
Rewards and benchmarks

Key takeaways

1. Reinforcement learning post-trains a model against a reward someone chose. When the reward is a test suite, a program grades every attempt.
2. A SWE-bench task is a real merged pull request cut in two: the issue is the prompt and the maintainer's tests are the grader, with no partial credit.
3. A benchmark is useful until its answers leak into the training data. SWE-bench measured coding for three years, then its coauthors stopped reporting it.

Common misunderstandings of how models are scored

Which claim would you defend right now? Choose before the evidence.

“A reward has to come from a person; a program cannot judge whether an answer is good.”

“A patch that fixes the reported bug but breaks one unrelated test still earns most of the credit.”

“A benchmark score describes the model, so it stays valid as long as the model does.”

Keep your choice. Look for the mechanism or counterexample that would make it fail.

Post-training, step 2: reinforcement learning

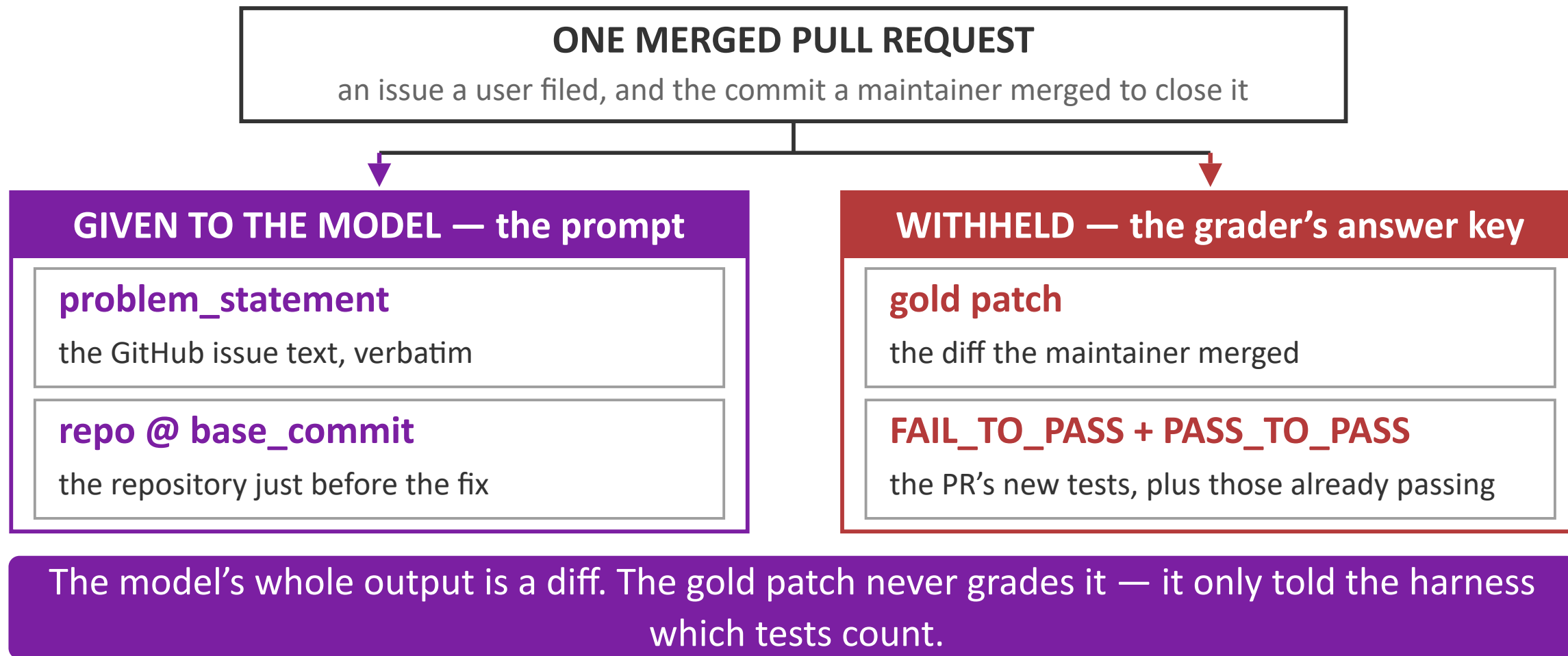
The model is optimized against a reward, and the reward is a design decision

Step	What happens	What it optimizes
Instruction tuning (step 1)	humans write ideal answers for sample prompts	imitate a demonstrated style
Reward model	humans rank several model answers; a second model learns the ranking	a cheap stand-in for human preference
RL on the reward model	the model's own outputs are scored and reinforced	whatever the reward model rewards
Verifiable reward	reward = the tests pass, or the answer checks out	behavior a program can grade

The last row is why coding improved fastest: a test suite is a reward function that never gets tired — and it is the same signal the benchmark on the next slides uses to grade.

