The wrong question

"Which agent is smarter?" compresses several different decisions into one. The model matters. So do the tools around it, the instructions it receives, the repository it enters, the checks it can run, and the cost of a plausible-looking mistake.

Our practical question is narrower: **which working setup gives this task the best chance of reaching an accepted result with the least avoidable effort?**

That shift matters because coding agents are not answer boxes. They inspect files, choose tools, change state, recover from failures, and decide when to stop. A useful comparison must evaluate that whole loop.

What we measured ourselves

In July 2026, we completed 294 decision-bearing Codex runs through ChatGPT subscription access. We compared GPT-5.6 Luna, Terra, and Sol across the common reasoning settings available in the runtime, then tested Terra and Sol Ultra in a separate appendix.

After removing two defective task families, the valid common policy view was 209 accepted results from 210 rows. Terra and Sol each passed 70 of 70. Luna passed 69 of 70. That is not a useful general model ranking. The tasks were mostly too easy to separate the models.

The more useful result concerned effort. Medium, high, extra high, and max each passed all 42 valid common rows. Medium used less time and fewer input tokens. On these bounded tasks, spending more reasoning did not buy a better accepted result.

Terra and Sol told the same story in the valid Ultra comparison. High, max, and Ultra each passed 14 of 14 matched tasks. Ultra took materially more time and tokens. We therefore treat higher effort as an escalation for a named risk, not as a default quality switch.

The test that taught us to distrust a clean score

The original responsive frontend task, PUB8, appeared to separate models and reasoning levels. Repeats made the pattern look unstable. A deeper audit found the real problem: the oracle could reject valid responsive layouts because it mistook padding offsets for columns and required one narrow DOM structure.

We quarantined all PUB8 model comparisons. We then built and calibrated a separately versioned PUB8V2 oracle, but did not run new measured rows because no real routing decision was blocked.

This is a business lesson as much as a testing lesson. A precise score can be wrong for a precise reason. The check must represent the outcome users care about. For interface work, that means deterministic desktop and mobile verification, not confidence in a single structural heuristic.

What our Claude comparisons can and cannot say

Our earlier subscription tests included narrow task contracts shared across Claude Code and Codex. They produced real separation, but only inside those boundaries.

Claude Sonnet at high and extra high passed the exact CE7 task, a read-only, secret-safe metadata inspection. It failed the exact CE9 helper task because it missed a whitespace-only validation case. Claude Sonnet high also passed four near-miss guardrails, correctly refusing to treat broader secret or backend work as equivalent to the narrow tasks.

Those results are useful routing evidence for the exact fixtures. They do not tell us that one product is generally better at security work, backend work, or software engineering. The model versions, subscription access, and product surfaces have also changed since May 2026.

What public users report

July and August discussions on Reddit and Hacker News are striking because they do not converge. Some users describe Codex as stronger for long-running repository implementation. Others prefer Claude Code for interactive work, review, or interface tasks. Some make the opposite split. Many use both.

The disagreements are informative. They suggest that repository shape, instructions, product ergonomics, quota conditions, and review habits change

the experience. They do not provide controlled proof of model quality. Reddit threads also inherit strong community selection effects, while plan limits and product behavior can change faster than an article can be updated.

Examples include a July Codex discussion about task fit and workflow, an August Claude Code discussion from dual subscribers, and a Hacker News comparison that repeatedly returns to specification quality. We treat these as reports from users, not benchmark rows.

A better way to choose

Start by writing the acceptance check before choosing the agent. What file or behavior must change? What must remain untouched? Which tests, screenshots, or artifacts would prove the result? What would make a plausible answer dangerous?

Then choose the lightest supported setup that can complete that bounded loop. For ordinary Codex work resembling our valid July tasks, medium is the practical starting hypothesis. For difficult evaluation or adversarial review, Sol high is a reasonable scout, not a proven winner. Max and Ultra belong behind an explicit effort question.

For work outside the measured boundary, run a small real trial. Keep the task fixed, verify it externally, and record the correction burden. Do not promote one successful session into a universal routing rule.

The decision in one minute

1. Define the task boundary and the failure you cannot afford.
2. Write deterministic acceptance checks before the run.
3. Start at medium when current exact evidence supports it.
4. Escalate effort only for a named difficulty or costly failure.
5. Treat community reports as hypotheses to test in your own environment.
6. Keep frontend selection open until desktop and mobile behavior are verified.

The useful outcome is not a winner. It is a repeatable way to spend less, review better, and know when the evidence no longer applies.

Evidence and limits

The controlled findings apply to Codex ChatGPT subscription access, the July 2026 runtime, the frozen tool profile, and the exact fixtures retained in our benchmark. The Claude comparisons apply only to the narrow May 2026 fixtures and the model lanes tested then. Public forum reports are self-selected and time-sensitive.

This article is a demand experiment as well as an evidence synthesis. Instavar currently has little search visibility for Codex or reasoning-level queries. That does not establish absent demand. It only means our existing pages have not earned meaningful visibility for those searches. We will judge this article by qualified discovery and reading progression after it receives enough exposure, not by a raw traffic target alone.