Asking the Right Question: Epistemic Inquiry as a Learnable Reasoning Skill for Scientific Discovery

Anonymous Authors[1]

Abstract

Effective scientific reasoning depends not only on extending chains of thought, but on identifying what is missing, surfacing implicit assumptions and posing the right questions to unlock a solution. This epistemic dimension is central not just to discovery, but to scientific partnership: a genuine AI co-scientist must be able to articulate its knowledge gaps in a form that human experts can engage with, including gaps that can only be filled by tacit domain knowledge no model possesses. We introduce Chain-of-Questions (CoQ), a prompting and training framework in which a model explicitly generates targeted epistemic questions before attempting to solve a problem, producing a structured, inspectable representation of what it does not know. Across science, chemistry, mathematics, and coding benchmarks, CoQ consistently outperforms chain-of-thought prompting at matched token budgets, and question decompositions trained on one domain transfer to held-out chemistry targets without any chemistry training suggesting that the skill of decomposing uncertainty into questions is a portable reasoning skill, and not a benchmark artifact. Together, these results point toward a concrete capability that separates a scientific tool from a scientific collaborator: not the ability to answer, but the ability to ask.

1. Introduction

Recent progress in large language model (LLM) reasoning has been mostly driven forward by one design idea: explore more intermediate tokens before committing to an answer. Chain-of-thought prompting (Wei et al., 2022; Kojima et al., 2022) formalized the idea, self-consistency (Wang et al., 2023) and recent test-time-compute scaling (Snell et al., 2025) extended it through aggregation, and reinforcement-learning-tuned reasoning models such as DeepSeek-R1 (Guo et al., 2025) have pushed it to long, sparse-feedback trajectories. The implicit bet is that, given enough sampled derivations, a path to the right answer will emerge.

What this view leaves out is the prior question—*what would I need to know to solve this?*—and the act of seeking out the answer. A skilled human solver, faced with a hard problem, often does not produce a longer derivation; they identify the missing piece (a lemma, a definition, an algorithmic recipe, a possible decomposition) and inquire about it. We argue that a key bottleneck in current reasoning is not insufficient sampling of derivations, but insufficient identification of *what would help if it were known*.

Treating question-asking as a capability in its own right rather than a prompting trick matters in several ways. *Operationally*, the ability to identify what would help plausibly tracks the ability to solve the original problem; if so, training a model to ask should improve its solving on benchmarks where the model never sees questions at test time. We test this transfer claim directly (Section 5.2). *Epistemically*, recent work shows that LLMs are often poorly calibrated about what they know (Kadavath et al., 2022; Zhao et al., 2025), and that they routinely fail to flag what is missing rather than what is wrong (Fu et al., 2026); explicit epistemic questioning is a finer-grained interface than a single confidence score, supporting deferral, escalation, or expert query, and exposing ignorance in a form that downstream systems can act on. *Computationally*, sampling more chain-of-thought trajectories (Snell et al., 2025) buys variance, but is wasteful when the bottleneck is missing information rather than missing search; a single, well-targeted question, by contrast, can be answered locally by a cheap draft, escalated to a stronger tier (cf. speculative decoding (Leviathan et al., 2023; Chen et al., 2023)), delegated to a tool, or asked of a human expert. And in the longer run, the goal we want from AI is not just answer-producers but partners that, like skilled human researchers, can identify and pose the right questions—what has been called the *night science* of exploratory inquiry, distinct from the systematic testing that follows it (Yanai & Lercher, 2019). Co-scientist systems

[1]Anonymous Institution, Anonymous City, Anonymous Region, Anonymous Country. Correspondence to: Anonymous Author <anon.email@domain.com>.

Preliminary work. Submitted to the AI for Science workshop (ICML 2026). Do not distribute.

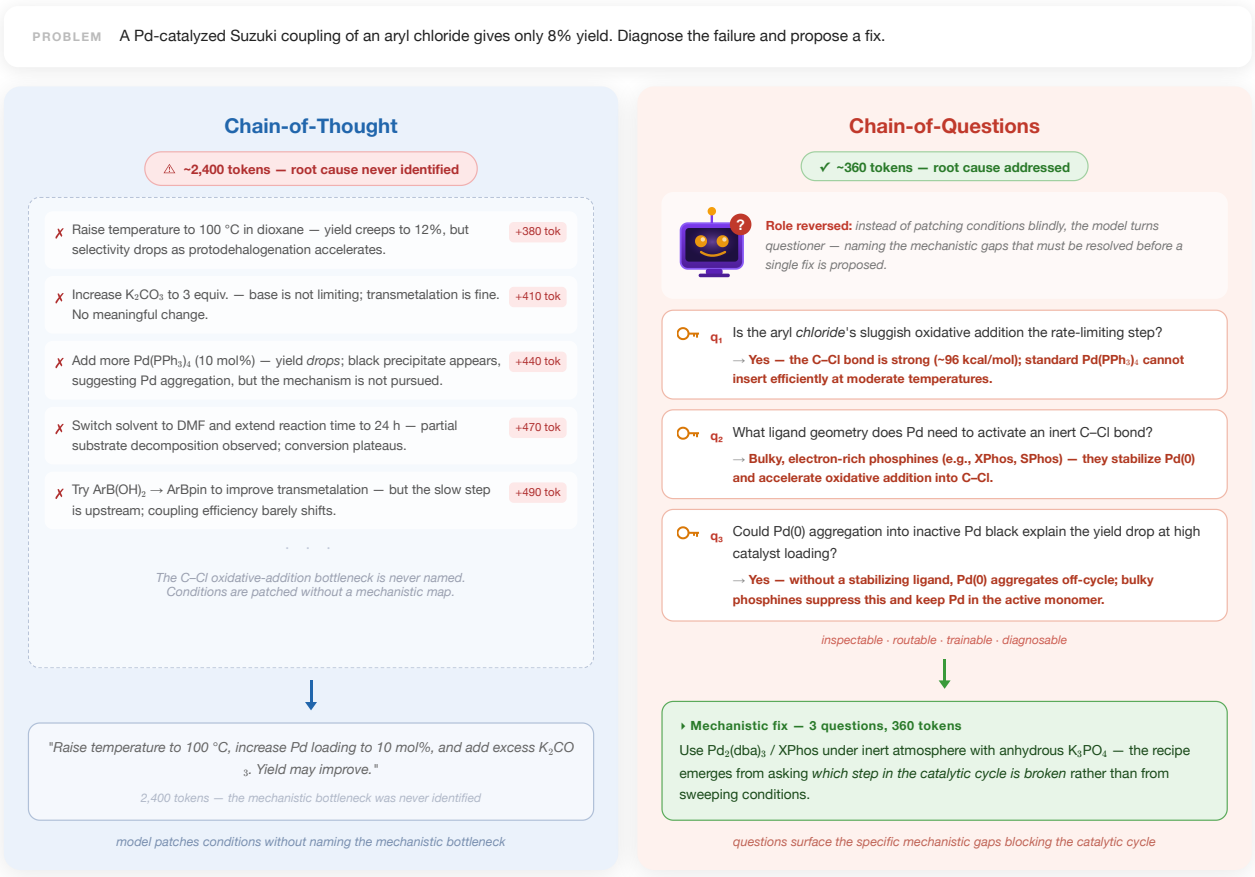

Figure 1. **Chain-of-Thought vs. Chain-of-Questions on a single problem.** CoT patches conditions — temperature, base, catalyst loading, solvent — expending tokens without identifying which elementary step is broken. CoQ first surfaces mechanistic gaps as a discrete, explicit list. Here, three targeted questions localize the failure to oxidative addition into the C–Cl bond and point to a bulky electron-rich phosphine (XPhos) as the fix. Thus, the solution is more likely to follow in a few targeted questions.

pursuing this kind of partnership (Bran et al., 2023; Lu et al., 2024; Romera-Paredes et al., 2024; Gottweis et al., 2025) will need an explicit, supervisable representation of the model's perceived knowledge gaps which is what epistemic questioning provides.

Some prior work treats question generation as an inference-time heuristic: Self-Ask uses sub-questions to bridge compositional fact retrieval (Press et al., 2023); Reflexion adds verbal self-critique (Shinn et al., 2023); ReAct interleaves thoughts with tool actions (Yao et al., 2023b). A separate line trains models to ask *clarifying* questions of users (Andukuri et al., 2024; Chi et al., 2024; Li et al., 2024; 2026; Zhang et al., 2025), recovering missing user input rather than missing knowledge. Concurrent work on stepping-stone question generation trains a single frontier model on AIME (Hu et al., 2026). Close in spirit, Ruan et al. (2025) train language models from *latent thoughts* treated as a latent variable to be inferred and bootstrapped from text. None of this prior work, however, treats question generation as a transferable reasoning capability supervised independently of final-task accuracy and evaluated across domains. That is the gap this paper sets out to fill.

We introduce *Chain of Questions* (CoQ): a prompting and training framework in which the model first poses and, optionally, answers a list of targeted *epistemic* questions whose answers would close the reasoning gap, before producing the final solution. The interface is a single artifact, the chain of epistemic questions, that is consumed in different ways depending on the experiment: as an inference-time scaffold; as supervised latent structure synthesized from a stronger teacher; as a transfer object, where a model trained to question on a source benchmark is evaluated on an unrelated target; and as a diagnostic object for mechanistic ablations and routing.

Contributions.

1. We propose CoQ as a prompting and training frame-

work in which a single artifact, a chain of epistemic questions, serves several distinct roles: inference-time scaffold, latent supervision signal, cross-domain transfer object, and routing unit. We provide a latent-variable formalism connecting this interface to chain-of-thought sampling and to recent latent-thought reasoning (Ruan et al., 2025) (Sections 3, 5.2).

2. *Inference-time evidence.* Across mathematics, science, and coding benchmarks, CoQ prompting improves over matched CoT prompting on a broad model matrix, with the most reliable gains where the task bottleneck is identifying missing information rather than executing a known procedure (Section 5.1).

3. *Training-time evidence.* The question list serves as a latent supervision signal: a model trained on teacher-synthesized question decompositions for one benchmark improves on unrelated target benchmarks under standard prompting (+11.02 pp, +8.75 pp). A complementary curriculum trains the model to ask the question that would best repair a masked step in a partial solution, and an adaptive-trained questioner improves a fixed CoQ scaffold by +8.33 pp over the base questioner (Section 5.2).

4. *Mechanistic evidence.* An oracle-answer ablation separates the contribution of asking, of question quality, and of answer value. Separately, a question-level cascade router shows that routing over explicit subquestions reaches the same accuracy as routing over CoT steps while escalating to the strong tier *40% less often*, making the question list a practical unit of cost-aware compute allocation (Section 5.3).

The remainder of the paper situates the work against prior and concurrent question-asking research (Section 2), formalizes the CoQ interface (Section 3), states the experimental setup and the three lines of evidence that organize the study (Section 4), reports the inference-time, training-time, and mechanistic results (Section 5), and discusses what the empirical picture supports (Section 6).

2. Related Work

A full per-cluster discussion and side-by-side comparison matrix are in Appendix A; we note the main contrasts here.

CoQ differs from chain-of-thought (Wei et al., 2022; Kojima et al., 2022), self-consistency (Wang et al., 2023), and test-time-compute scaling (Snell et al., 2025; Guo et al., 2025) in that it samples *decompositions* rather than derivations, storing them as a representation that can be consumed by a solver, used as training data, or routed across tiers. Inference-time question-asking methods such as Self-Ask (Press et al., 2023), ReAct (Yao et al., 2023b), and Reflexion (Shinn et al., 2023) treat questions as heuristics supervised only by final-task outcomes; we *train* on questions, judge quality independently of outcomes, and evaluate cross-benchmark transfer. A separate line trains models to ask *clarifying* questions of users (Andukuri et al., 2024; Chi et al., 2024; Li et al., 2024; Zhang et al., 2025); epistemic questions target the model's own knowledge gaps, use different supervision signals, and transfer across benchmarks. The closest concurrent work, ARQ (Hu et al., 2026), trains stepping-stone question generation on AIME with a single frontier model; it differs from CoQ in scope (single benchmark, single model class), transfer design, and mechanistic decomposition (Appendix A.5). Speculative decoding (Leviathan et al., 2023; Chen et al., 2023) and calibration work (Kadavath et al., 2022; Zhao et al., 2025; Fu et al., 2026) each address one dimension of what CoQ unifies; none treats question-asking as a supervisable, transferable representation.

Co-scientist systems and the epistemic gap. A growing line of work builds toward AI as a scientific partner: ChemCrow (Bran et al., 2023) augments LLMs with chemistry tools, the AI Scientist (Lu et al., 2024) targets end-to-end automated research, FunSearch (Romera-Paredes et al., 2024) couples LLM proposers with verifiers for mathematical discovery, and Google's co-scientist project (Gottweis et al., 2025) pursues multi-agent collaboration with human scientists. A consistent gap across these systems is that question-asking appears only as a control-flow primitive—"ask the tool what the boiling point is", "ask the user to confirm a parameter"—rather than as a *trainable, supervisable* representation. A co-scientist that cannot identify and articulate a specific knowledge gap cannot extract tacit expert knowledge, cannot prioritize follow-up experiments, and cannot expose its reasoning to expert review. CoQ is designed to fill exactly this role: it provides the explicit, judged representation of epistemic gaps that co-scientist architectures will need if they are to move from tool to genuine collaborator.

