

Short answer: yes, on the commonly quoted standard API rates, GPT-5.2 costs about 40% more per token than GPT-5.1. The comparison is straightforward: the reported rates rise from \$1.25 to \$1.75 per million input tokens and from \$10 to \$14 per million output tokens. Each increase is 40%.

That does not mean every GPT-5.2 request will cost exactly 40% more. Your bill depends on the model variant, input and output token counts, cached-token pricing, and how much reasoning the task requires. Nor does a higher price by itself prove that GPT-5.2 is a better upgrade for your workload. Here is how to read the price claim, check the dates, and decide whether the change matters in daily use.

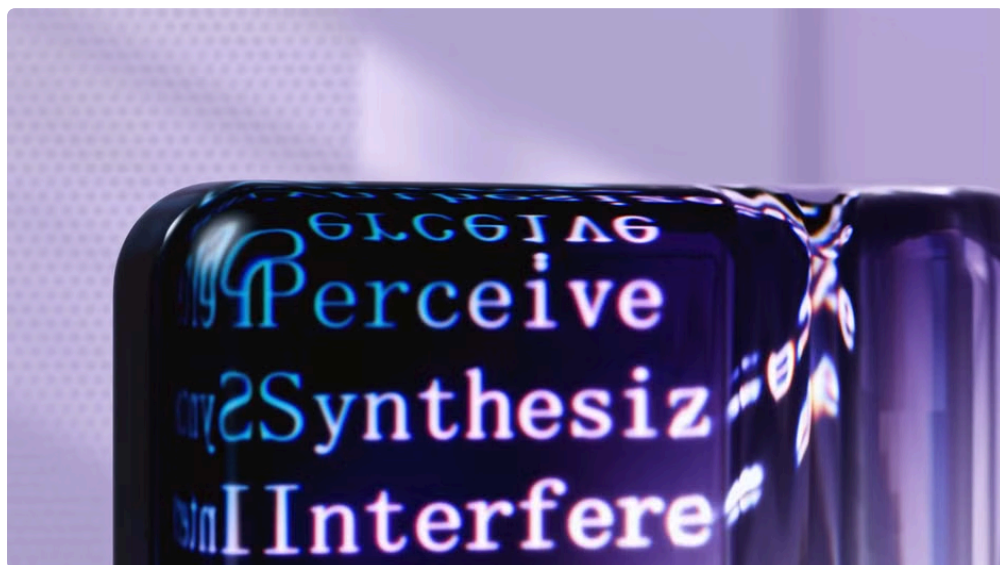
The GPT-5.2 cost comparison

	Model Input price per 1M tokens	Output price per 1M tokens	Comparison with GPT-5.1
GPT-5.1	\$1.25	\$10.00	
Baseline GPT-5.2	\$1.75	\$14.00	40% higher on both listed rates

These are standard per-token API rates in the price comparison discussed here, not a promise that every account, product, or model configuration is billed at those exact amounts. Before estimating a production bill, check OpenAI's current API pricing page for the specific model ID and any cached-input, batch, or other rate that applies to your use.

What "40% higher cost" means in practice

Take a request that uses 100,000 input tokens and 10,000 output tokens. At the listed GPT-5.1 rates, the token charges would be \$0.125 for input and \$0.10 for output, or \$0.225 total. At the GPT-5.2 rates, they would be \$0.175 and \$0.14, or \$0.315 total. That is 40% more for this particular token mix.



It is still only a token-cost example. If GPT-5.2 completes the task with fewer retries, less prompt scaffolding, or fewer generated tokens, the total cost difference could be smaller. If it produces longer answers or uses more reasoning tokens, the difference could be larger. Measure full task cost, not just the price per million tokens.

Where the price claim comes from

The "40% higher cost" wording should be treated as a **reported price claim**, not as a quality finding. A comparison attributed to Suprmind Data Lab, described as a secondary source, reports that GPT-5.2 is about 40%

more expensive than GPT-5.1. The arithmetic matches the listed \$1.25-to-\$1.75 input and \$10-to-\$14 output rate changes. But a secondary-source claim is not a substitute for checking OpenAI's live pricing documentation.



That distinction matters. A lab can calculate a percentage from published rates, but the result depends on which prices it used, when it collected them, and which model variants it compared. A careful price note should include the source, the date checked, the model IDs, and whether it compares input, output, or an assumed request mix. “GPT-5.2 costs 40% more” is useful shorthand; it is not a complete estimate for your application.

Announced date versus usable date

Model-release timelines often mix together several events: an announcement, access in a product, API availability, and broad access for all customers. Those dates are not interchangeable. A model can be announced on one day and become usable through a particular API or account tier later.

