level under CircularEval, 1,164 groups), ChartQA test (2,500), MMMU val (900), MathVista testmini (1,000), RealWorldQA test (765).

Benchmark	Capability	Format	R0	R29	Δ
MMBench	General VQA	MCQ	55.11	73.50	$+18.39$
ChartQA	Chart, Structured Reading	Short Free-Form	72.64	78.52	$+5.88$
MMMU	Knowledge-Intensive Reasoning	MCQ + Free-Form	42.67	41.11	-1.56
MathVista	Visual Math	MCQ + Short Free-Form	58.40	58.50	$+0.10$
RealWorldQA	Real-World Scene	MCQ	59.80	63.14	$+3.34$
Average			**57.72**	**62.95**	$\mathbf{+5.23}$

$+1.41$, despite MMMU -1.56. The trajectory therefore records a benchmark trade-off rather than uniform improvement.

5.2 Five-Stage Trajectory

Table 4 reports the round-by-round average score. The 31 rounds partition into five descriptive stages, placed where the trajectory shifts direction or a qualitatively new operation enters (full per-round recipe listing in Appendix E).

Stage 1: Cold-Start Collapse (R1–R4). The initial three-source mixture scores below baseline. R2 adds ShareGPT4V caption data (average answer length about 910 characters) while reducing MathV, and ChartQA and RealWorldQA fall sharply. R3 introduces an image-token placement bug and also performs poorly. The failure motivates format validation, introduced at R6, after which the same problem does not recur through R31.

Stage 2: Slow Climb (R5–R14). R5 is the first round to exceed R0. Subsequent mixture adjustments gradually improve the average, with MCQ conversion and chart-computational synthesis added at R13–R14. By R14, the gain over R0 reaches 2.10 pp. The next stage tests whether subset filtering can improve on these mixture adjustments.

Stage 3: Subset Filtering Breakthrough (R15–R17). Filtering eight MathV360K subsets at R15 improves the average by 1.37 pp over R14. Extending the exclusion list to 11 subsets at R16 reduces performance. R17 returns to the eight-subset filter and uses the full remaining pool, reaching 62.04. These comparisons motivate the later analysis of filtering granularity and retained visual diversity.

Stage 4: Plateau & Verify-Only Denoise (R18–R26). The additions and removals tested at R18–R22 remain below R17. R23 then applies verification-based filtering: the teacher flags labels for removal without rewriting answers. The average reaches 62.46, a gain of 0.42 pp over R17. Restoration and synthesis variants at R24–R26 do not exceed this average. Section 6 compares the effects of rewriting and filtering.

Stage 5: Factor Isolation (R27–R31). R27's groundedness filter raises MathVista but lowers MMMU. R28 and R29 separate two removal groups: A, external-knowledge samples mainly from ScienceQA; and B, chart/table samples flagged as defective. Removing A alone at R29 yields the best reported average, 62.95. R30 and R31 extend the removal policy and both regress. These comparisons motivate the A/B analysis in §6.2. The full trajectory remains non-monotonic, with cold-start failures, a plateau, and later regressions alongside the gains.

5.3 Cross-Round Operation Footprint

Table 5 summarizes where each operation family appears. Compose assembles recipes throughout the search, while Denoise supplies the filtering interventions at the major turning points. Codify preserves lessons from failures and successful interventions for later plans.

Implicit Component Evidence. Three cases illustrate the loop's feedback paths. After R18–R22 remain below R17, Diagnose identifies label noise and Plan selects verification-based filtering at R23 ($+0.42$ pp vs R17). R27's mixed result motivates the revised hypotheses tested in R28 and R29, illustrating how reflection informs later plans. Following R3's format failure, a validation gate is introduced at R6; the same failure does not recur through R31. These cases connect diagnosis,

Table 4: 31-round average-score trajectory (pp), split R0–R15 (top) / R16–R31 (bottom); *Key op* annotates seven pivot rounds. Full per-round recipe listing in Appendix E.

	R0	R1	R2	R3	R4	R5	R6	R7	R8	R9	R10	R11	R12	R13	R14	R15
Avg	57.72	54.97	44.70	44.97	48.21	57.97	58.52	58.89	58.68	59.08	59.46	59.42	59.18	59.52	59.82	61.19
Δ Prev	—	−2.75	−10.27	+0.27	+3.24	+9.76	+0.55	+0.37	−0.21	+0.40	+0.38	−0.04	−0.24	+0.34	+0.30	+1.37
Δ R0	—	−2.75	−13.02	−12.75	−9.51	+0.25	+0.80	+1.17	+0.96	+1.36	+1.74	+1.70	+1.46	+1.80	+2.10	+3.47
Key op				<img> bug												8-subset

	R16	R17	R18	R19	R20	R21	R22	R23	R24	R25	R26	R27	R28	**R29**	R30	R31
Avg	60.51	62.04	61.66	61.29	61.76	61.86	61.40	62.46	62.10	62.21	62.02	62.36	61.90	**62.95**	62.42	61.10
Δ Prev	−0.68	+1.53	−0.38	−0.37	+0.47	+0.10	−0.46	+1.06	−0.36	+0.11	−0.19	+0.34	−0.46	**+1.05**	−0.53	−1.32
Δ R0	+2.79	+4.32	+3.94	+3.57	+4.04	+4.14	+3.68	+4.74	+4.38	+4.49	+4.30	+4.64	+4.18	**+5.23**	+4.70	+3.38
Key op		full 192K						denoise-vo				ground-flt	−B	**−A**		

Table 5: Cross-round footprint of the five data-operation families over R1–R31.