3. Chain-of-Questions Prompting

Let x be a problem and let M be a language model. A standard CoT solver samples a reasoning trace $r \sim M(\cdot \mid x)$ and extracts an answer y from r. Under CoQ, the model is first asked to write a list of *epistemic* questions $q_{1:m} \sim M(\cdot \mid x, \pi_{\text{ask}})$—specific, gap-targeting queries about facts, lemmas, or subskills whose answers would help close the reasoning gap, not clarifying queries directed at a user. The same model then answers each question and produces the final solution; example π_{ask} templates used across our benchmarks are given in Appendix B.

The default mode is therefore self-contained at inference: a

single model asks, answers, and solves. Two modifications appear in subsequent sections of the paper. In some diagnostic conditions (Section 5.3), the answers $a_{1:m}$ instead come from a stronger oracle or tool, or are withheld so that only the question structure is tested. At training time, question traces from a stronger teacher serve as supervised latent structure: a student is fine-tuned to imitate the teacher's questioning behavior on one source domain, and is then evaluated on unrelated target benchmarks (Section 5.2).

Relation to other decomposition methods. CoQ is not a new search algorithm or action ontology. Least-to-most prompting commits to an ordered subproblem decomposition (Zhou et al., 2023); Tree-of-Thoughts commits to a search over intermediate states (Yao et al., 2023a); ReAct interleaves natural-language thoughts with tool actions (Yao et al., 2023b). CoQ is more minimal: the model writes down the questions it believes would help, and the rest of the experimental design decides what is done with them. That minimality is what lets one questioning protocol serve the various roles laid out in Section 1—from inference-time prompting to supervised training data—rather than being tied to a single inference strategy.

4. When, Why, and at What Cost

Our central question is when, why, and at what cost epistemic question-asking helps. Answering it requires three complementary lines of evidence, which form the structural backbone of the paper:

1. **Inference-time evidence** (Section 5.1). A matched-prompt benchmark matrix asks whether CoQ improves on CoT across domains and model scales, with token-normalized comparisons that control for completion-token cost.
2. **Question-training evidence** (Section 5.2). Cross-domain latent-question training and adaptive questioner training test whether question-asking is a benchmark-bound prompt artifact or a transferable, trainable reasoning capability.
3. **Mechanistic evidence** (Section 5.3). The oracle-answer ablation and learned per-question routing decompose any CoQ gain into structural benefit, question quality, answer value, and routing benefit.

The remainder of this section states the protocol shared across all three. For each benchmark we run matched CoT and CoQ families: the CoT side contains a direct chain-of-thought prompt and small variants; the CoQ side contains list-style question generation and a stronger question-and-answer prompt. For coding benchmarks we additionally maintain a direct-control lane using benchmark-native prompting, so the structured-prompt comparison is not conflated with public-leaderboard anchors.

We evaluate greedy and sampled decoding. For sampled runs the primary metric is pass@k. Because CoQ traces can be longer than CoT traces, we also report a completion-token-normalized pass@k: if CoQ costs R times as many output tokens per sample, we compare CoQ pass@k to CoT pass@$\lfloor R\,k \rfloor$ at approximately equal completion-token budget. Raw pass@k can confuse more reasoning with more budget; the equal-token view answers a sharper question. We refer to the largest k at which both sides have a paired data point under this normalization as the *equal-token endpoint*.

Each benchmark is run under three CoT and three CoQ prompt variants (Appendix B); results are reported as *family means*, pass@k averaged across the three variants, to reduce sensitivity to individual prompt phrasing.

5. Experiments

We organize the experiments along the three lines of evidence introduced in Section 4: an inference-time comparison (§5.1), latent-question transfer (§5.2), and a mechanistic decomposition with learned per-question routing (§5.3).

5.1. Inference-Time Evidence: CoQ vs. CoT

The benchmark suite spans mathematics (MATH (Hendrycks et al., 2021), AIME), science (GPQA-Diamond (Rein et al., 2024)), chemistry (ChemIQ (Runcie et al., 2026)), and code (HumanEval (Chen et al., 2021), MBPP (Austin et al., 2021), LiveCodeBench (Jain et al., 2025)). The model matrix covers open instruction models from 7B to 70B (Qwen, Llama, Phi, Granite, Mistral), reasoning-tuned models (DeepSeek-R1 distillations (Guo et al., 2025), QwQ-32B), and frontier closed models (Gemini Flash-Lite, GPT-4o mini) accessed through APIs.

Across the matched-prompt matrix, CoQ is not a uniform improvement over CoT: depending on the benchmark, model family, and decoding budget, the family-mean delta ranges from clearly positive to slightly negative. We therefore report by domain, decoding budget, and model family rather than aggregating into a single number. Figure 2 shows the selected token-normalized slices that carry the main empirical story; the full matrix is retained as a numerical reference in Table 3. The rest of this subsection summarizes the per-domain reads, with detail in Appendices C.5–C.11.

Science (GPQA-Diamond). On GPQA-Diamond, the gain under CoQ is concentrated in the low-token regime: Granite-3.3-8B, Llama-3.1-8B, Phi-4-mini, Mistral-7B, and Qwen2-7B all improve in the first paired token-normalized comparison (Figure 2 and Appendix C.7). The remaining

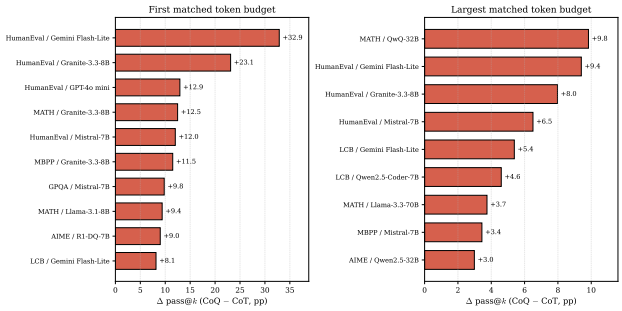

Figure 2. **Selected token-normalized CoQ gains.** The left panel reports the first paired equal-completion-token budget; the right panel reports the largest paired equal-completion-token budget. Values are percentage-point deltas, CoQ minus CoT.

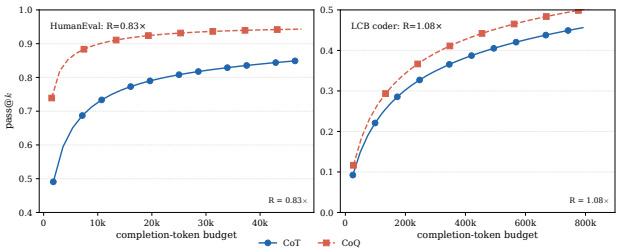

Figure 3. Why token-normalized pass@k is part of the main protocol. Left: Gemini Flash-Lite on HumanEval is a favorable low-token case. Right: Qwen2.5-Coder-7B on LiveCodeBench shows a coding success that persists across paired completion-token budgets.

models are either neutral or close to the benchmark ceiling.

Math (MATH, AIME). On MATH, Llama-3.3-70B improves under CoQ on both raw and token-normalized pass@k. For Granite-3.3-8B and Llama-3.1-8B the gain is not visible in the raw equal-sample view but emerges under token normalization, particularly at low sampling budgets, where the additional CoQ verbosity is small enough to fit an extra sampled attempt within the matched CoT budget. Reasoning models (DeepSeek-R1-Distill-Qwen-7B and -32B, QwQ-32B) contribute their clearest gains in low-sampling or low-token regimes rather than in the tail. On AIME, the token-normalized improvements concentrate on Gemini Flash-Lite, Qwen2.5-32B, and R1-Distill-Qwen-7B (Appendices C.5 and C.6).

Code (HumanEval, MBPP, LiveCodeBench). On HumanEval, CoQ improves over CoT in the low-token regime: Gemini Flash-Lite is the strongest case (pass@1 $0.49 \rightarrow 0.74$), with five further models positive (Figure 2 and Appendix C.8); full numbers in Table 5. LiveCodeBench shows a more selective version of the same pattern: Gemini Flash-Lite, Qwen2.5-Coder-7B, Granite-3.3-8B, and Mistral-7B are favorable under token-normalized budgets, while Llama-3.3-70B goes the other way and we treat it as an informative counterexample. MBPP is mixed but gives two wins (Granite-3.3-8B and Mistral-7B) plus a low-budget reasoning win for R1-Distill-Qwen-7B. Per-benchmark details and the remaining results are reported in Appendices C.8–C.10.

5.2. Epistemic Questions Transfer Across Scientific Domains

A model trained to ask useful questions about mathematics problems can improve reasoning on chemistry benchmarks it has never seen—without any chemistry training data. This cross-domain transfer is the central empirical claim of this section, and it is what distinguishes epistemic question-asking from ordinary in-context learning or domain-specific fine-tuning.

Table 1. **Cross-domain latent-question transfer.** A Qwen2-7B model fine-tuned on question traces from a source domain is evaluated on held-out chemistry targets under standard prompting. Chemistry was chosen as the target precisely because it shares no surface similarity with either source.

Transfer (source → target)	Best CoT	Best CoQ	Δ
AIME → ChemBench	46.61%	57.63%	**+11.02 pp**
GPQA-Diamond → ChemIQ	39.38%	48.12%	**+8.75 pp**

Table 1 and Figure 4 show the result. A Qwen2-7B model trained on mathematics question traces (AIME) improves chemistry reasoning (ChemBench) by $+11.02$ percentage points over the best matched CoT strategy on the same checkpoint. The GPQA-to-ChemIQ pair reproduces the direction at $+8.75$ points. The two pairs share neither source domain, target domain, nor dataset construction. What transfers is the asker itself—the ability to decompose uncertainty into targeted questions on a domain the model has never been trained on.

Why this matters for scientific AI. The gap between a tool and a scientific co-author is not purely a matter of domain knowledge—it is a matter of knowing what to ask. A system that has internalized the skill of surfacing its own knowledge gaps can apply that habit to new domains, just as a skilled scientist brings their question-forming instincts to an unfamiliar field. Our transfer result provides a concrete, measurable instance of this: the questioning behavior learned from one domain is a portable capability, not a benchmark-specific artifact.

How the transfer works. We treat the question list q as a latent variable in the joint $p_\theta(y, q \mid x) = p_\theta(q \mid x)\, p_\theta(y \mid x, q)$, training both factors on teacher-synthesized $q^\star$ obtained from solved source-task pairs:

$$\mathcal{L}_{\text{lat}}(\theta) \;=\; -\,\mathbb{E}_{(x,q^\star,y^\star)\sim D}\big[\log p_\theta(q^\star \mid x) + \log p_\theta(y^\star \mid x, q^\star)\big]\,. \tag{1}$$

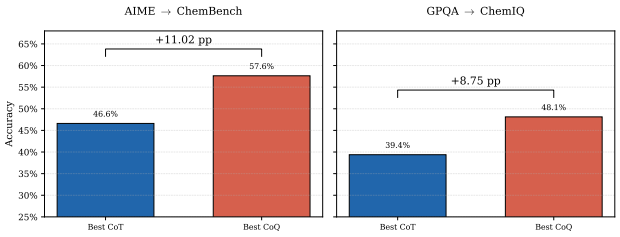

Figure 4. **Latent-question transfer.** Each panel shows the best predeclared CoT variant against the best predeclared CoQ variant for the same transferred checkpoint. The individual v1–v3 strategy rows appear in Table 6.

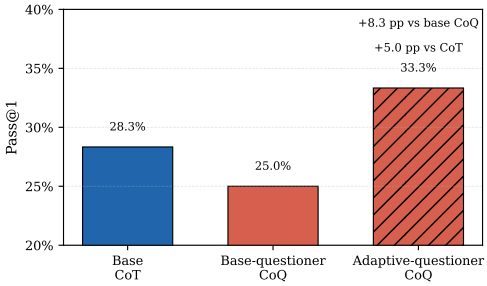

Figure 5. **Adaptive questioner training on ChemBench.** Solver held fixed; only the questioner changes. An adaptively trained questioner improves over the base questioner by 8.33 pp and over base CoT by 5.00 pp.

The first term trains the asker $p_\theta(q \mid x)$; the second trains the solver to use a high-quality decomposition when one is provided. At test time, the student generates questions autonomously on the target domain. Appendix C.12 gives the full formal connection to latent-thought reasoning (Ruan et al., 2025).

Adaptive questioner training. The transfer results above use a questioner trained by imitation of a teacher. We also test whether a model can be trained to ask *better* questions through a gap-targeting curriculum: the model is shown a partial solution with masked steps and must generate questions that would recover the missing reasoning (Appendix D). In a ChemBench pilot with a fixed solver ($n = 60$), replacing the base Qwen2-7B questioner with the adaptively trained one raises pass@1 from 25.00% to 33.33% (+8.33 pp) under the same two-question scaffold, and exceeds base CoT (28.33%) by 5.00 points (Figure 5). This confirms that question-asking quality is trainable independent of final-answer supervision—a prerequisite for using it as a supervisable co-scientist capability.

5.3. Mechanistic Decomposition and Routing

The mechanistic line of evidence asks where any CoQ gain actually comes from—asking, filtering, answering, or escalation—and probes the deployability of the interface as a cost-aware reasoning substrate.