For GPT-5.1 and GPT-5.2, use the dates in OpenAI's release announcements to establish when each model was announced, then check the relevant API documentation or model page for when it became available to your account and in what form. Record both dates. If a price comparison says “at launch,” it should specify whether that means announcement day or the first day customers could actually send API requests.

This is a practical issue, not a footnote. Teams comparing invoices from different weeks may be comparing different access windows or price schedules. A model may also appear in a ChatGPT product before, after, or independently of its API release. A release-date table should therefore have separate columns for announcement and usable availability rather than one vague <https://suprmind.ai/hub/ai-models-index/> “launch date.”

Price is not preference: what blind-vote leaderboards can tell you

Launch benchmarks and blind preference votes answer different questions. A benchmark measures performance on a defined set of tasks under particular scoring rules. A blind-vote arena asks people to compare anonymous model responses and choose which they prefer. That can reveal perceived usefulness, clarity, or style. It does not directly measure API price, factual accuracy across every domain, or the model's reliability in your production workflow.

For preference comparisons, look at the LMArena (formerly LMSYS Chatbot Arena) text leaderboard with style control. Style control is worth noting because a response that sounds more polished or verbose may win a preference vote even when it is not more correct. Blind voting reduces the influence of model names, but it does not remove every source of bias: prompt selection, voter mix, response formatting, and sample size all matter.

The Hugging Face dataset `lmarena-ai/leaderboard-dataset` can help analysts inspect underlying leaderboard data and run their own comparisons. Check the dataset's documentation and snapshot date before drawing conclusions. Rankings can move as new votes arrive, and a leaderboard snapshot is not automatically a controlled GPT-5.1-versus-GPT-5.2 test using your prompts.

Compare the behavior you pay for

Before paying a 40% premium, define what "better" means for your use case. It could mean fewer factual errors, higher completion rates on tool calls, better coding fixes, lower latency, or less human review. A model can win blind preference votes and still regress on a narrow but business-critical workflow.

Run a small, paired evaluation. Give GPT-5.1 and GPT-5.2 the same representative prompts, tools, and constraints. Hide the model identity from reviewers. Score task success and error severity, then record token use, latency, retries, and review time. Include routine tasks as well as edge cases. Keep the prompts and scoring rules fixed so that a later model update can be compared against the same baseline.

Upgrade decisions: account for gains and regressions

A higher per-token price can still be worthwhile if the new model reduces other costs. For example, a model that completes a task in one call instead of two may offset some of its higher token rate. But do not assume that outcome from a launch announcement or a single leaderboard rank. Measure it.

Also test for regressions. A new model may improve complex reasoning while changing how it follows formatting instructions, handles refusals, calls tools, or responds to short prompts. Small style changes can break downstream parsers. A model that is more capable in general can still be less suitable for a specific prompt chain.

Use a staged rollout rather than switching every request at once:

1. **Set a baseline.** Save a representative prompt set, current success rates, token use, latency, and error examples from GPT-5.1.
2. **Run a paired test.** Compare both models on the same inputs and evaluate outputs without showing reviewers which model produced them.
3. **Calculate total cost.** Include input and output tokens, retries, tool calls, human review, and any extra application work needed to handle changed behavior.
4. **Check for regressions.** Review critical failure modes, not only average scores. One serious regression can outweigh small gains elsewhere.
5. **Roll out gradually.** Route a limited share of traffic to GPT-5.2, monitor quality and spend, and retain a fast rollback path.

Why release cadence across labs complicates comparisons

OpenAI is not the only lab shipping model updates. Anthropic, Google DeepMind, and other providers also release models and revise access or product offerings over time. That release cadence makes comparisons time-sensitive:

the model available today may not be the exact version tested in an earlier report, and a competitor's price or capability may change before your evaluation is complete.

Do not compare a newly released model's best launch benchmark with an older model's average production results. Use comparable dates, versions, task sets, and measurement methods. Keep a simple changelog with the announcement date, first usable date for your account, model identifier, price checked, and evaluation results. This is less glamorous than a leaderboard screenshot, but it is much more useful when a billing increase needs explaining.

Bottom line

GPT-5.2 is reported to cost about 40% more than GPT-5.1 at the cited standard API rates. The listed comparison is \$1.25 versus \$1.75 per million input tokens and \$10 versus \$14 per million output tokens. Treat the 40% higher cost figure as a price claim grounded in those rates, and verify the current prices and model IDs in OpenAI's documentation before making a budget decision.

The upgrade question is separate. Use announced dates and usable dates distinctly; treat blind-vote preference and launch benchmarks as different kinds of evidence; and test GPT-5.2 against GPT-5.1 on your own tasks, including regressions and total cost. A higher price may be justified—but only if the improvement shows up in the work your users actually need done.. Pretty simple.