math-eval: Reproducible Mathematical Reasoning Generation and Evaluation

Xinmu Ge
Shanghai Innovation Institute · Shanghai Jiao Tong University
g3ra1d@sjtu.edu.cn
31 August 2026 · Software documentation report

Abstract. math-eval is a software workflow for reproducible evaluation of mathematical reasoning systems. It accepts canonical JSONL records, generates model outputs through vLLM or an OpenAI-compatible API, preserves raw sample artifacts, and replays parsing and scoring independently. This separation lets an evaluation change its parser or metrics without repeating model inference. The implementation records configuration, prompt, environment, and artifact hashes; supports resumable runs and deterministic data partitions; and validates shard merges before replay. Its default math-v5.2-dual contract reports boxed-only strict results as formal scores and retains whole-response soft results for diagnosis when no complete box is present. Mathematical equivalence is evaluated with Math-Verify 0.9.0. Canonical datasets are maintained separately by math-vault, preserving their provenance and licensing records.

1. Scope and workflow

math-eval evaluates generated mathematical reasoning without coupling an evaluation result to a single model call. A canonical JSONL input contains at least an identifier, a problem, and a string answer. The generation stage uses either vLLM or an OpenAI-compatible API and writes sample-level raw JSONL artifacts. The replay stage then reads frozen artifacts, applies a parser and mathematical verifier, and writes parsed verdicts and metrics.

canonical JSONL → generation → raw artifacts → replay parser/scoring → parsed results and metrics

This arrangement is deliberate: modifying parser behavior or recomputing metrics does not re-call a model. Raw and parsed artifacts remain separate, so the source of a reported result can be replayed from the saved generation output.

2. Reproducible execution

A run records configuration and prompt snapshots, hashes, and environment state in its manifest. The storage layer writes raw data in shards and supports resuming an interrupted run with the same configuration and run identifier. When a process is interrupted or fails, the resume path can repair a damaged final in-progress JSONL line and regenerate only samples not yet committed to storage.

Independent workers may run deterministic partitions of one dataset. The partition count identifies workers, while raw shard size controls only output-file boundaries. A later merge creates one ordinary run directory. Before accepting a merge, the tool checks for missing, duplicate, incomplete, or hash-inconsistent partitions, and it rejects parts whose code revision or Python/core-package versions do not match. The manifest records checkpoint paths but does not fingerprint model-weight contents; cross-machine users must therefore use the same clean revision and ensure equivalent weights at the recorded path.

3. Parsing and scoring contract

The default parser identifier is math-v5.2-dual. Its strict result is the formal score: it selects the final complete \boxed{} answer. If a response has no complete box, strict evaluation records no candidate. The soft result may instead submit a nonempty full response to the verifier, but only as a diagnostic signal. A response truncated after a complete box still uses that completed box; an incomplete trailing box does not replace an earlier complete one.

math-eval records correct, incorrect, no-candidate, parse-error, and verification-error states separately. Parse and verification failures count as failures but are not conflated with mathematical disagreement. Prediction normalization has a five-second hard timeout; a timeout is a parse error, so a pathological symbolic expression cannot stall an entire replay.

For mathematical equivalence, the current pipeline uses Math-Verify 0.9.0 with shared extraction configuration for gold answers and predictions. Metrics are recomputed from parsed verdicts, including accuracy, pass@k, and failure rates. The implementation also retains earlier parser identifiers when an older evaluation contract must be reproduced.

4. Data boundary and documented use

math-eval does not define a new source dataset. math-vault maintains curated, traceable snapshots of public mathematical reasoning datasets and records their provenance and licensing. Its canonical JSONL files can be used directly as math-eval inputs. This boundary keeps data conversion and source records separate from generation and scoring software.

The paper *Towards Understanding On-Policy Distillation through the Lens of Test-Time Scaling* documents a use of math-eval and math-vault in an evaluation setting. That reference is a documented use case rather than an endorsement or a claim about the general applicability of the artifacts.

5. Availability and citation

Software documentation report citation: Ge, X. (2026). *math-eval: Reproducible Mathematical Reasoning Generation and Evaluation*. Software documentation report, published 31 August 2026.

Related software artifact citation: Ge, X. (2026). *math-eval: Reproducible Mathematical Reasoning Generation and Evaluation* (Version v0.1.0). Zenodo. https://doi.org/10.5281/zenodo.21411208.

Source code: https://github.com/Geraldxm/math-eval
Related software artifact DOI: https://doi.org/10.5281/zenodo.21411208
License: Apache-2.0
Build this report: uv run --script docs/build_report.py

References

[1] X. Ge. math-vault: Curated, Traceable Snapshots of Public Mathematical Reasoning Datasets, Version v0.1.0, 2026. Zenodo. https://doi.org/10.5281/zenodo.21411214.

[2] X. Ge et al. Towards Understanding On-Policy Distillation through the Lens of Test-Time Scaling, 2026. arXiv:2608.11829. https://arxiv.org/abs/2608.11829.

[3] Hynek Kydlicek. Math-Verify, Version 0.9.0, 2026. Python package. https://pypi.org/project/math-verify/0.9.0/.