SWE-bench: a pull request, cut in two

“SWE” is software engineering: 2,294 tasks harvested from 12 popular Python repositories



Instapoll

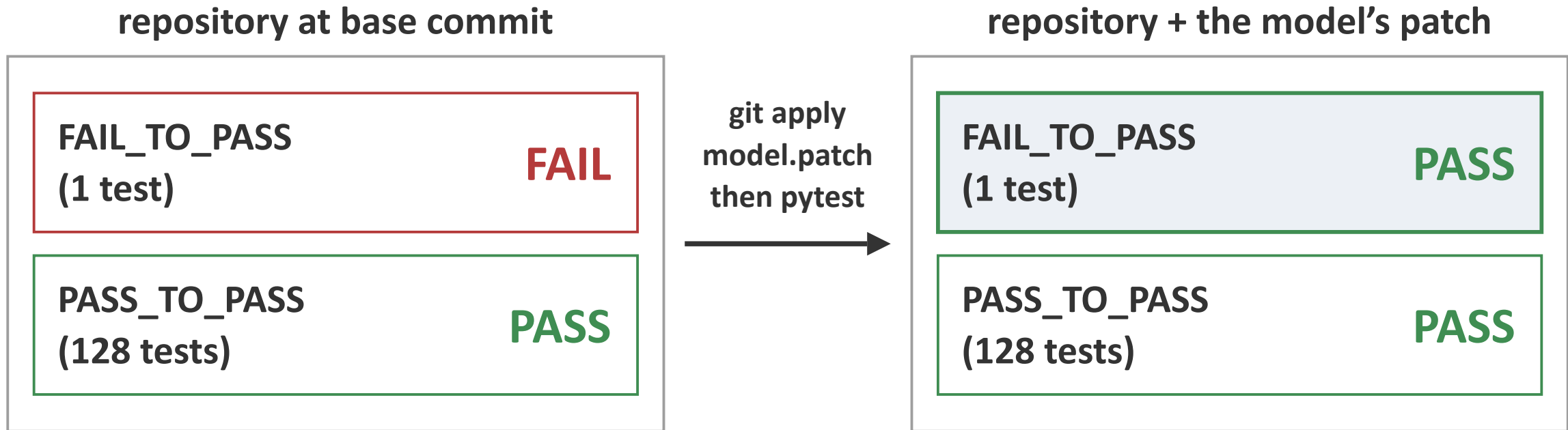
Instapoll · Review · Nov 17 B

A model's patch for a SWE-bench task makes the task's new test pass, and the reported bug is fixed. But one of the repository's 128 previously passing tests now fails. How does SWE-bench score the task?

- A. Resolved: the new test passing is what the task asked for
- B. Mostly resolved: credit is scaled by the fraction of tests that pass, 128 of 129
- C. Not resolved: every previously passing test must still pass, so the task scores zero
- D. Sent to a human reviewer, since the harness cannot tell a flaky test from a regression