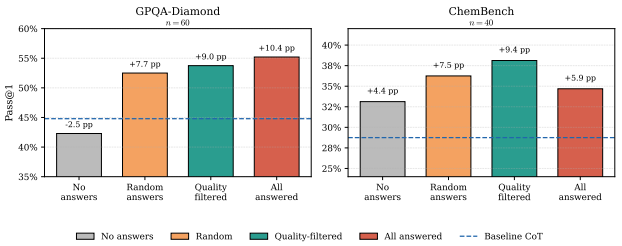

Figure 6. **Oracle-answer ablation.** Each panel shows the four question-answer treatments requested by the ablation—no answers, random answers, quality-filtered answers, and all answers—with the baseline CoT condition shown as a dashed reference line. GPQA-Diamond favors answering all questions; ChemBench favors filtering to higher-quality questions.

Oracle-answer ablation. We run an inference-only ablation with five conditions on the same prompt family. The solver generates questions; an oracle-quality model judges and answers them; the final solver then receives (a) all answers, (b) only quality-filtered answers, (c) a random matched subset of answers, (d) no answers (questions only), or (e) no question phase at all. The five conditions cleanly disentangle three effects: structure without answers (d vs. e), question quality (b vs. c), and answer value (a vs. d). The ablation is designed to distinguish "asking helped" from "a stronger model gave away useful facts."

Figure 6 shows the two settings in which the ablation is currently most informative. On GPQA-Diamond, giving oracle answers to all generated questions raises pass@1 from the baseline CoT value of 44.79% to 55.21% (+10.42 points), while quality-filtered answers reach 53.75%. ChemBench gives a complementary pattern: the quality-filtered condition is the best treatment, reaching 38.12% versus a 28.75% baseline (+9.38 points), while answering every question is weaker. The mechanistic read is the same on both: answers help, but *which* questions are answered matters.

Question-level routing. One of the motivations for explicit questions in Section 1 is that they are the right unit of compute allocation: each entry in a CoQ trace is short, self-contained, and tied to a specific knowledge gap, which means it can in principle be priced individually and answered by the smallest model that can answer it correctly. We test this operationalization here, treating each CoQ entry (or each CoT reasoning step) as a unit that may be answered locally or escalated to a stronger tier.

We compare three policies on GPQA-Diamond: *all-draft* (cheap model answers every unit), *all-frontier* (strong model answers every unit), and *confidence routing* (draft model answers, escalating units whose log-probability falls below a threshold). The same three policies are applied to both

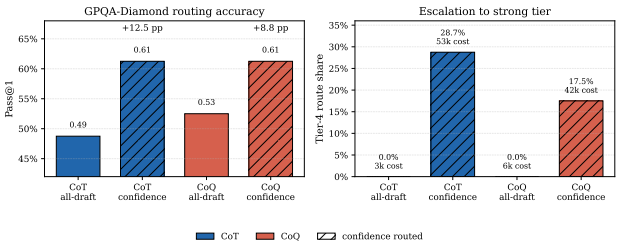

Figure 7. **Question-level routing on GPQA-Diamond.** Left: confidence routing improves pass@1 for both CoT and CoQ relative to their all-draft baselines, and reaches the same 61.25% on both sides. Right: at that same accuracy, CoT escalates 28.75% of units to the strong tier while CoQ escalates only 17.5%. Equal accuracy at lower escalation cost means a CoQ unit is, on average, a more selective query.

CoT and CoQ units.

Confidence routing reaches the same accuracy on both sides (61.25% pass@1, Figure 7, left): it improves CoT by $+12.50$ points over its all-draft baseline of 48.75% and improves CoQ by $+8.75$ points over its all-draft baseline of 52.50%. The cost picture, however, is asymmetric. Reaching that same 61.25% accuracy requires escalating 28.75% of CoT units to the strong tier, but only 17.5% of CoQ units (Figure 7, right). At equal accuracy, the question list compresses where escalation is needed: a CoQ unit is more often a self-contained query whose draft answer is reliably either correct or calibrated-low, so the policy escalates more selectively. The practical implication is that, on this benchmark, the same answer quality can be obtained for roughly 60% of the strong-tier cost when the unit of routing is a question rather than a reasoning step.

6. Discussion

What the empirical picture lets us claim. The strongest defensible reading of our results is not "questions beat thoughts." It is sharper: explicit questions make reasoning traces *actionable*. They give us handles for filtering, routing, latent training, external answering, and failure analysis. When those handles align with the task bottleneck—identifying missing information rather than executing a known procedure—the interface earns its cost; when they do not, it is overhead. This is why we report by domain, decoding budget, and model family rather than under a single aggregate.

Potential for AI co-scientists. A tool answers the question it is given. A co-author identifies what question should be asked. Our evidence suggests epistemic questioning is a concrete, measurable step toward the second category: a system that has internalized the habit of decomposing uncertainty into explicit questions can surface knowledge gaps in domains it was never trained on, direct compute toward the gaps that matter, and expose its reasoning in a form that human experts can engage with. This is not the full co-scientist vision, but it is a prerequisite for it.

Falsifiable claims. The framing makes our negative results legible. If CoQ underperforms CoT after token normalization, the questions are not yet worth their cost for that domain or prompt. If the oracle-answer ablation improves only when answers are supplied, the value lies in external knowledge access rather than question structure. If latent-question training helps only on source-like targets, the learned questions are benchmark-specific rather than portable. These distinctions are hard to draw with ordinary scratchpads; making them tractable is what an interface paper should deliver.

7. Limitations

Question quality is generator-dependent: a weak questioner can make CoQ look bad even when the downstream architecture is sound, and we address this only partially through the oracle-answer ablation. Token-normalization is essential rather than optional, so many of our claims are gated on cost-controlled views that more permissive readings of the literature do not require. Finally, we deliberately exclude RL post-training; our latent-question experiments use supervised teacher-synthesized traces only, which is a scope choice rather than a claim that RL is unhelpful (Hu et al., 2026).

8. Conclusion

We observed that explicit question-asking is a learnable, transferable, and cost-measurable dimension of LLM reasoning — one that the field has largely left unsupervised. The evidence for this comes from three directions that reinforce each other: CoQ improves over CoT at inference time where the task bottleneck is identifying missing information rather than executing a known procedure; question decompositions trained on one domain transfer to unrelated targets under standard prompting; and routing over explicit subquestions reaches the same accuracy as routing over chain-of-thought steps at roughly 60% of the strong-tier escalation cost. None of these results requires questions to be uniformly better than thoughts. What they require is only that questions are a distinct interface — one that makes reasoning traces filterable, trainable, and routable in ways that unstructured scratchpads are not. We hope this work contributes one concrete building block toward AI systems that are productive epistemic partners in scientific inquiry, systems that, like skilled researchers, know not just how to answer, but what to ask.

References

Andukuri, C., Fränken, J.-P., Gerstenberg, T., and Goodman, N. STaR-GATE: Teaching language models to ask clarifying questions. In *First Conference on Language Modeling*, 2024. URL https://openreview.net/forum?id=CrzAj0kZjR.

Austin, J., Odena, A., Nye, M., Bosma, M., Michalewski, H., Dohan, D., Jiang, E., Cai, C., Terry, M., Le, Q., and Sutton, C. Program synthesis with large language models, 2021. URL https://arxiv.org/abs/2108.07732.

Bai, Y., Kadavath, S., Kundu, S., Askell, A., Kernion, J., Jones, A., Chen, A., Goldie, A., Mirhoseini, A., McKinnon, C., Chen, C., Olsson, C., Olah, C., Hernandez, D., Drain, D., Ganguli, D., Li, D., Tran-Johnson, E., Perez, E., Kerr, J., Mueller, J., Ladish, J., Landau, J., Ndousse, K., Lukosuite, K., Lovitt, L., Sellitto, M., Elhage, N., Schiefer, N., Mercado, N., DasSarma, N., Lasenby, R., Larson, R., Ringer, S., Johnston, S., Kravec, S., El Showk, S., Fort, S., Lanham, T., Telleen-Lawton, T., Conerly, T., Henighan, T., Hume, T., Bowman, S. R., Hatfield-Dodds, Z., Mann, B., Amodei, D., Joseph, N., McCandlish, S., Brown, T., and Kaplan, J. Constitutional AI: Harmlessness from AI feedback, 2022. URL https://arxiv.org/abs/2212.08073.

Bran, A. M., Cox, S., Schilter, O., Baldassari, C., White, A., and Schwaller, P. Augmenting large language models with chemistry tools. In *NeurIPS 2023 AI for Science Workshop*, 2023. URL https://openreview.net/forum?id=wdGIL6lx3l.

Castagna, F., Sassoon, I., and Parsons, S. Critical-questions-of-thought: Steering LLM reasoning with argumentative querying, 2024. URL https://arxiv.org/abs/2412.15177.

Chen, C., Borgeaud, S., Irving, G., Lespiau, J.-B., Sifre, L., and Jumper, J. Accelerating large language model decoding with speculative sampling, 2023. URL https://arxiv.org/abs/2302.01318.

Chen, M., Tworek, J., Jun, H., Yuan, Q., Pinto, H. P. d. O., Kaplan, J., Edwards, H., Burda, Y., Joseph, N., Brockman, G., Ray, A., Puri, R., Krueger, G., Petrov, M., Khlaaf, H., Sastry, G., Mishkin, P., Chan, B., Gray, S., Ryder, N., Pavlov, M., Power, A., Kaiser, L., Bavarian, M., Winter, C., Tillet, P., Such, F. P., Cummings, D., Plappert, M., Chantzis, F., Barnes, E., Herbert-Voss, A., Guss, W. H., Nichol, A., Paino, A., Tezak, N., Tang, J., Babuschkin, I., Balaji, S., Jain, S., Saunders, W., Hesse, C., Carr, A. N., Leike, J., Achiam, J., Misra, V., Morikawa, E., Radford, A., Knight, M., Brundage, M., Murati, M., Mayer, K., Welinder, P., McGrew, B., Amodei, D., McCandlish, S., Sutskever, I., and Zaremba, W. Evaluating large language models trained on code, 2021. URL https://arxiv.org/abs/2107.03374.

Chi, Y., Lin, J., Lin, K., and Klein, D. Clarinet: Augmenting language models to ask clarification questions for retrieval, 2024. URL https://arxiv.org/abs/2405.15784.

Fu, H. Y., Shrivastava, A., Moore, J., West, P., Tan, C., and Holtzman, A. AbsenceBench: Language models can't tell what's missing. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems Datasets and Benchmarks Track*, 2026. URL https://openreview.net/forum?id=pmLMrqhIpb.

Glaese, A., McAleese, N., Trębacz, M., Aslanides, J., Firoiu, V., Ewalds, T., Rauh, M., Weidinger, L., Chadwick, M., Thacker, P., Campbell-Gillingham, L., Uesato, J., Huang, P.-S., Comanescu, R., Yang, F., See, A., Dathathri, S., Greig, R., Chen, C., Fritz, D., Elias, J. S., Green, R., Mokrá, S., Fernando, N., Wu, B., Foley, R., Young, S., Gabriel, I., Isaac, W., Mellor, J., Hassabis, D., Kavukcuoglu, K., Hendricks, L. A., and Irving, G. Improving alignment of dialogue agents via targeted human judgements, 2022. URL https://arxiv.org/abs/2209.14375.

Gottweis, J., Weng, W.-H., Daryin, A., Tu, T., Palepu, A., Sirkovic, P., Myaskovsky, A., Weissenberger, F., Rong, K., Tanno, R., Saab, K., Popovici, D., Blum, J., Zhang, F., Chou, K., Hassidim, A., Gokturk, B., Vahdat, A., Kohli, P., Matias, Y., Carroll, A., Kulkarni, K., Tomasev, N., Guan, Y., Dhillon, V., Vaishnav, E. D., Lee, B., Costa, T. R. D., Penadés, J. R., Peltz, G., Xu, Y., Pawlosky, A., Karthikesalingam, A., and Natarajan, V. Towards an AI co-scientist, 2025. URL https://arxiv.org/abs/2502.18864.

Guo, D., Yang, D., Zhang, H., Song, J., Wang, P., Zhu, Q., Xu, R., Zhang, R., Ma, S., Bi, X., Zhang, X., Yu, X., Wu, Y., Wu, Z. F., Gou, Z., Shao, Z., Li, Z., Gao, Z., et al. DeepSeek-R1 incentivizes reasoning in LLMs through reinforcement learning. *Nature*, 645(8081):633–638, September 2025. ISSN 1476-4687. doi: 10.1038/s41586-025-09422-z. URL http://dx.doi.org/10.1038/s41586-025-09422-z.

Hendrycks, D., Burns, C., Kadavath, S., Arora, A., Basart, S., Tang, E., Song, D., and Steinhardt, J. Measuring mathematical problem solving with the MATH dataset. In *Thirty-fifth Conference on Neural Information Processing Systems Datasets and Benchmarks Track (Round 2)*, 2021. URL https://openreview.net/forum?id=7Bywt2mQsCe.

Hu, H., Fu, T., Jiang, M., Miller, A. H., Bachrach, Y., and Foerster, J. N. Asking the right questions: Improving reasoning with generated stepping stones, 2026. URL https://arxiv.org/abs/2602.19069.

Jain, N., Han, K., Gu, A., Li, W.-D., Yan, F., Zhang, T., Wang, S., Solar-Lezama, A., Sen, K., and Stoica, I. LiveCodeBench: Holistic and contamination free evaluation of large language models for code. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=chfJJYC3iL.