Family	Rounds	Role In Trajectory
Acquire	R1–R19	Builds the initial pool; absent after plateau
Synthesize	R13–R26	Task-specific augmentation once base recipe stabilizes
Denoise	R15–R31	Drives both step-changes (subset filtering, verify-only relabeling, groundedness filtering)
Compose	R1–R31	Recipe assembly throughout; ratio tuning in early stages
Codify	R3–R27	Writes validation gates and operational rules into memory

retained experience, and subsequent actions. They are not controlled single-component ablations, so they do not quantify the independent contribution of each pathway.

6 Empirical Findings

The six findings below concern targeted synthesis, filtering decisions, and recipe trade-offs. Each identifies its supporting rounds and scope. Appendix F provides additional comparisons.

6.1 Targeted Synthesis

Targeted Synthesis With A 27B Teacher. Task-specific data synthesized by the 27B teacher (chart-computational augmentation, MCQ conversion) improves its target benchmark in both of the comparisons below. R13→R14's ChartComp 1.6K raises ChartQA $74.72 \rightarrow 77.10$ ($+2.38$ pp)—one of the largest single-benchmark jumps in the study; R14→R15's filter pushes ChartQA to 78.00. On the saturated R23 base, adding 3.1K more ChartComp (R25) lifts ChartQA to 78.84 (all-time high) but averages -0.25 pp vs R23. "Unsaturated" is judged by whether successive rounds continue to improve the average score.

6.2 Filtering Decisions

Label Rewriting And Verification-Based Filtering. Rewriting and filtering make different changes to supervision. The rewriting attempt associated with R15 changed about 87% of answers and was abandoned after problematic rewrites were observed. R23 instead retains the original answers and removes samples flagged as incorrect at confidence 0.9 or above. It removes 31.0% of the input pool and improves the average by 0.42 pp over R17. Restoring 2,449 removed MapQA samples at R24 reduces the average by 0.36 pp relative to R23. These observations support the tested filtering configuration, without establishing that rewriting fails in general or that the confidence threshold must be set to 0.9.

Separate Criteria For Label Correctness And Grounding. Label verification checks answer correctness; groundedness screening asks whether the image provides enough information to answer the question. These criteria can guide separate filtering operations. R27–R29 compare removals centered on external-knowledge samples, mainly ScienceQA (A), and other flagged samples (B). Relative to R23, R27 scores -0.10 pp, R28 -0.56 pp, and R29 $+0.49$ pp. R29 improves MathVista by 1.83 pp and RealWorldQA by 1.41 pp. R28 retains ScienceQA but still scores 41.00 on MMMU, so deleting ScienceQA is not a necessary condition for that decline. These comparisons do not establish statistical independence between the filters.

Provenance Filtering And Quality Scores. Provenance filtering selects subsets by image-path prefix. Quality-score filtering removes samples below a threshold on an LLM-generated score. R15's 8-subset filter yields $+1.37$ pp; R17 keeps that filter and expands to full scale, adding $+0.85$ pp. R19's quality-score diagnosis labels 82.6% of 192K MathV360K samples low-quality. Inspection links these scores to short answers (average 20 characters), raising concern that length drives the judgments. Removing the lowest-scored bucket at R22 reduces the average by 0.64 pp. These results favor provenance filtering in the tested short-answer setting, where a low quality score can reflect answer format.

6.3 Recipe Trade-offs

Filtering Granularity And Visual Diversity. Apparently low-quality subsets can carry useful visual diversity. Removing FigureQA/IconQA/Super-CLEVR at R16 reduces the average by 0.68 pp; removing MapQA at R22 costs 0.64 pp. Fully removing ScienceQA at R31 costs 1.85 pp. R21, R24, and R28 further examine subset removal or restoration (Appendix F). The effect depends on which samples are removed: R23's verification-based filter removes 31% without lowering MMMU, and removing flagged ScienceQA samples at R29 improves the average. Whole-subset removal and selective filtering therefore require separate comparisons.

Answer-Length Trade-offs In The Tested Recipes. Several recipes show trade-offs between long-answer supervision and short-answer performance. R2 adds 30K ShareGPT4V samples (average 910 characters), with ChartQA/RealWorldQA about 24/26 pp below baseline. R19 adds 15K complex-reasoning samples (500–1200 characters), and MathVista falls 2.10 pp relative to R17. R26 replaces 8,999 geometry answers with teacher CoT, reducing MathVista/MMMU by 0.42/1.45 pp relative to R23. These interventions differ in source, composition, and length, so they do not isolate answer length alone. R8 uses raw CoT, followed by condensed versions at R9–R11. R9 averages 0.19 pp above R7, although other mixture changes prevent attribution to CoT length alone.

Remaining Headroom. R29 is the best observed configuration within the first 31 rounds. The reported search does not establish the maximum achievable score on any benchmark. Further data search can continue beyond R31.

