Sonnet 5

Performance on the CRED single-error benchmark

Overall performance

Sonnet 5 detects 77.5% of the injected errors, but its ability to explore files does not ensure systematic cross-checking. Tested as a Claude Code agent at high effort, it is much stronger at comparing prose with tables than at checking whether the code implements the paper's stated methods.

This assessment uses the CRED taxonomy and controlled error-injection framework introduced in *Verifying the Verifiers*. The agent can inspect the manuscript, analysis scripts and rendered tables, with a supplied execution verdict. The test comprises 100 host papers, each in two separately evaluated single-error versions: 200 trials with known defects.

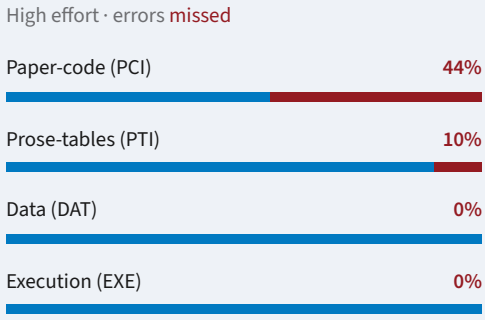
The reconstructed list-price cost is $0.487 per paper. That is a comparison measure, not a new charge for the subscription-based run. Only one Sonnet 5 effort setting is represented here; the result is for this model-and-harness configuration.

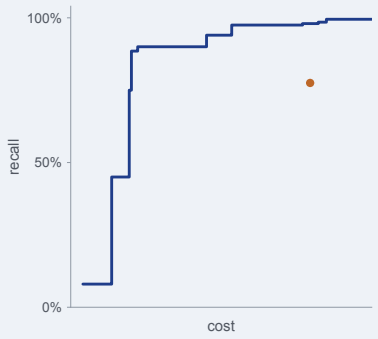
Against the 06-SEP-2026 frontier, cheaper single-shot configurations do better. GLM 5.3 Flash high detects 88.5% for less than a cent per paper. Kimi K3 reaches 98% at $0.390, also less than Sonnet's cost. The gap is therefore not simply a trade-off between price and recall.

This does not show that agentic verification is inherently inefficient. It shows that this configuration is outperformed in observed recall and reconstructed cost; the same frontier also contains stronger agentic systems. Model choice and how the model is used both matter.

CRED chapter performance and frontier position

High effort · errors missed





Sonnet catches most prose-table contradictions, but misses almost half the paper-code mismatches and sits below the frontier.

Where it is weak

Paper-code inconsistencies account for 40 of 45 scored misses. Sonnet misses 44% of that chapter, against 10% of prose-table inconsistencies. This is a difference in within-chapter performance, not merely a consequence of there being more paper-code tests.

Variable construction is the largest missed subtype: 22 of the 40 PCI misses. The other PCI misses concern sample or treatment definitions and estimation choices. The recurring issue is connecting a specific methodological promise to the operations that actually construct or analyze the data.

All injected data-integrity and execution defects receive detection credit. Execution cases, however, include a supplied code-running verdict. Success there does not demonstrate independent replication. The pattern is selective coverage: strong results on some checks coexist with many missed implementation mismatches.

A symptomatic miss

In a benchmark copy of a study of SNAP reporting rules, the paper says it combines two education groups using an equal-weighted average of their labor-turnover rates. The code instead sums hires, separations and employment across groups before computing the rate. This weights each group's rate by its employment, not equally.

Those constructions can differ whenever group sizes and rates differ. No new regression is needed to establish the mismatch: the claimed aggregation and the implemented arithmetic are different. Whether correcting it changes the policy estimate would require a separate analysis.

Sonnet returns two other findings, about comparison states and an unsupported event-study claim, but omits the aggregation mismatch. It can identify sophisticated methodological issues while leaving a basic outcome definition unreconciled. This is an observed gap in coverage, not evidence that the model cannot understand weighted averages.

Based on the framework in *Verifying the Verifiers*. Scores follow CRED's published matcher, which can under-credit valid detections. Costs use dated price lists; comparisons are descriptive.
Live benchmark