Kadavath, S., Conerly, T., Askell, A., Henighan, T., Drain, D., Perez, E., Schiefer, N., Hatfield-Dodds, Z., DasSarma, N., Tran-Johnson, E., Johnston, S., El-Showk, S., Jones, A., Elhage, N., Hume, T., Chen, A., Bai, Y., Bowman, S., Fort, S., Ganguli, D., Hernandez, D., Jacobson, J., Kernion, J., Kravec, S., Lovitt, L., Ndousse, K., Olsson, C., Ringer, S., Amodei, D., Brown, T., Clark, J., Joseph, N., Mann, B., McCandlish, S., Olah, C., and Kaplan, J. Language models (mostly) know what they know, 2022. URL https://arxiv.org/abs/2207.05221.

Kojima, T., Gu, S. S., Reid, M., Matsuo, Y., and Iwasawa, Y. Large language models are zero-shot reasoners. In Oh, A. H., Agarwal, A., Belgrave, D., and Cho, K. (eds.), *Advances in Neural Information Processing Systems*, 2022. URL https://openreview.net/forum?id=e2TBb5y0yFf.

Leviathan, Y., Kalman, M., and Matias, Y. Fast inference from transformers via speculative decoding, 2023. URL https://arxiv.org/abs/2211.17192.

Li, B. Z., Kim, B., and Wang, Z. QuestBench: Can LLMs ask the right question to acquire information in reasoning tasks? In *The Thirty-ninth Annual Conference on Neural Information Processing Systems Datasets and Benchmarks Track*, 2026. URL https://openreview.net/forum?id=gpwA9aZLTZ.

Li, G., Hammoud, H. A. A. K., Itani, H., Khizbullin, D., and Ghanem, B. CAMEL: Communicative agents for "mind" exploration of large language model society. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023. URL https://openreview.net/forum?id=3IyL2XWDkG.

Li, S. S., Balachandran, V., Feng, S., Ilgen, J. S., Pierson, E., Koh, P. W., and Tsvetkov, Y. MediQ: Question-asking LLMs and a benchmark for reliable interactive clinical reasoning. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*, 2024. URL https://openreview.net/forum?id=W4pIBQ7bAI.

Li, S. S., Mun, J., Brahman, F., Hosseini, P., Thomas, B. G., Sin, J. M., Ren, B., Ilgen, J. S., Tsvetkov, Y., and Sap, M. ALFA: Aligning LLMs to ask good questions a case study in clinical reasoning. In *Second Conference on Language Modeling*, 2025. URL https://openreview.net/forum?id=12u7diwku0.

Lu, C., Lu, C., Lange, R. T., Foerster, J., Clune, J., and Ha, D. The AI scientist: Towards fully automated open-ended scientific discovery, 2024. URL https://arxiv.org/abs/2408.06292.

Nakano, R., Hilton, J., Balaji, S., Wu, J., Ouyang, L., Kim, C., Hesse, C., Jain, S., Kosaraju, V., Saunders, W., Jiang, X., Cobbe, K., Eloundou, T., Krueger, G., Button, K., Knight, M., Chess, B., and Schulman, J. WebGPT: Browser-assisted question-answering with human feedback, 2022. URL https://arxiv.org/abs/2112.09332.

Press, O., Zhang, M., Min, S., Schmidt, L., Smith, N. A., and Lewis, M. Measuring and narrowing the compositionality gap in language models, 2023. URL https://arxiv.org/abs/2210.03350.

Rein, D., Hou, B. L., Stickland, A. C., Petty, J., Pang, R. Y., Dirani, J., Michael, J., and Bowman, S. R. GPQA: A graduate-level google-proof q&a benchmark. In *First Conference on Language Modeling*, 2024. URL https://openreview.net/forum?id=Ti67584b98.

Romera-Paredes, B., Barekatain, M., Novikov, A., Balog, M., Kumar, M. P., Dupont, E., Ruiz, F. J. R., Ellenberg, J. S., Wang, P., Fawzi, O., Kohli, P., and Fawzi, A. Mathematical discoveries from program search with large language models. *Nature*, 625(7995):468–475, 2024. doi: 10.1038/s41586-023-06924-6. URL https://doi.org/10.1038/s41586-023-06924-6.

Ruan, Y., Band, N., Maddison, C. J., and Hashimoto, T. Reasoning to learn from latent thoughts, 2025. URL https://arxiv.org/abs/2503.18866.

Runcie, N. T., Deane, C. M., and Imrie, F. Assessing the chemical intelligence of large language models. *Journal of Chemical Information and Modeling*, 66(1):216–227, 2026. doi: 10.1021/acs.jcim.5c02145.

Shinn, N., Cassano, F., Gopinath, A., Narasimhan, K. R., and Yao, S. Reflexion: language agents with verbal reinforcement learning. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023. URL https://openreview.net/forum?id=vAElhFcKW6.

Shridhar, K., Stolfo, A., and Sachan, M. Distilling reasoning capabilities into smaller language models, 2023. URL https://arxiv.org/abs/2212.00193.

Snell, C. V., Lee, J., Xu, K., and Kumar, A. Scaling LLM test-time compute optimally can be more effective than scaling parameters for reasoning. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=4FWAwZtd2n.

Sun, L., Han, Y., Zhao, Z., Ma, D., Shen, Z., Chen, B., Chen, L., and Yu, K. SciEval: A multi-level large language model evaluation benchmark for scientific research, 2024. URL https://arxiv.org/abs/2308.13149.

Wang, X., Wei, J., Schuurmans, D., Le, Q. V., Chi, E. H., Narang, S., Chowdhery, A., and Zhou, D. Self-consistency improves chain of thought reasoning in language models. In *The Eleventh International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=1PL1NIMMrw.

Wei, J., Wang, X., Schuurmans, D., Bosma, M., Ichter, B., Xia, F., Chi, E. H., Le, Q. V., and Zhou, D. Chain of thought prompting elicits reasoning in large language models. In Oh, A. H., Agarwal, A., Belgrave, D., and Cho, K. (eds.), *Advances in Neural Information Processing Systems*, 2022. URL https://openreview.net/forum?id=_VjQlMeSB_J.

Wu, Q., Bansal, G., Zhang, J., Wu, Y., Li, B., Zhu, E., Jiang, L., Zhang, X., Zhang, S., Liu, J., Awadallah, A. H., White, R. W., Burger, D., and Wang, C. AutoGen: Enabling next-gen LLM applications via multi-agent conversations. In *First Conference on Language Modeling*, 2024. URL https://openreview.net/forum?id=BAakY1hNKS.

Yanai, I. and Lercher, M. Night science. *Genome Biology*, 20:179, 2019. doi: 10.1186/s13059-019-1800-6.

Yao, S., Yu, D., Zhao, J., Shafran, I., Griffiths, T. L., Cao, Y., and Narasimhan, K. R. Tree of thoughts: Deliberate problem solving with large language models. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023a. URL https://openreview.net/forum?id=5Xc1ecxO1h.

Yao, S., Zhao, J., Yu, D., Du, N., Shafran, I., Narasimhan, K. R., and Cao, Y. ReAct: Synergizing reasoning and acting in language models. In *The Eleventh International Conference on Learning Representations*, 2023b. URL https://openreview.net/forum?id=WE_vluYUL-X.

Zhang, M. J., Knox, W. B., and Choi, E. Modeling future conversation turns to teach LLMs to ask clarifying questions. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=cwuSAR7EKd.

Zhao, R., Köksal, A., Modarressi, A., Hedderich, M. A., and Schütze, H. Do we know what LLMs don't know? a study of consistency in knowledge probing, 2025. URL https://arxiv.org/abs/2505.21701.

Zhou, D., Schärli, N., Hou, L., Wei, J., Scales, N., Wang, X., Schuurmans, D., Cui, C., Bousquet, O., Le, Q. V., and Chi, E. H. Least-to-most prompting enables complex reasoning in large language models. In *The Eleventh International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=WZH7099tgfM.

A. Extended Related Work

This appendix expands the body discussion of related work (Section 2) into a self-contained survey of question-asking research relevant to CoQ. We organize the material into the same clusters as the body and conclude with a side-by-side comparison matrix (Table 2) on the axes that determine whether a given prior method addresses the same scientific question we ask.

A.1. Reasoning Traces, Sampled Compute, and Test-Time Scaling

Chain-of-thought (CoT) prompting demonstrated that asking a model to write intermediate reasoning before answering can unlock multi-step reasoning at sufficient model scale (Wei et al., 2022). The zero-shot variant (Kojima et al., 2022) reduced the interface to a short instruction ("let's think step by step"). Self-consistency (Wang et al., 2023) showed that sampling multiple reasoning trajectories and aggregating answers is a reliable improvement over single-trace decoding. More recent work on test-time compute (Snell et al., 2025) formalizes the trade-off between model size and inference budget, while reasoning-tuned models such as DeepSeek-R1 (Guo et al., 2025) demonstrate that long reasoning traces can be elicited via reinforcement learning without supervised fine-tuning.

Our work sits inside this lineage but reframes the unit being sampled. Where CoT samples derivations and self-consistency aggregates over them, CoQ samples *decompositions*: a list of questions whose answers would close the reasoning gap. Sampling and aggregation are still available, but they now occur over a representation that is both shorter and more directly inspectable. Our token-normalized analyses (Sections 4, 5.1) ask whether this shift in representation is worth its compute, an analysis that single-task CoT extensions typically omit.

A.2. Decomposition and Search

Least-to-most prompting (Zhou et al., 2023) explicitly decomposes problems into ordered subproblems and solves them sequentially; Tree-of-Thoughts (Yao et al., 2023a) generalizes linear reasoning into search over intermediate states. Both methods commit to a control structure (sequential decomposition, or tree search) and require the user to specify a state representation. CoQ is lighter: it produces an unordered question list as a representation, and any control structure (sequential answering, parallel routing, no answering) can be applied on top. The benefit is that the same artifact can be used at inference time, as training data, and as a routing interface, without committing to one consumption pattern.

A.3. Inference-Time Question-Asking

Self-Ask (Press et al., 2023) demonstrated that an LLM can generate and answer its own subquestions to bridge compositional fact-retrieval gaps, particularly when paired with an external search tool. The method is purely inference-time: no training is performed on the question-asking behavior, and question quality is read off final-task accuracy. Reflexion (Shinn et al., 2023) adds a self-critique loop in which the model verbalizes errors and adjusts its next attempt. ReAct (Yao et al., 2023b) interleaves thoughts and tool actions for information gathering. Critical Questions of Thought (Castagna et al., 2024) extends this by having the model generate argumentative questions that probe its own derivation, and ALFA (Li et al., 2025) uses preference-style rubrics over question quality.

These methods share two structural limitations from our viewpoint. First, question quality is supervised only by final-task outcomes, which entangles asking, answering, and producing. A question that happens to appear in a correct trace cannot be distinguished from one that is causally necessary. Our oracle-answer ablation (Section 5.3) is designed to break exactly this entanglement. Second, the methods are evaluated on the benchmarks they target, not as transferable capabilities; whether the question-asking behavior is a benchmark-bound prompt artifact or a general reasoning skill is not tested. Our latent-question transfer experiments (Section 5.2) test this directly.

A.4. Clarifying-Question Generation for Disambiguation

A second large cluster of work trains models to ask clarifying questions of *users* when inputs are ambiguous or underspecified. STaR-GATE (Andukuri et al., 2024) self-trains an instruct model to generate questions that elicit user task descriptions, evaluated by user satisfaction. Clarinet (Chi et al., 2024) fine-tunes an LLM to ask retrieval-disambiguating queries, measured by retrieval accuracy. MEDIQ (Li et al., 2024) studies clarifying behavior for clinical reasoning where confidence-thresholded asking matters. QuestBench (Li et al., 2026) introduces a benchmark for clarifying behavior. Zhang et al. (2025) train models on simulated future conversation turns so that asking a clarifying question now is rewarded against a simulated

future of better answers.

These systems share an architecture: the questioner asks the user; the user provides missing input; the system then completes the original task. This architecture is the right one for ambiguity resolution but is not the architecture we study. The questions in CoQ are not addressed to the user; they probe the model's own knowledge gaps and may be answered by the same model, a stronger model, a tool, or no answerer at all. Concretely:

- *Evaluation locus.* Clarifying questions are evaluated by user satisfaction or retrieval accuracy on a single task; epistemic questions can be evaluated by domain experts who never see the original task prompt.
- *Conditioning.* Clarifying questions are conditioned on natural ambiguity in user inputs; epistemic questions are conditioned on the gap between the model's competence and the task's requirements.
- *Transferability.* Clarifying-question training signals are intrinsically per-task; epistemic-question training, as we show in Section 5.2, transfers across benchmarks.

This separation is not a definitional convenience: it determines who answers the question, how correctness is assigned, and how the skill is supervised.

A.5. Concurrent Work: Asking the Right Questions (ARQ)

Hu et al. (2026) is the closest concurrent work on training language models to ask better reasoning questions. Their stepping-stone formulation has the model emit auxiliary questions that decompose competition-math problems, with both supervised and reinforcement post-training applied to a single frontier-scale model on AIME. The main ARQ result is consistent with our own thesis: trainable question generation can improve reasoning.

Three differences make our paper a complement to, not a duplicate of, ARQ.