Paired-Round Attribution Standard. Paired comparisons help distinguish individual interventions from changes to an entire recipe. R21/R22 compare removals involving MapQA and non-math residue; R27–R29 examine the groundedness-filter groups; R30/R31 test extensions of R29's removal policy. These designs provide more focused evidence than adjacent-round deltas alone. Comparisons that change several recipe dimensions remain trajectory observations rather than single-variable causal tests. The record motivates controlled component and search-policy comparisons, but does not replace them.

7 Conclusion

The 31-round record shows how data operations improve the five-benchmark average under fixed base-model initialization, training hyperparameters, and evaluation protocol. Operation families organize the interventions, and the feedback loop carries each outcome into subsequent planning. Paired-round comparisons help interpret their effects. Global memory retains rules from accepted and rejected candidates, keeping later decisions traceable to earlier experiments. The recipes, logs, and rules support continued iteration from R29. Future studies can reuse these artifacts with other models or benchmarks under a newly defined experimental contract.

Limitations. The findings in §6 are bounded by one base model (Qwen3.5-0.8B), five benchmarks, SFT training, and teacher-based data operations. A single-base trajectory cannot separate model-specific effects from general data-side effects. Preference-based post-training (Rafailov et al., 2023) remains outside the study's scope. Other settings may require an expanded failure vocabulary. The operation families, paired comparisons, and rule memory provide reusable structure, but the direction and magnitude of individual findings need validation under each new contract. Fixed training hyperparameters do not imply equal training cost across rounds: changing the pool size changes the number of examples processed over three epochs. The compute estimate describes the overall search budget, rather than a matched-cost comparison between recipes. Benchmark reuse risks overfitting.

AI USE STATEMENT

We used generative AI tools for writing assistance, literature search, and implementation support. For writing, the tools helped draft and revise passages, polish language, and improve readability and organization. For literature search and discovery, they helped identify potentially relevant work for consideration in the paper. AI tools also assisted with code revisions. The Method and Setup sections describe the language-model operations within MM-DataLoop.

We reviewed AI-assisted manuscript revisions and checked their consistency with the reported experimental records. The authors retain responsibility for the selection and interpretation of cited work, the accuracy of citations, and the wording of scientific claims, and the correctness of the implementation. We take responsibility for the final content of this paper, including material prepared with generative AI assistance.

ETHICS STATEMENT

This study uses publicly available datasets and synthetic examples derived from them, as described in Appendix D. We respect the source datasets' licenses and research-use restrictions; any released data or derived artifacts will be subject to the applicable redistribution terms.

REPRODUCIBILITY STATEMENT

To support reproducibility, we specify the training and evaluation settings in Section 4, the fixed and mutable components in Appendix C, and the complete 31-round recipe record in Appendix E. We will open-source the pipeline code, data-processing scripts, and training and evaluation configurations, and release per-round recipes, checkpoint fingerprints, training and evaluation logs, and rule records, together with instructions for reproducing the experiments. Data releases will respect the original sources' licenses. As discussed in Appendix G, rerunning evaluation on a fixed checkpoint and independently retraining a recipe are distinct reproducibility settings; training and teacher-generation variability may lead to differences in reproduced scores.

REFERENCES

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. *arXiv preprint arXiv:2502.13923*, 2025.

Yuntao Bai, Saurav Kadavath, Sandipan Kundu, Amanda Askell, Jackson Kernion, Andy Jones, Anna Chen, Anna Goldie, Azalia Mirhoseini, Cameron McKinnon, et al. Constitutional AI: Harmlessness from AI feedback. *arXiv preprint arXiv:2212.08073*, 2022.

Daoyuan Chen, Yilun Huang, Zhijian Ma, Hesen Chen, Xuchen Pan, Ce Ge, Dawei Gao, Yuexiang Xie, Zhaoyang Liu, Jinyang Gao, et al. Data-juicer: A one-stop data processing system for large language models. *arXiv preprint arXiv:2309.02033*, 2023.

Lin Chen, Jinsong Li, Xiaoyi Dong, Pan Zhang, Conghui He, Jiaqi Wang, Feng Zhao, and Dahua Lin. ShareGPT4V: Improving large multi-modal models with better captions. In *European Conference on Computer Vision (ECCV)*, 2024a.

Lin Chen, Jinsong Li, Xiaoyi Dong, Pan Zhang, Yuhang Zang, Zehui Chen, Haodong Duan, Jiaqi Wang, Yu Qiao, Dahua Lin, and Feng Zhao. Are we on the right way for evaluating large vision-language models? In *Advances in Neural Information Processing Systems (NeurIPS)*, 2024b.

Zhe Chen, Jiannan Wu, Wenhai Wang, Weijie Su, Guo Chen, Sen Xing, Muyan Zhong, Qinglong Zhang, Xizhou Zhu, Lewei Lu, Bin Li, Ping Luo, Tong Lu, Yu Qiao, and Jifeng Dai. InternVL: Scaling up vision foundation models and aligning for generic visual-linguistic tasks. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2024c.

Zixiang Chen, Yihe Deng, Huizhuo Yuan, Kaixuan Ji, and Quanquan Gu. Self-play fine-tuning converts weak language models to strong language models. In *International Conference on Machine Learning (ICML)*, 2024d.

Matt Deitke, Christopher Clark, Sangho Lee, Rohun Tripathi, Yue Yang, Jae Sung Park, Mohammadreza Salehi, Niklas Muennighoff, Kyle Lo, Luca Soldaini, et al. Molmo and PixMo: Open weights and open data for state-of-the-art multimodal models. *arXiv preprint arXiv:2409.17146*, 2024.