“Resolved” is a two-sided test condition

Every FAIL_TO_PASS test must flip, and every PASS_TO_PASS test must still hold

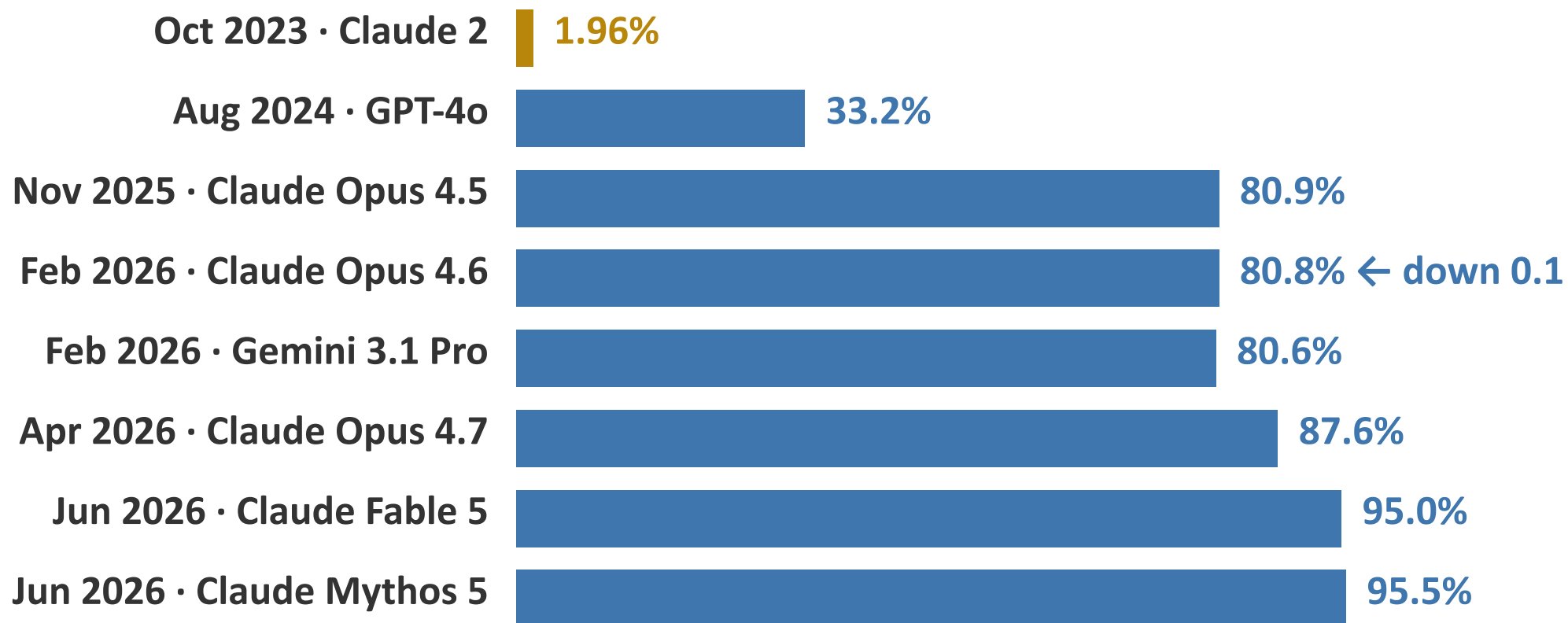


The shaded row is the one that must change. The plain row is the invariant: it is identical on both sides, and it is what a plausible but careless patch breaks.

One regression in PASS_TO_PASS scores zero. No partial credit, and no human reads the diff.

SWE-bench: 1.96% to 95.5% in three years

Each bar is a different harness, and the two colors are not the same test set



Amber is the original 2,294-issue test split; blue is the 500-issue Verified subset. Vendor-reported, different scaffolds — a trend line, not one experiment.

The number stopped measuring SWE quality

February 2026: SWE-bench Verified's own coauthors stopped reporting it

What a deeper re-review found	Detail
Tests too narrow — 49 problems	reject a functionally correct fix over an implementation detail the issue never stated
Tests too wide — 26 problems	demand behavior the problem statement never mentions
Contamination	the issues come from popular repositories that are also training data, so no canary string helps
Verbatim recall	frontier models reproduced the gold patch or the problem statement from the task id alone

GPT-5.2 solved problems believed unsolvable — its chain of thought recalled an argument name from a later version of the repository that the issue never mentioned.

Key Takeaways

1. Post-training's second step is reinforcement learning: the model's own outputs are scored and reinforced, against a reward model trained on human rankings or against a program that checks the answer. A test suite is a reward function that never gets tired, which is why coding improved fastest.
2. A SWE-bench task is a real merged pull request cut in two: the issue text and the pre-fix repository become the prompt, and the maintainer's tests become the grader. Resolved means every FAIL_TO_PASS test flips and every PASS_TO_PASS test still holds, with no partial credit.
3. SWE-bench measured real coding work for three years, until its issues, taken from popular public repositories, reached the training data and models began recalling the answers. Ask what a score can still distinguish before you trust it.