(i) Breadth of empirical evidence. ARQ evaluates one benchmark (AIME) on one model class (frontier-scale). We evaluate CoQ across mathematics, science, chemistry, and code benchmarks on more than twenty open and frontier models from 7B instruction-tuned to frontier closed APIs. The breadth is not cosmetic: it lets us *characterize* when explicit questioning helps and when it does not, an empirical question that single-benchmark evaluations cannot answer in principle. It also lets us identify the model-family and budget regimes where explicit questioning is cost-effective, which in turn determines where the interface is deployable in practice.

(ii) Cross-domain latent-question transfer. ARQ trains and evaluates within a single math benchmark. We instead ask whether teacher-synthesized question decompositions learned on one source benchmark can improve reasoning on an unrelated target benchmark. This tests a different hypothesis: that the portable object is a question-space representation of missing information, not only a stronger in-domain solver.

(iii) Mechanistic decomposition. Our oracle-answer ablation (Section 5.3) holds the prompt fixed but varies whether questions are answered by an oracle, only by a quality-filtered subset, randomly answered, not answered, or replaced by no question phase at all. The five conditions disentangle (a) the structural benefit of asking, (b) the value of *which* questions are asked, and (c) the value of the answers received. Without this decomposition, it is impossible to tell whether ARQ-style gains come from useful decomposition or from incidental information transfer through question-conditioned generation. We provide that decomposition.

For these reasons, we view ARQ as concurrent evidence that the broader hypothesis is plausible, not as an existing answer to the questions this paper studies.

A.6. Co-Scientist Systems and Scientific Discovery

A growing line of work targets language models as scientific partners. ChemCrow (Bran et al., 2023) augments LLMs with chemistry tool use; the AI Scientist (Lu et al., 2024) attempts end-to-end automated research; FunSearch (Romera-Paredes et al., 2024) couples LLM proposers with verifiers for mathematical discovery; Google's co-scientist project (Gottweis et al., 2025) explicitly targets multi-agent collaboration with human scientists. SciEval (Sun et al., 2024) provides a multi-level benchmark for scientific reasoning. The wider alignment program of WebGPT (Nakano et al., 2022), Sparrow (Glaese et al., 2022), and Constitutional AI (Bai et al., 2022) treats question–answer interaction as the training signal for helpful and grounded behavior.

A consistent thread across these systems is that question-asking is treated as a control-flow primitive ("ask the tool what the boiling point is", "ask the user to confirm a parameter") rather than as a trainable representation in its own right. Our position is that scientific partnership requires the latter. A co-scientist that cannot identify and articulate a specific knowledge gap cannot extract tacit expert knowledge, cannot prioritize follow-up experiments, and cannot expose its reasoning to expert review. CoQ provides the supervised, judged representation of such gaps that those systems will need.

A.7. Multi-Agent Question–Answer Frameworks

Multi-agent role-play frameworks such as CAMEL (Li et al., 2023) and AutoGen (Wu et al., 2024) provide infrastructure for setting one agent as a questioner and another as a responder, often for task decomposition. Constitutional AI (Bai et al., 2022) uses Q&A interaction as a training signal for harmlessness. These frameworks are valuable as *infrastructure*, but they do not train models to ask better epistemic questions and they do not isolate question-asking ability from final-task performance. CoQ is the kind of representation that such frameworks could benefit from supervising directly.

A.8. Cost-Aware Inference and Speculative Routing

Speculative decoding (Leviathan et al., 2023; Chen et al., 2023) allocates compute by having a cheap draft model propose tokens and a stronger verifier accept or reject them. The same idea has been generalized to model cascades and tier-aware inference at the request level. Our learned-routing extension (Section 5.3) lifts the speculative principle to the *question* level: a draft model emits subquestions and either answers each locally or escalates a marker to a stronger tier. Because each question is short and self-contained, escalation can be priced per question rather than per response, and a learned router can be trained from oracle-labeled minimum-tier data. We are not aware of prior work that uses question structure as the unit of cost-aware routing.

A.9. Calibration, Abstention, and Knowledge Probing

A long line of work studies whether models know what they do not know (Kadavath et al., 2022; Zhao et al., 2025; Fu et al., 2026). The dominant formulation is a global confidence estimate or an abstention decision over the final answer. CoQ provides a finer-grained interface: rather than estimating "how confident am I in the answer?", it asks "what specifically would I need to know to be confident?". Whether explicit question generation can improve abstention quality, calibration, or active information acquisition is a natural follow-up that we discuss in Section 6 but defer formal evaluation of to future work.

A.10. Comparison Matrix

Table 2 summarizes how the closest prior systems compare to CoQ on six axes that determine whether they address the same scientific question. The axes are: (i) the kind of question asked (*Epis.* = epistemic, *Clar.* = clarifying); (ii) whether the questioning behavior is *trained* or only inference-time; (iii) whether question quality is judged *independently* of final-task accuracy; (iv) whether the method evaluates *transfer* of the question-asking skill across benchmarks; (v) whether *token-normalized* cost analysis is reported; (vi) whether the question representation is reused as a *routing* or latent-transfer interface beyond inference-time prompting.

The table is summary, not score. The "Indep. judge" column captures whether question quality is supervised by anything other than final-task accuracy; we treat this as the central methodological gap. The "Transfer" column captures whether the trained question-asking skill is shown to improve unrelated benchmarks under standard prompting; this is the byproduct-of-reasoning hypothesis from Section 1. The "Routing/latent" column captures whether the question representation is reused beyond inference-time prompting.

The pattern is consistent: prior work addresses one or two of these axes; the *combination* is what defines our scope, and it is what motivates evaluating CoQ as a unifying interface rather than as a single new prompt.

B. Example Prompt Templates

This appendix gives the CoQ prompt templates used across the benchmark matrix. Each benchmark has its own family of CoQ variants; the most CoT-like variants are omitted here (they read as "think step by step; list a few questions; then solve" with minor phrasing changes). What follows are the variants whose framing or guiding sub-questions substantively differ

Table 2. How CoQ compares to the closest related systems on the axes that determine whether they address the same scientific question. ✓ = addressed; ✗ = not addressed; ❍ = partially addressed. "Q kind" is the dominant question type studied (Epis. = epistemic, Clar. = clarifying); "–" indicates question-asking is not the focus. "Indep. judge" is whether question quality is supervised by anything other than final-task accuracy.

System	Q kind	Trained	Indep. judge	Transfer	Token-norm.	Routing/latent
CoT (Wei et al., 2022)	–	✗	✗	✗	✗	✗
Self-Consistency (Wang et al., 2023)	–	✗	✗	✗	❍	✗
Least-to-Most (Zhou et al., 2023)	–	✗	✗	✗	✗	✗
Tree-of-Thoughts (Yao et al., 2023a)	–	✗	✗	✗	✗	✗
ReAct (Yao et al., 2023b)	Epis.	✗	✗	✗	✗	❍
Reflexion (Shinn et al., 2023)	Epis.	❍	✗	✗	✗	✗
Self-Ask (Press et al., 2023)	Epis.	✗	✗	✗	✗	✗
Socratic distill. (Shridhar et al., 2023)	Epis.	✓	✗	✗	✗	❍
STaR-GATE (Andukuri et al., 2024)	Clar.	✓	✗	✗	✗	✗
Clarinet (Chi et al., 2024)	Clar.	✓	✗	✗	✗	✗
MEDIQ (Li et al., 2024)	Clar.	✓	✗	✗	✗	✗
QuestBench (Li et al., 2026)	Clar.	✗	✗	✗	✗	✗
ARQ (concurrent) (Hu et al., 2026)	Epis.	✓	✗	✗	✗	✗
CoQ (this work)	Epis.	✓	✓	✓	✓	✓

from chain-of-thought, organized by benchmark and labelled v1, v2, … for ease of reference. The motivating example (Section 1) uses an additional, special-purpose prompt shown at the end.

B.1. Mathematics (MATH, AIME)

Math CoQ v1: open-ended sub-question decomposition

First, identify the sub-questions you need to answer to solve this problem (could be non-trivial questions to decompose the problem into smaller sub-problems, questions that fill knowledge gaps, etc.). List them clearly.
Problem: {problem}
Sub-questions to answer:
After listing the sub-questions, provide the complete solution.

Math CoQ v2: chain of questions and answers

To solve this math problem, reason step by step using a chain of questions and answers. Ask yourself the most important questions needed to understand and solve it (about the goal, given information, unknown quantities, and solution approach). For each question, write a short answer that updates your understanding. Once your questions have been answered and the solution is clear, provide your complete solution.
Problem: {problem}
Chain of questions and answers (reasoning first):

B.2. Science (GPQA-Diamond)

GPQA CoQ v1: scientific sub-question scaffold

Before answering this multiple-choice question, identify the key sub-questions and information gaps you need to address. Think about:

- What scientific concepts or principles are relevant?
- What information is given and what needs to be determined?
- What assumptions or conditions need to be considered?
- What are the relationships between the concepts involved?

List these key questions, then work through the solution systematically.

Question: {question}
Choices: A) {choice_a} B) {choice_b} C) {choice_c} D) {choice_d}
Key questions and solution:

GPQA CoQ v2: unknowns-first decomposition

Before solving this problem, think about what questions would help you understand and solve it step by step. Consider:

- What are the key unknowns or quantities to determine?
- What scientific principles or theories apply?
- What information is needed to evaluate each choice?
- What intermediate reasoning steps are required?

Once you've identified these questions, address each one systematically and provide your answer.
Question: {question}
Choices: A) {choice_a} B) {choice_b} C) {choice_c} D) {choice_d}
Questions to address and solution:

B.3. Code (HumanEval, MBPP)

Code CoQ v1: specification-and-edge-case sub-questions

First, identify the key sub-problems and questions you need to answer to solve this coding problem. Think about:

- What is the function supposed to do?
- What are the edge cases to consider?
- What data structures or algorithms might be needed?
- What are the constraints or requirements?

List these sub-questions clearly, then provide the complete solution.
Problem: {prompt}
Sub-questions and solution:

B.4. Chemistry (ChemIQ, ChemBench)

Chemistry CoQ v1: bulleted chemical sub-problems

To solve this chemistry problem effectively, start by breaking it down into sub-problems. Ask yourself:

- What is the problem asking for?
- What chemical concepts or principles might be useful?
- What relationships or properties can be applied?
- What steps are needed to reach the answer?

After identifying these sub-problems, solve each one and provide your complete solution.
Problem: {prompt}
Sub-problems and solution:

Chemistry CoQ v2: chain of questions and answers

To solve this chemistry problem, reason step by step using a chain of questions and answers. Ask yourself the most important questions needed to understand and solve it (about the goal, given information, chemical concepts, relationships, and methods). For each question, write a short answer that updates your understanding. Once your questions have been answered and the solution is clear, provide your complete solution.
Problem: {prompt}
Chain of questions and answers (reasoning first):

B.5. Competitive Programming (LiveCodeBench, BigCodeBench, APPS)

Competitive-programming CoQ v1: chain of questions and answers

You are a highly capable competitive programming assistant.
Reason step by step using a chain of questions and answers. Ask yourself the most important questions needed to understand and solve the problem (goal, inputs/outputs, constraints, edge cases, algorithms). For each question, write a short answer. Once your questions have been answered and the solution is clear, write the final Python code only.
Problem:
{problem}
Starter code (you may modify or replace it):
{starter_code}
Chain of questions and answers (reasoning first):

B.6. Modular Exponentiation Recipe (Motivating Example)

The *+Recipe* condition in Table **??** prepends the following short note before the problem statement.

Modular exponentiation: Carmichael + CRT recipe

Appendix: Modular Exponentiation Guide. To compute $a^k \bmod n$ efficiently:
1. Carmichael / Euler reduction (preferred). If $\gcd(a, n) = 1$, compute the Carmichael function $\lambda(n)$ (or $\varphi(n)$). Then reduce the exponent: $e = k \bmod \lambda(n)$. Compute $a^e \bmod n$ directly (e.g., via binary exponentiation). This often suffices without CRT.
2. Chinese Remainder Theorem (only if needed). If n has several coprime factors and residues differ, factor $n = \prod_i p_i^{e_i}$. Compute $r_i = a^k \bmod p_i^{e_i}$ (reducing exponents modulo $\lambda(p_i^{e_i})$). Solve $x \equiv r_i \pmod{p_i^{e_i}}$.
3. When $\gcd(a, n) > 1$**.** Evaluate powers directly on each prime-power factor where a shares factors with n.
Example: $4^{109} \bmod 185$. $\lambda(185) = \mathrm{lcm}(\varphi(5), \varphi(37)) = \mathrm{lcm}(4, 36) = 36 \Rightarrow 109 \bmod 36 = 1 \Rightarrow$ result $= 4$.

B.7. Teacher Prompts for Gap-Targeted Question Training

Gap-targeted question training (Appendix D) supervises a student to generate questions that recover specific masked steps in a worked solution. The teacher is asked to produce two contrastive question lists per masked example: a *good* list whose questions target the specific gap, and a *bad* list whose questions are plausible but tangential.

Teacher prompt: good (gap-targeting) questions

Here is a math problem and a PARTIAL solution with some steps missing.
Problem: {problem}
Partial solution:
{partial_solution}
Generate 1–3 specific questions that, if answered, would help recover the missing reasoning. The questions should:

- Target the specific gap (not generic).
- Be answerable from mathematical knowledge.
- Help reconstruct the missing steps.