Haodong Duan, Junming Yang, Yuxuan Qiao, Xinyu Fang, Lin Chen, Yuan Liu, Xiaoyi Dong, Yuhang Zang, Pan Zhang, Jiaqi Wang, Dahua Lin, and Kai Chen. VLMEvalKit: An open-source toolkit for evaluating large multi-modality models. In *ACM International Conference on Multimedia (MM)*, 2024.

Samir Yitzhak Gadre, Gabriel Ilharco, Alex Fang, Jonathan Hayase, Georgios Smyrnis, Thao Nguyen, Ryan Marten, Mitchell Wortsman, Dhruba Ghosh, Jieyu Zhang, Eyal Orgad, Rahim Entezari, Giannis Daras, Sarah Pratt, Vivek Ramanujan, Yonatan Bitton, Kalyani Marathe, Stephen Mussmann, Richard Vencu, Mehdi Cherti, Ranjay Krishna, Pang Wei Koh, Olga Saukh, Alexander Ratner, Shuran Song, Hannaneh Hajishirzi, Ali Farhadi, Romain Beaumont, Sewoong Oh, Alexandros Dimakis, Jenia Jitsev, Yair Carmon, Vaishaal Shankar, and Ludwig Schmidt. DataComp: In search of the next generation of multimodal datasets. In *Advances in Neural Information Processing Systems (NeurIPS), Datasets and Benchmarks Track*, 2023.

Caglar Gülçehre, Tom Le Paine, Srivatsan Srinivasan, Ksenia Konyushkova, Lotte Weerts, Abhishek Sharma, Aditya Siddhant, Alex Ahern, Miaosen Wang, Chenjie Gu, et al. Reinforced self-training (ReST) for language modeling. *arXiv preprint arXiv:2308.08998*, 2023.

Sirui Hong, Mingchen Zhuge, Jonathan Chen, Xiawu Zheng, Yuheng Cheng, Ceyao Zhang, Jinlin Wang, Zili Wang, Steven Ka Shing Yau, Zijuan Lin, et al. MetaGPT: Meta programming for a multi-agent collaborative framework. In *International Conference on Learning Representations (ICLR)*, 2024.

Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. LoRA: Low-rank adaptation of large language models. In *International Conference on Learning Representations (ICLR)*, 2022.

Qian Huang, Jian Vora, Percy Liang, and Jure Leskovec. MLAgentBench: Evaluating language agents on machine learning experimentation. In *International Conference on Machine Learning (ICML)*, 2024.

Hugo Laurençon, Léo Tronchon, Matthieu Cord, and Victor Sanh. What matters when building vision-language models? In *Advances in Neural Information Processing Systems (NeurIPS)*, 2024.

Bo Li, Yuanhan Zhang, Dong Guo, Renrui Zhang, Feng Li, Hao Zhang, Kaichen Zhang, Peiyuan Zhang, Yanwei Li, Ziwei Liu, and Chunyuan Li. LLaVA-OneVision: Easy visual task transfer. *Transactions on Machine Learning Research (TMLR)*, 2025.

Haotian Liu, Chunyuan Li, Qingyang Wu, and Yong Jae Lee. Visual instruction tuning. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2023.

Haotian Liu, Chunyuan Li, Yuheng Li, and Yong Jae Lee. Improved baselines with visual instruction tuning. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2024a.

Yuan Liu, Haodong Duan, Yuanhan Zhang, Bo Li, Songyang Zhang, Wangbo Zhao, Yike Yuan, Jiaqi Wang, Conghui He, Ziwei Liu, Kai Chen, and Dahua Lin. MMBench: Is your multi-modal model an all-around player? In *European Conference on Computer Vision (ECCV)*, 2024b.

Chris Lu, Cong Lu, Robert Tjarko Lange, Jakob Foerster, Jeff Clune, and David Ha. The AI scientist: Towards fully automated open-ended scientific discovery. *arXiv preprint arXiv:2408.06292*, 2024a.

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

Ahmed Masry, Do Xuan Long, Jia Qing Tan, Shafiq Joty, and Enamul Hoque. ChartQA: A benchmark for question answering about charts with visual and logical reasoning. In *Findings of the Association for Computational Linguistics (ACL)*, 2022.

Brandon McKinzie, Zhe Gan, Jean-Philippe Fauconnier, Sam Dodge, Bowen Zhang, Philipp Dufter, Dhruti Shah, Xianzhi Du, Futang Peng, Floris Weers, et al. MM1: Methods, analysis & insights from multimodal LLM pre-training. In *European Conference on Computer Vision (ECCV)*, 2024.

Charles Packer, Sarah Wooders, Kevin Lin, Vivian Fang, Shishir G. Patil, Ion Stoica, and Joseph E. Gonzalez. MemGPT: Towards LLMs as operating systems. *arXiv preprint arXiv:2310.08560*, 2023.

Joon Sung Park, Joseph C. O'Brien, Carrie J. Cai, Meredith Ringel Morris, Percy Liang, and Michael S. Bernstein. Generative agents: Interactive simulacra of human behavior. In *ACM Symposium on User Interface Software and Technology (UIST)*, 2023.

Guilherme Penedo, Hynek Kydlíček, Loubna Ben Allal, Anton Lozhkov, Margaret Mitchell, Colin Raffel, Leandro Von Werra, and Thomas Wolf. The FineWeb datasets: Decanting the web for the finest text data at scale. In *Advances in Neural Information Processing Systems (NeurIPS), Datasets and Benchmarks Track*, 2024.

Rafael Rafailov, Archit Sharma, Eric Mitchell, Stefano Ermon, Christopher D. Manning, and Chelsea Finn. Direct preference optimization: Your language model is secretly a reward model. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2023.

Min Shi, Fuxiao Liu, Shihao Wang, Shijia Liao, Subhashree Radhakrishnan, De-An Huang, Hongxu Yin, Karan Sapra, Yaser Yacoob, Humphrey Shi, et al. Eagle: Exploring the design space for multimodal LLMs with mixture of encoders. *arXiv preprint arXiv:2408.15998*, 2024a.

Wenhao Shi, Zhiqiang Hu, Yi Bin, Junhua Liu, Yang Yang, See-Kiong Ng, Lidong Bing, and Roy Ka-Wei Lee. Math-LLaVA: Bootstrapping mathematical reasoning for multimodal large language models. In *Findings of the Association for Computational Linguistics (EMNLP)*, 2024b.

Noah Shinn, Federico Cassano, Ashwin Gopinath, Karthik Narasimhan, and Shunyu Yao. Reflexion: Language agents with verbal reinforcement learning. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2023.

Avi Singh, John D. Co-Reyes, Rishabh Agarwal, Ankesh Anand, Piyush Patil, Xavier Garcia, Peter J. Liu, James Harrison, Jaehoon Lee, Kelvin Xu, et al. Beyond human data: Scaling self-training for problem-solving with language models. *Transactions on Machine Learning Research (TMLR)*, 2024.

Shengbang Tong, Ellis Brown, Penghao Wu, Sanghyun Woo, Manoj Middepogu, Sai Charitha Akula, Jihan Yang, Shusheng Yang, Adithya Iyer, Xichen Pan, Austin Wang, Rob Fergus, Yann LeCun, and Saining Xie. Cambrian-1: A fully open, vision-centric exploration of multimodal LLMs. *arXiv preprint arXiv:2406.16860*, 2024.

Guanzhi Wang, Yuqi Xie, Yunfan Jiang, Ajay Mandlekar, Chaowei Xiao, Yuke Zhu, Linxi Fan, and Anima Anandkumar. Voyager: An open-ended embodied agent with large language models. *Transactions on Machine Learning Research (TMLR)*, 2024a.

Peng Wang, Shuai Bai, Sinan Tan, Shijie Wang, Zhihao Fan, Jinze Bai, Keqin Chen, Xuejing Liu, Jialin Wang, Wenbin Ge, Yang Fan, Kai Dang, Mengfei Du, Xuancheng Ren, Rui Men, Dayiheng Liu, Chang Zhou, Jingren Zhou, and Junyang Lin. Qwen2-VL: Enhancing vision-language model's perception of the world at any resolution. *arXiv preprint arXiv:2409.12191*, 2024b.

Yizhong Wang, Yeganeh Kordi, Swaroop Mishra, Alisa Liu, Noah A. Smith, Daniel Khashabi, and Hannaneh Hajishirzi. Self-instruct: Aligning language models with self-generated instructions. In *Annual Meeting of the Association for Computational Linguistics (ACL)*, 2023.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Brian Ichter, Fei Xia, Ed H. Chi, Quoc V. Le, and Denny Zhou. Chain-of-thought prompting elicits reasoning in large language models. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2022.

xAI. RealWorldQA. `https://huggingface.co/datasets/xai-org/RealworldQA`, 2024.

Sang Michael Xie, Hieu Pham, Xuanyi Dong, Nan Du, Hanxiao Liu, Yifeng Lu, Percy Liang, Quoc V. Le, Tengyu Ma, and Adams Wei Yu. DoReMi: Optimizing data mixtures speeds up language model pretraining. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2023a.

Sang Michael Xie, Shibani Santurkar, Tengyu Ma, and Percy Liang. Data selection for language models via importance resampling. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2023b.

Can Xu, Qingfeng Sun, Kai Zheng, Xiubo Geng, Pu Zhao, Jiazhan Feng, Chongyang Tao, Qingwei Lin, and Daxin Jiang. WizardLM: Empowering large pre-trained language models to follow complex instructions. In *International Conference on Learning Representations (ICLR)*, 2024a.

Guowei Xu, Peng Jin, Hao Li, Yibing Song, Lichao Sun, and Li Yuan. LLaVA-CoT: Let vision language models reason step-by-step. *arXiv preprint arXiv:2411.10440*, 2024b.

Jiasheng Ye, Peiju Liu, Tianxiang Sun, Yunhua Zhou, Jun Zhan, and Xipeng Qiu. Data mixing laws: Optimizing data mixtures by predicting language modeling performance. *arXiv preprint arXiv:2403.16952*, 2024.

Weizhe Yuan, Richard Yuanzhe Pang, Kyunghyun Cho, Xian Li, Sainbayar Sukhbaatar, Jing Xu, and Jason E. Weston. Self-rewarding language models. In *International Conference on Machine Learning (ICML)*, 2024.