Questions:

Teacher prompt: bad (contrastive) questions

Here is a math problem with some solution steps masked.
Problem: {problem}
Partial solution:
{partial_solution}
Generate 2–3 questions that are RELATED to the problem domain but would NOT help recover the masked steps. These should be plausible math questions but miss the actual epistemic need. Types of bad questions to generate:

- Too generic (“What formula should I use?”).
- Wrong aspect: ask about a value already given in the visible steps.
- Too broad: ask how to solve the whole problem class.

- Tangential: ask for a definition that does not resolve the missing step.
- Premature: ask for the final answer when early or middle steps are masked.

Keep the questions ON-TOPIC but NOT HELPFUL for recovering the specific missing reasoning.
Bad questions (plausible but unhelpful):

C. Per-Experiment Details

This appendix gives benchmark- and experiment-level detail that supports the main-text claims without overloading the body. Each subsection states the protocol, the per-model results, and the omissions and caveats relevant to its benchmark family.

C.1. Core CoQ versus CoT Matrix

The matched-prompt matrix spans mathematics, science, chemistry, and code. CoQ prompt families are compared against matched CoT families under greedy and sampled decoding. A separate direct-control lane is used for benchmarks where public or benchmark-native prompting differs from our matched-prompt protocol; that lane provides the public anchor, while the CoQ/CoT family-mean comparison is the unit on which we evaluate whether question structure helps.

The qualitative read is consistent across the matrix: CoQ is not uniformly better than CoT, but the gain concentrates where the task bottleneck is identifying missing information, choosing a sub-task, or deciding what evidence to seek, rather than executing a known procedure.

Two design choices follow from the protocol. First, the paper reports pass@k only from final merged summaries, not from intermediate checkpoints. Second, completion-token-normalized pass@k is treated as a core analysis rather than a secondary check, because CoQ traces are on average longer than matched CoT traces and equal-sample comparisons alone can confuse more reasoning with more compute.

C.2. Compute Resources

Open-model evaluations and repair reruns were executed as containerized Slurm jobs on an internal GPU cluster. A typical single-model lane used one GPU worker with 80 GB-class accelerator memory for 7B–32B models, larger-memory or multi-GPU workers for 70B models when required by the inference engine, 16–64 CPU cores for data loading and grading, and scratch storage for transient generations. Sampling lanes generated up to 32 completions per problem and prompt strategy. Small HumanEval, MBPP, GPQA-Diamond, AIME, MATH, ChemIQ, and ChemBench lanes typically completed within minutes to a few hours, whereas APPS, BigCodeBench, LiveCodeBench, and sampled 32B/70B lanes were submitted with multi-hour to one-day wall-clock limits and sometimes required resume or rerun after wall-clock, provider, or request-timeout failures.

Frontier-model, oracle-ablation, routing, speculative-questioning, and adaptive-question-generation experiments were API-bound CPU jobs; their runtime was dominated by provider latency and token volume rather than local GPU time. Merge, grading, auditing, and plotting jobs ran on CPU workers and usually completed within minutes to under an hour. The full research campaign consumed more compute than the final figures alone: preliminary sweeps, failed submissions, contaminated runs, and diagnostic experiments were audited and, when needed, rerun.

C.3. Cross-Benchmark Reference Tables

Tables 3 and 4 keep the corresponding full numerical matrix, including neutral and negative slices, in the appendix.

C.4. Benchmark-Separated Plot Banks

Each benchmark family has a plot bank with four views: equal-sample bars at pass@1, pass@8, and pass@32 when available; low- and high-budget token-normalized bars; equal-sample pass@k curves; and completion-token-normalized curves. The underlying numbers for each bank are stored alongside the figures as `bar_data.tsv` and `bar_data.json`.

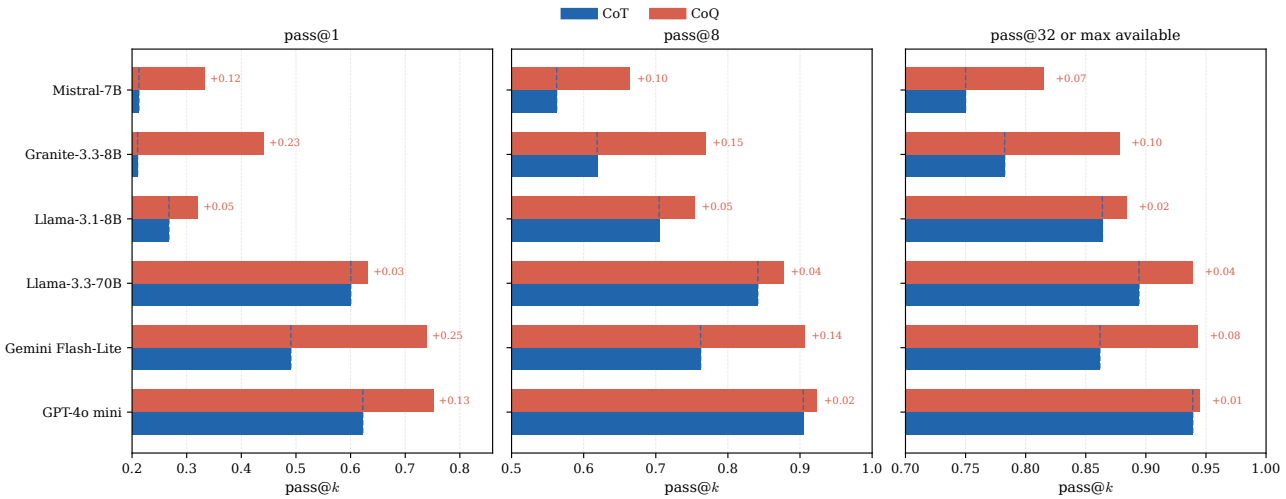

Figure 8. HumanEval equal-sample bar chart bank. Panels report pass@1, pass@8, and pass@32 where available, falling back to the largest sampled budget for lanes with smaller grids.

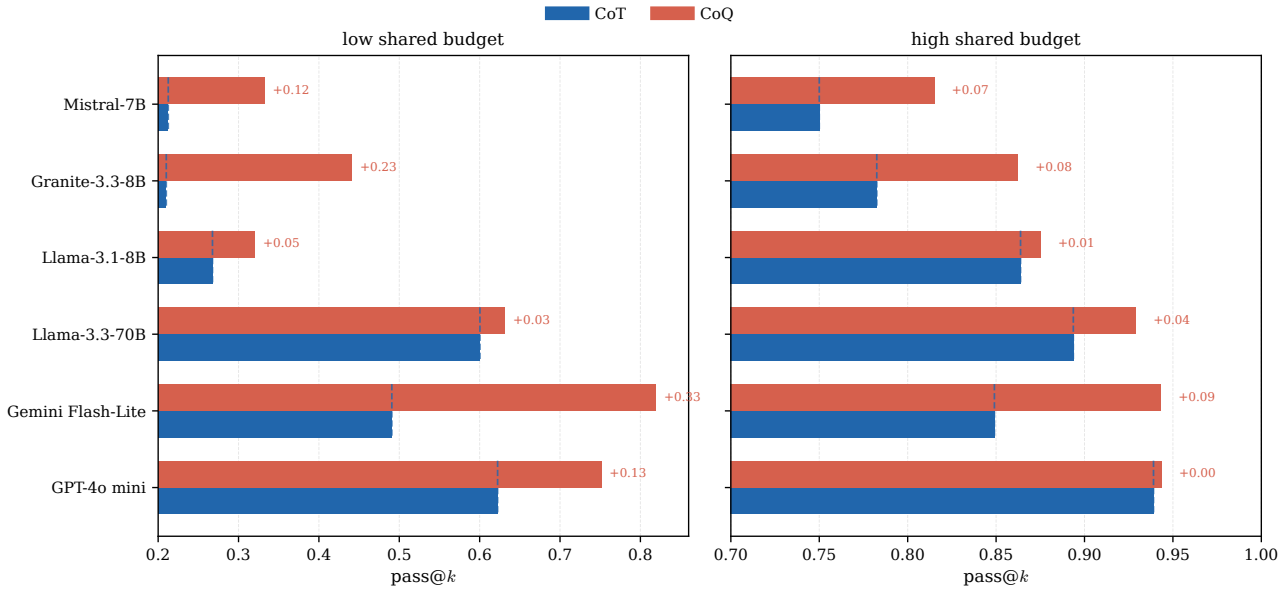

Figure 9. HumanEval token-normalized bar chart bank. The low-budget and high-budget panels compare matched CoT and CoQ family means at paired completion-token budgets.

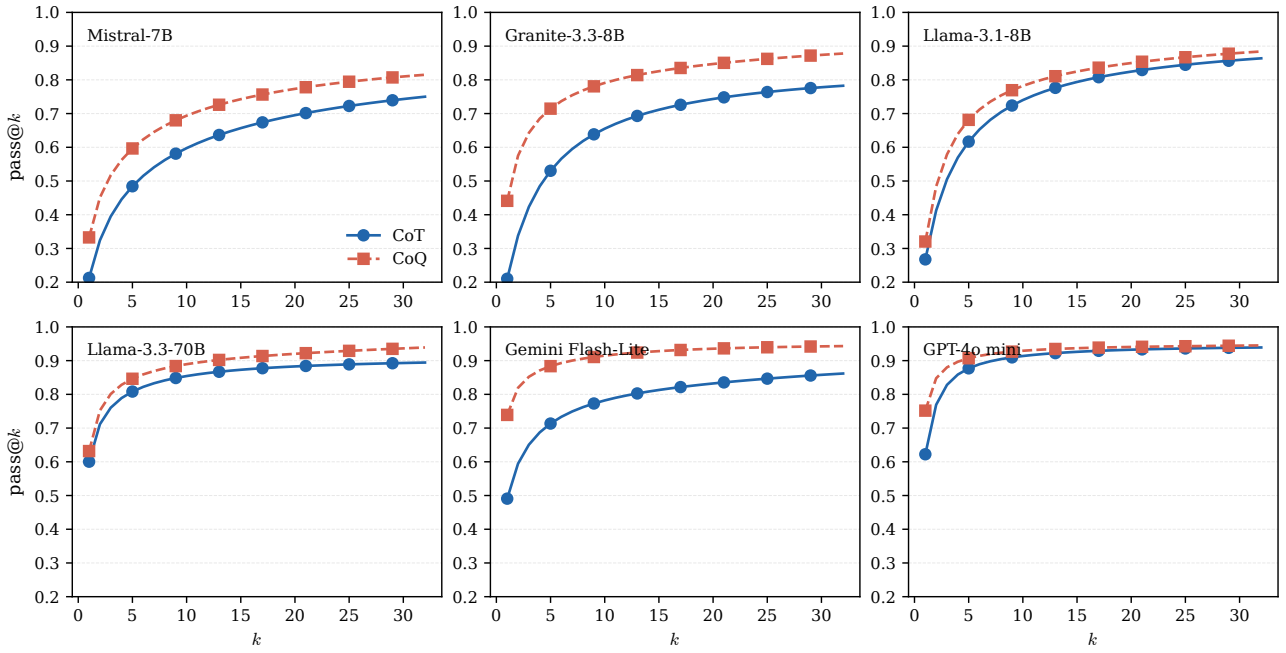

Figure 10. HumanEval equal-sample pass@k curve bank. These panels preserve the full sample-budget trajectory used to choose compact main-text summaries.

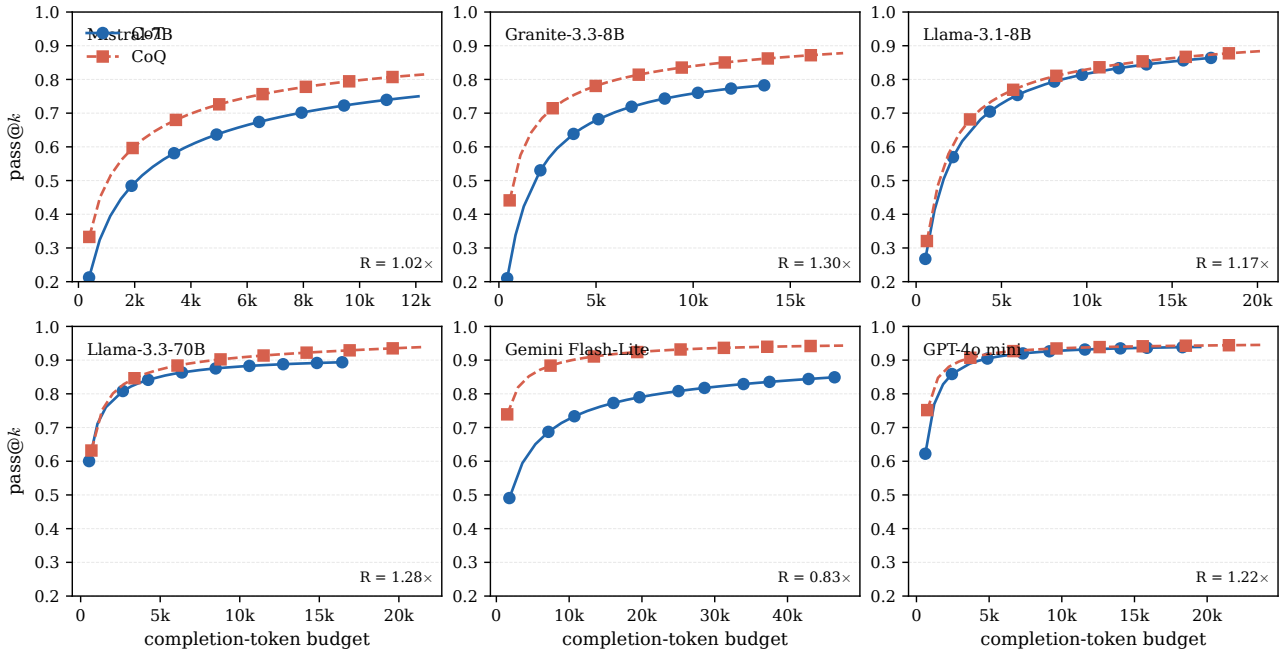

Figure 11. HumanEval completion-token-normalized curve bank. These panels show how the CoT–CoQ comparison evolves as the shared completion-token budget grows.