Zheng Yuan, Hongyi Yuan, Chengpeng Li, Guanting Dong, Keming Lu, Chuanqi Tan, Chang Zhou, and Jingren Zhou. Scaling relationship on learning mathematical reasoning with large language models. *arXiv preprint arXiv:2308.01825*, 2023.

Xiang Yue, Yuansheng Ni, Kai Zhang, Tianyu Zheng, Ruoqi Liu, Ge Zhang, Samuel Stevens, Dongfu Jiang, Weiming Ren, Yuxuan Sun, Cong Wei, Botao Yu, Ruibin Yuan, Renliang Sun, Ming Yin, Boyuan Zheng, Zhenzhu Yang, Yibo Liu, Wenhao Huang, Huan Sun, Yu Su, and Wenhu Chen. MMMU: A massive multi-discipline multimodal understanding and reasoning benchmark for expert AGI. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2024.

Eric Zelikman, Yuhuai Wu, Jesse Mu, and Noah D. Goodman. STaR: Bootstrapping reasoning with reasoning. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2022.

A Failure Diagnostic Vocabulary: Full Definitions

This appendix gives the full definition of each of the five diagnostic tags introduced in §3.4; the main text lists tag names and one-line summaries only.

Format-Violation. The output does not match the format required by the task even when its content is otherwise correct—MCQ answers not following the option-letter format, JSON responses missing fields or with mistyped keys, numeric answers with extra units, or answers in a language that does not match the prompt. This tag concerns formal compliance only, not truth of content, and typically maps to Structural Repair or Rule-based Filtering.

Length-Drift. Output length systematically deviates from the target benchmark's expected answer-length distribution, shifting training signal or evaluation calibration. The verdict uses a per-benchmark length-tolerance interval derived from the baseline checkpoint's output statistics. It can co-fire with format-violation but is distributional rather than per-sample.

Hallucination. The output contains objects, attributes, relations, or assertions without visual or factual support—for example, claims about elements not present in the image, or content factually wrong yet unrelated to the image. It targets *something that should not appear* appearing in the output, and is complementary to reasoning-skip's focus on *something that should appear* being absent from an answer that requires intermediate steps.

Reasoning-Skip. In tasks requiring an explicit reasoning process, the output is missing at least one key intermediate step, so the answer cannot be reconstructed from the trace alone—e.g., a numeric result without the calculation, a skipped intermediate reading in a multi-step chart query, or a comparison-task conclusion without the comparison itself. This tag applies only to CoT or long-answer tasks and is the primary diagnostic trigger for Synthesize-layer interventions.

Task-Mix-Mismatch. Task-mix-mismatch describes task-level score changes that are not adequately explained by the other diagnostic tags and may reflect changes in the training mixture. It is assessed from benchmark or category deltas, recipe changes, and error clusters rather than a single sample. For example, a decline concentrated in one task after changing source proportions can motivate a Compose intervention. The tag guides planning but does not identify the cause.

B Verification Checks And Evaluation Thresholds

Verify combines an evaluation verdict with three data-integrity checks. The score-comparison script treats MMBench and ChartQA as primary benchmarks. MMMU, MathVista, and RealWorldQA are secondary. A positive primary delta yields keep unless a secondary benchmark exceeds its regression threshold, in which case it yields partial. With no positive primary delta, the verdict is discard. Regression thresholds are 2 pp for MMBench, MMMU, and RealWorldQA, and 3 pp for ChartQA and MathVista. These thresholds apply to score changes. The integrity checks are described separately below.

Byte-Level Contamination Scan. For every sample in the round's training pool, the scan performs bidirectional byte-level matching against the questions and answers of all frozen benchmarks. On the text side, matching uses a normalized byte sequence with rolling-hash window intersection and a minimum coverage threshold. On the image side no pixel-level perceptual hashing is performed; only traceable metainformation (URL, file hash, paired caption) participates—*this deliberately favors precision over recall* so as not to misclassify legitimate reuse of publicly shared images as contamination. All hits retain sample-pair ID, hit position, coverage span, source benchmark, and disposition (isolate, drop, or keep), and are written into the training pool's lineage.

Verifier Model Spot-Check. For samples *newly introduced or edited in the round*, one independently sampled re-judgment is performed, stratified by source, operation type, and family. The Verifier Model version is aligned with Execute so judgments remain comparable. The spot-check does not modify the training pool; its output is a set of metrics—pass rate, agreement with Execute-stage decisions, and family-wise pass-rate differences—with thresholds anchored to historical rates. Its purpose is to catch early signals of Verifier Model drift.

Targeted Regression Check. For every data operation newly introduced or edited in the round, a regression case set is generated to confirm the operation's actual behavior still meets its interface contract—*these checks verify contract compliance, not whether the operation brought a final evaluation gain*. Cases are organized by family: Acquire on metainformation and license completeness; Synthesize on lineage traceability; Denoise on reproducibility of keep/drop decisions; Compose on recipe conformance to Plan's output; and Codify on field completeness of rule entries.

C Frozen-Vs-Mutable Contract

Table 6 separates fixed settings from the data recipes and rules that evolve across rounds. Evaluation code is checked by fingerprint where supported; training hyperparameters are fixed in the submission configuration. Teacher version and decoding settings are held constant, but generated outputs may still vary. The planning agent follows a fixed protocol; its model identity is recorded rather than byte-fingerprinted. Evaluation repeatability does not bound these generation effects.

Table 6: Frozen-vs-mutable contract for the 31-round iteration study.

	Component	Enforcement
Frozen	Base-Model Initialization	Qwen3.5-0.8B weight fingerprint locked before R0
	LoRA Hyperparameters	String literals in submission script; no override path
	Training Hyperparameters	Fixed optimizer, schedule, batch size 128, 3 epochs, cutoff 2048
	Evaluation Code + Protocol	SHA-256 fingerprint (§4)
	Benchmark Suite (5)	Fixed list; no additions or replacements across R1–R31
	Synthesis Teacher (Qwen3.8-27B)	Fixed model path, hardcoded prompts and decoding config
	Planning Protocol	Fixed SOP, failure vocab (§3.4), op menu (Table 1), max-3-op
Mutable	Training Pool	The single object under optimization
	Data Recipe	Round-indexed operation record derived from pool
	Codified Rules	Append / update / retire per round via Reflect

D Infrastructure & Data Sources: Full Record

This appendix expands §4 into the full record of the infrastructure stack, teacher-model version freezing, data-source list, and licensing/contamination audit.

Training & Evaluation Infrastructure. Training uses an in-house fine-tuning framework built on an open-source LLM training stack; evaluation inference is carried by vLLM. Both versions are held constant across the iteration, with a deterministic sampling configuration (temperature $= 0$, top-p $= 1$, fixed max-new-tokens per benchmark). All Synthesize and Verifier Screening calls use Qwen3.8-27B, whose version is frozen before R1; the teacher's ability ceiling is a conditioning factor of the empirical results.

Data Sources & Licenses. All training data derives from public datasets, or from Synthesize outputs generated on top of them; no closed-source data enters the pool. External data sources enter through Acquire and versioned ingestion during R1–R19, after which no further data from outside the pipeline is added—every subsequent addition to the training pool comes from Synthesize derivation or Compose recomposition of existing subsets. This temporal boundary scopes the later rounds to the acquired source pool.

License and contamination were audited upon Acquire: all source datasets are publicly available under open or research-use licenses, with research-only-licensed data participating only in training and never entering the publicly released recipe; the question stems and official answers of the five frozen benchmarks were, immediately upon Acquire, subjected to full byte-level contamination scanning, with hit samples isolated from the training pool before training (scan mechanism in §B).

E Full 31-Round Record

Table 7 expands the pivot-row *Key op* annotations of Table 4 into a per-round one-line recipe label for all 31 rounds. Numerical entries (Avg, Δ Prev, Δ R0) are identical to Table 4; only the per-round recipe column is added.

Table 7: 31-round recipe listing accompanying Table 4. Recipe column gives a one-line label for each round; numerical columns repeat Table 4 for co-lookup.

Rnd	Recipe	Avg	Δ Prev	Δ R0
R0	Baseline	57.72	—	—
R1	LLaVA 100K + MathV 50K + ChartQA 500	54.97	−2.75	−2.75
R2	+ShareGPT4V 30K; MathV→14.9K	44.70	−10.27	−13.02
R3	−MathV; <image> bug	44.97	+0.27	−12.75
R4	bug fixed; MathV back	48.21	+3.24	−9.51
R5	**LLaVA 50K; MathV 75K = 126K**	**57.97**	**+9.76**	**+0.25**
R6	LLaVA 40K; MathV 100K	58.52	+0.55	+0.80
R7	LLaVA 25K; MathV 125K	58.89	+0.37	+1.17
R8	+raw CoT 6.6K	58.68	−0.21	+0.96
R9	condensed CoT; LLaVA 15K	59.08	+0.40	+1.36
R10	LLaVA 20K; MathV 123K	59.46	+0.38	+1.74
R11	CoT 7.4K	59.42	−0.04	+1.70
R12	+ShareGPT4V 15K, geo 14.5K; MathV 114.5K	59.18	−0.24	+1.46
R13	+MCQ 4.9K (synth)	59.52	+0.34	+1.80
R14	+ChartComp 1.6K (synth)	59.82	+0.30	+2.10
R15	**MathV 8-subset filter; 123K cap = 150K**	**61.19**	**+1.37**	**+3.47**
R16	filter 11 subsets	60.51	−0.68	+2.79
R17	**8-subset filter; full 192K = 219K**	**62.04**	**+1.53**	**+4.32**
R18	LLaVA 30K	61.66	−0.38	+3.94
R19	+LLaVA complex-reasoning 15K	61.29	−0.37	+3.57
R20	+MMBench MCQ 2.8K (synth)	61.76	+0.47	+4.04
R21	−MapQA 5.3K, non-math 2.2K	61.86	+0.10	+4.14
R22	−MapQA 5.3K only	61.40	−0.46	+3.68
R23	**R17 → verify-only denoise → 151K**	**62.46**	**+1.06**	**+4.74**
R24	R23 + MapQA 2.4K	62.10	−0.36	+4.38
R25	R23 + synth chart-comp 3.1K	62.21	+0.11	+4.49
R26	R23; geo 9K→27B CoT	62.02	−0.19	+4.30
R27	R23 grounded-flt; protect AI2D/TQA/LLaVA	62.36	+0.34	+4.64
R28	R23 − non-knowledge 6.1K (B)	61.90	−0.46	+4.18
R29	**R23 − ScienceQA flag 9,967 (A) = 141K**	**62.95**	**+1.05**	**+5.23**
R30	R29 − LLaVA flag 7,797	62.42	−0.53	+4.70
R31	R29 − ScienceQA rest 6,028	61.10	−1.32	+3.38