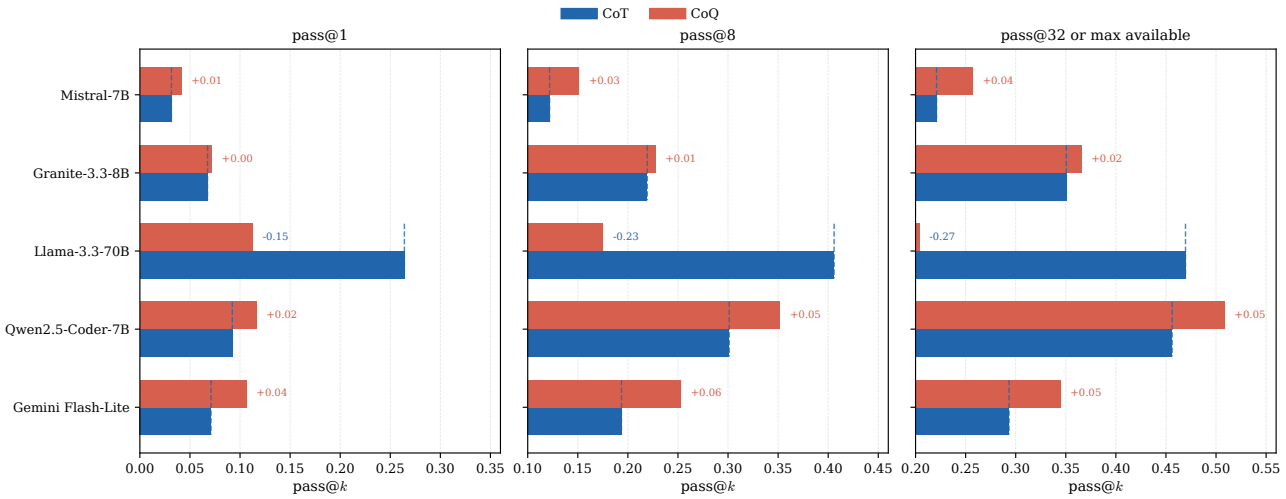

Figure 12. LiveCodeBench equal-sample bar chart bank. Panels report pass@1, pass@8, and pass@32 where available, falling back to the largest sampled budget for lanes with smaller grids.

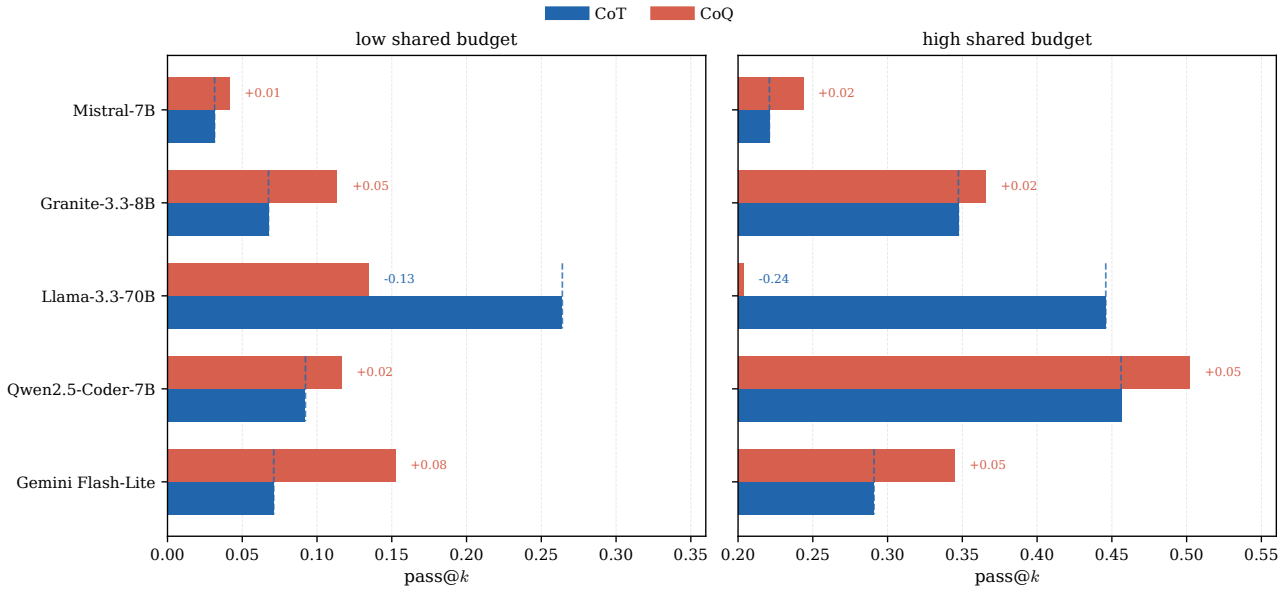

Figure 13. LiveCodeBench token-normalized bar chart bank. The low-budget and high-budget panels compare matched CoT and CoQ family means at paired completion-token budgets.

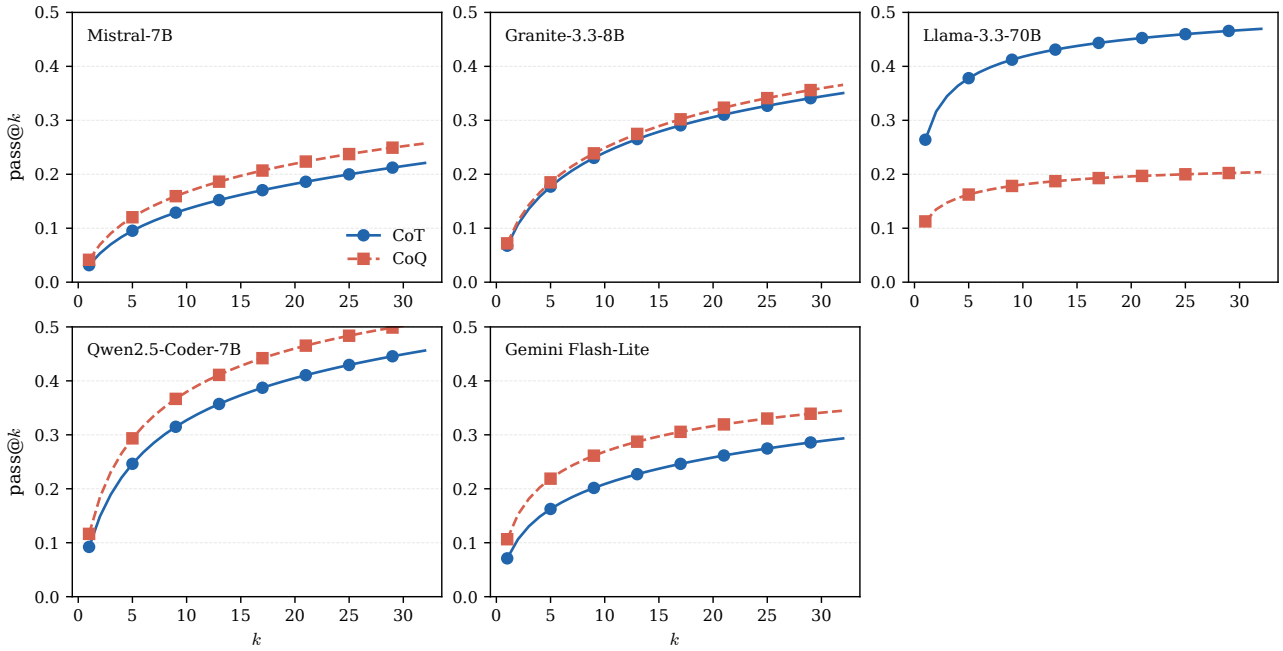

Figure 14. LiveCodeBench equal-sample pass@k curve bank. These panels preserve the full sample-budget trajectory used to choose compact main-text summaries.

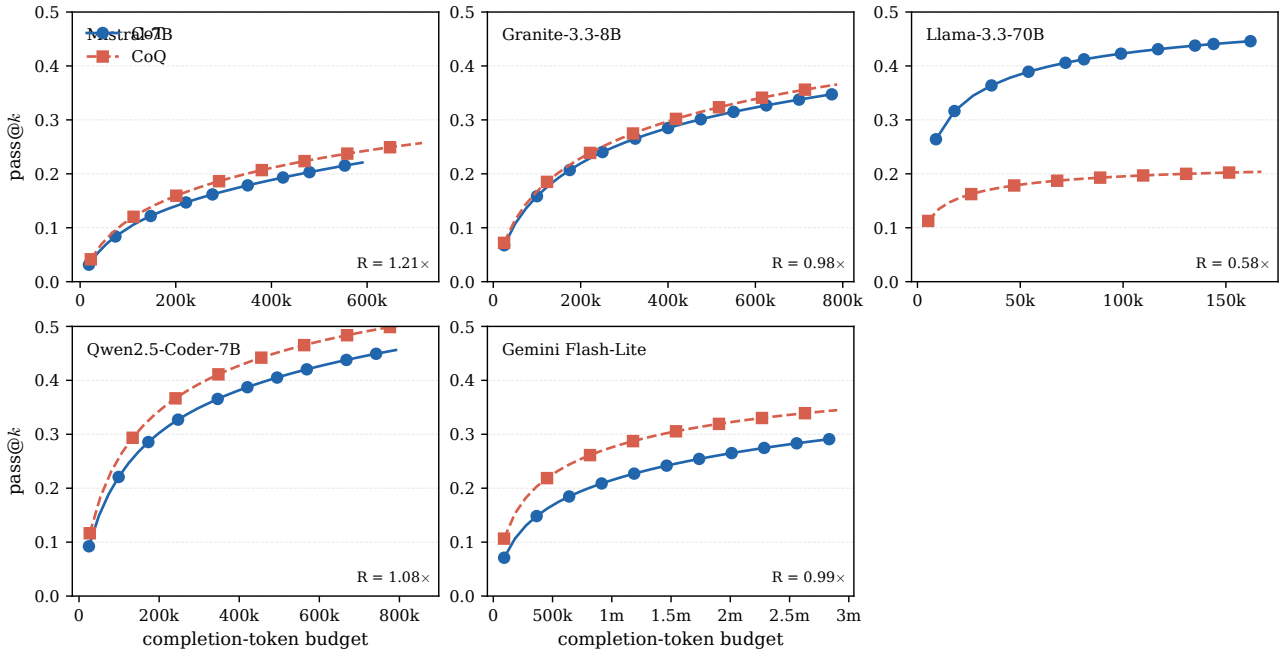

Figure 15. LiveCodeBench completion-token-normalized curve bank. These panels show how the CoT–CoQ comparison evolves as the shared completion-token budget grows.

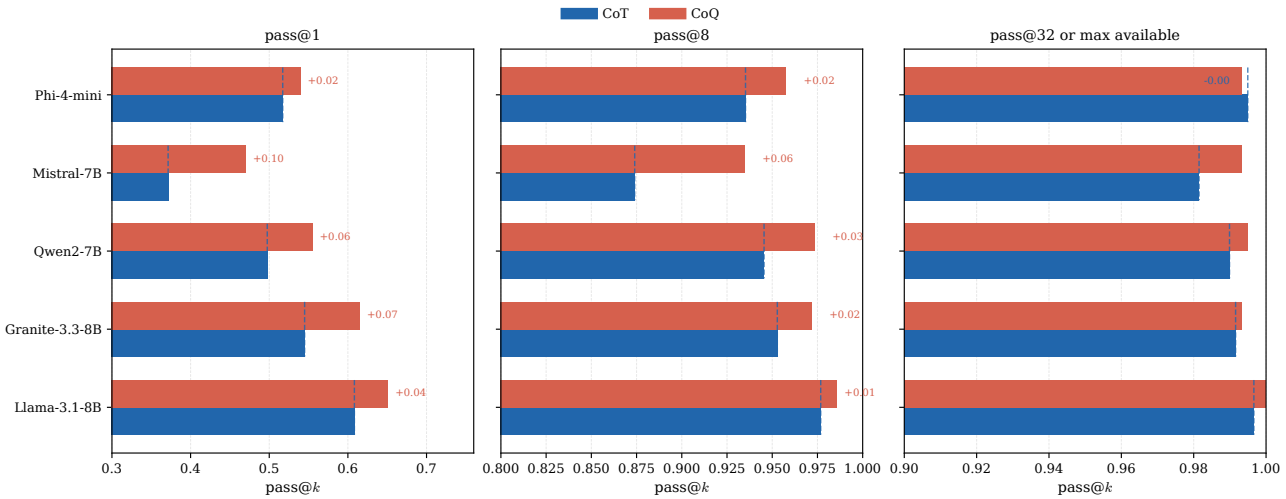

Figure 16. GPQA-Diamond equal-sample bar chart bank. Panels report pass@1, pass@8, and pass@32 where available, falling back to the largest sampled budget for lanes with smaller grids.