Evaluation-Rerun Records. The four-benchmark suite averages seeds 42 and 43; MMMU uses two evaluation runs. R2, R3, and R23 have additional evaluations of the same checkpoint. Their reported scores average the available complete evaluations, including four runs for R23 using the repeated seed pair. These are not four distinct seeds or independent training runs. R29 has the standard evaluation but no additional complete reevaluation. Its reported average is 0.49 pp above R23. Same-checkpoint reruns describe evaluation repeatability; they do not establish statistical significance or quantify training and teacher variability.

F Extended Findings Archive

This appendix records additional observations about format validation, mixture changes, and task-specific interventions. Each comparison is limited to the reported recipes.

Format Validation After R3. R3 contains an image-token placement error and scores 44.97 on average. R4 corrects the format and restores MathV, reaching 48.21. Because both change, the 3.24 pp recovery cannot be attributed solely to token placement. A validation gate is introduced at R6, and the same format failure does not recur through R31. This record motivates checking format validity before interpreting recipe effects.

MathV Volume And Mixture Changes. MathV increases from 75K to 100K to 125K across R5–R7, while the average rises from 57.97 to 58.52 to 58.89. LLaVA simultaneously decreases from 50K to 40K to 25K, so these comparisons measure mixture changes rather than MathV volume alone. R15 and R17 use the same eight-subset exclusion policy; expanding the retained pool from 123K to about 192K adds 0.85 pp. R18 adds 10K LLaVA to R17 and loses 0.38 pp. More data therefore does not have a uniform effect across sources.

Provenance Filtering And Quality Scores. R15's eight-subset filter improves the average by 1.37 pp over R14, and R17's larger retained pool adds 0.85 pp over R15. R19's quality-score diagnosis labels 82.6% of MathV360K samples low-quality, with short answers prominent among them. Removing MapQA at R22 reduces the average by 0.64 pp relative to R17. This contrast motivates inspecting what a quality score measures before using it to select data (§6.2).

The R17 Plateau In The Tested Recipes. R18–R22 change the R17 recipe but score lower by 0.38, 0.75, 0.28, 0.18, and 0.64 pp, respectively. R23's verification-based filtering improves on R17 by 0.42 pp; R29 adds another 0.49 pp relative to R23. R30 and R31 extend the R29 removal policy and score lower by 0.53 and 1.85 pp. This sequence identifies productive changes within the tested search path, not the best achievable result for an operation family.

Subset Exclusion And Filtering Granularity. The eight excluded MathV360K subsets are PMC-VQA, A-OKVQA, DocVQA, TextVQA, VQA2.0, VizWiz, VQA-AS, and VQA-RAD, totaling about 148K examples. R15 samples 123K examples after excluding these sources. Relative to R14, its average improves by 1.37 pp, with MathVista $+3.15$ and MMMU $+1.72$. R16 additionally excludes FigureQA, IconQA, and Super-CLEVR and loses 0.68 pp. R17 returns to the R15 exclusion policy at full retained-pool scale. These comparisons show that extending a successful filter can remove useful supervision.

Short-Answer Synthesis On R17 And R23. Adding 2.8K synthetic MCQs at R20 changes MMMU by $+0.11$ pp and MathVista by -1.15 pp relative to R17. Adding 3.1K chart-computational examples at R25 changes the same benchmarks by -1.23 and -0.77 pp relative to R23. MathVista declines in both comparisons, whereas MMMU moves in different directions. These results support a recipe-dependent trade-off, not a universal loss from synthetic short answers.

Geometry CoT Replacement At R26. R26 replaces 8,999 geometry answers with teacher-generated CoT accepted through answer-matching rejection sampling. Relative to R23, the average falls by 0.44 pp and MathVista by 0.42 pp. The accepted traces have a median length of 358 characters. Answer matching checks the final answer, not every intermediate reasoning step. This particular replacement does not improve the target metric; it does not establish that CoT distillation cannot transfer to this model.

MMMU Open-Ended Evaluation. The fixed kit represents 53 of 900 MMMU items as pseudo-MCQ questions, comparing the reference answer with "Other Answers." These items constitute about 5.9% of the benchmark. In the R27 diagnostic, 49 of them (5.4% of all items) are marked wrong, while about four match the reference. Inspection identifies genuine answer errors, so the 49 failures cannot all be treated as extraction errors. They are also not an immutable ceiling: a future model may answer more of them correctly. The same protocol applies to R0 and subsequent rounds, and its limitations alone do not explain their relative score differences.

G Reproducibility

We will release the pipeline, fixed evaluation package, per-round recipes, checkpoint fingerprints, training and evaluation logs, and rule records. Reproduction should distinguish rerunning evaluation on the same checkpoint from independently retraining a recipe. The reported same-checkpoint checks provide reference evaluation tolerances of 1.5 pp for MathVista and 0.5 pp for the other benchmarks. These tolerances are not confidence bounds for independent training or teacher-generated data.