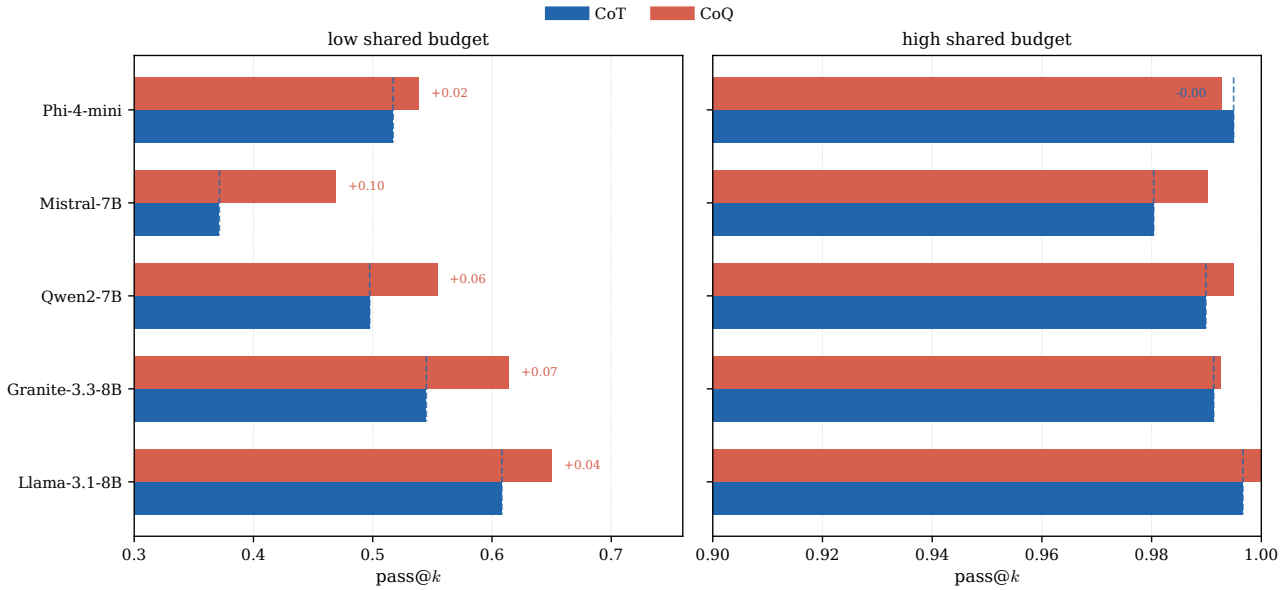

Figure 17. GPQA-Diamond token-normalized bar chart bank. The low-budget and high-budget panels compare matched CoT and CoQ family means at paired completion-token budgets.

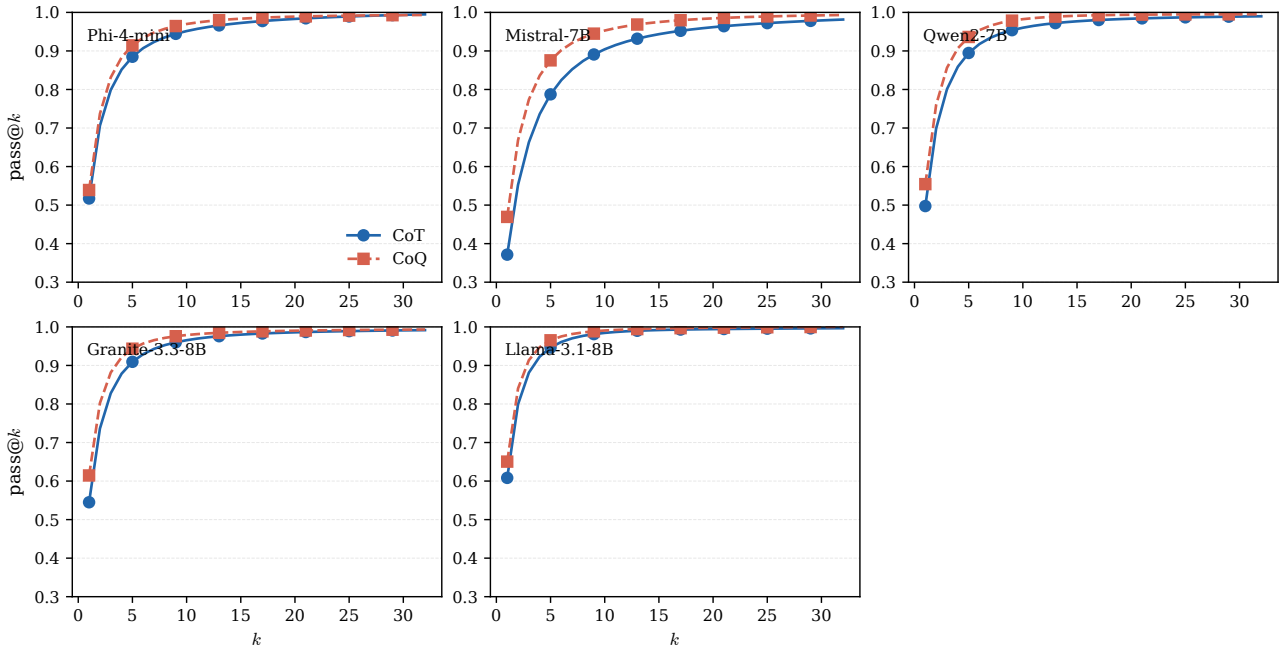

Figure 18. GPQA-Diamond equal-sample pass@k curve bank. These panels preserve the full sample-budget trajectory used to choose compact main-text summaries.

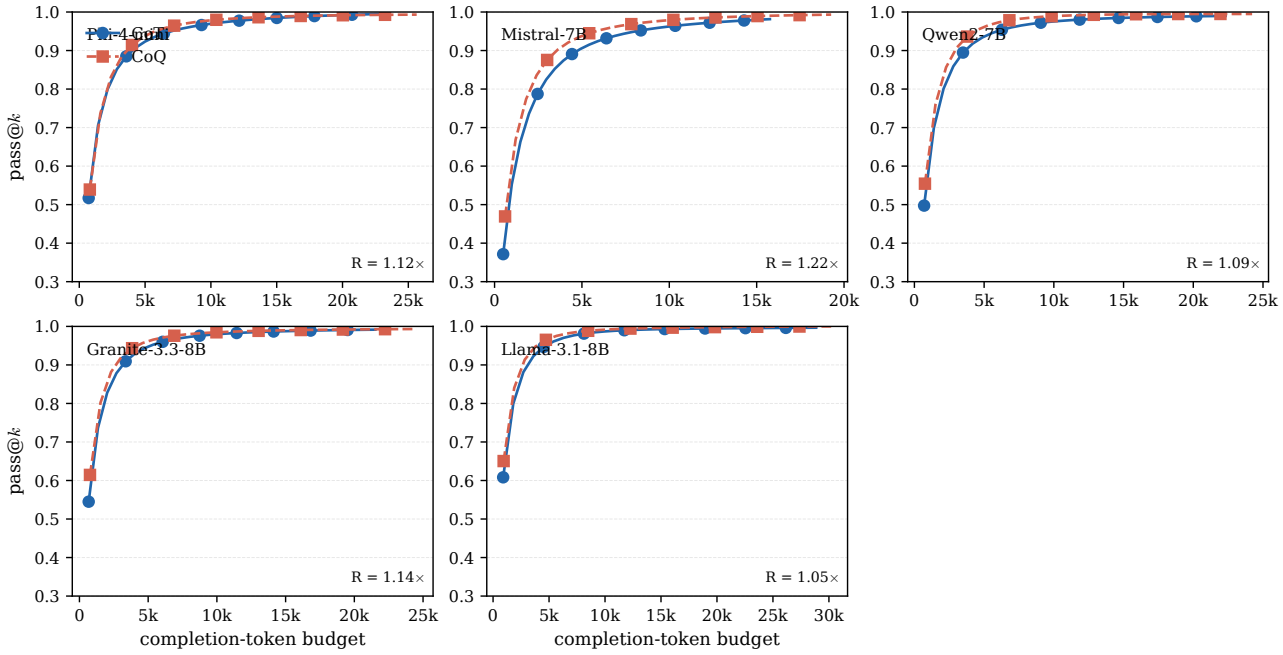

Figure 19. GPQA-Diamond completion-token-normalized curve bank. These panels show how the CoT–CoQ comparison evolves as the shared completion-token budget grows.

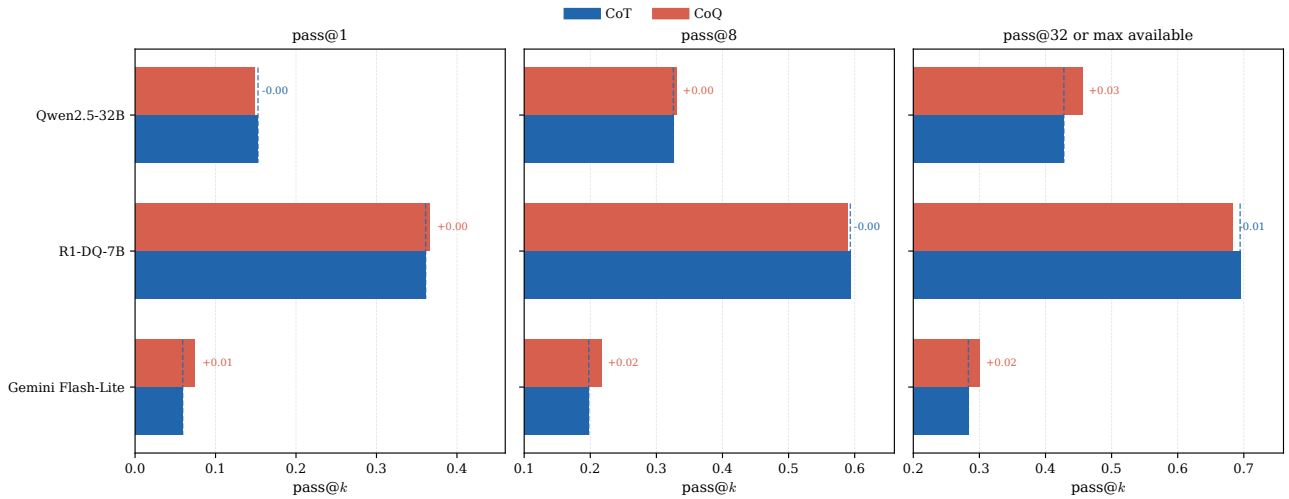

Figure 20. AIME equal-sample bar chart bank. Panels report pass@1, pass@8, and pass@32 where available, falling back to the largest sampled budget for lanes with smaller grids.

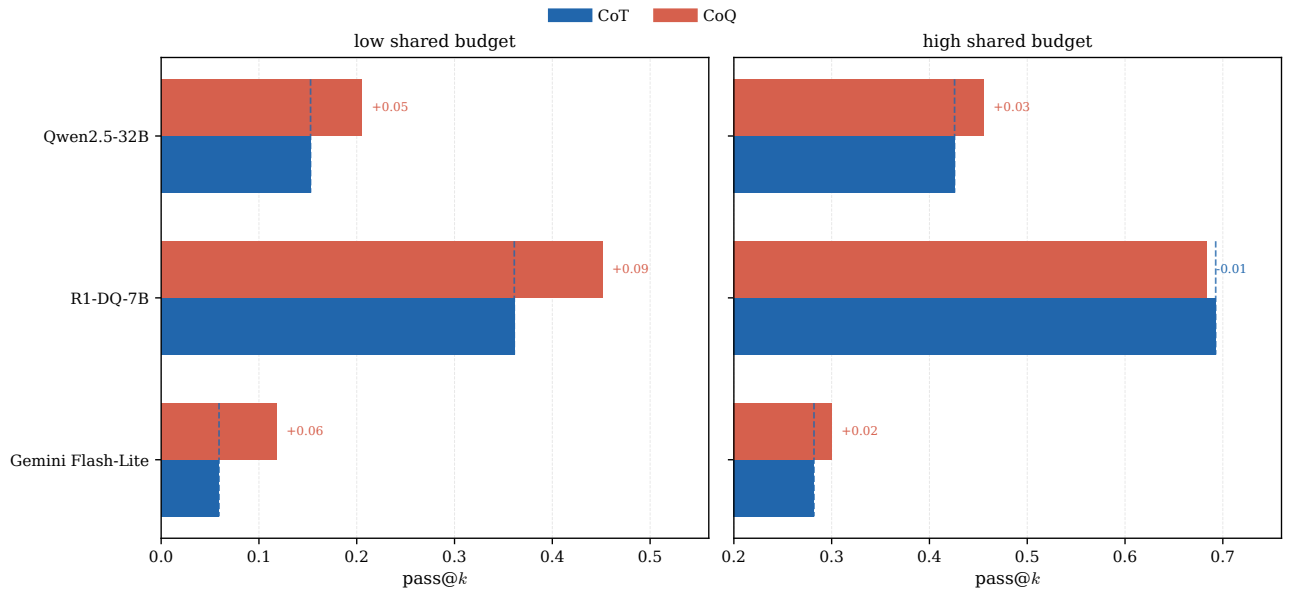

Figure 21. AIME token-normalized bar chart bank. The low-budget and high-budget panels compare matched CoT and CoQ family means at paired completion-token budgets.

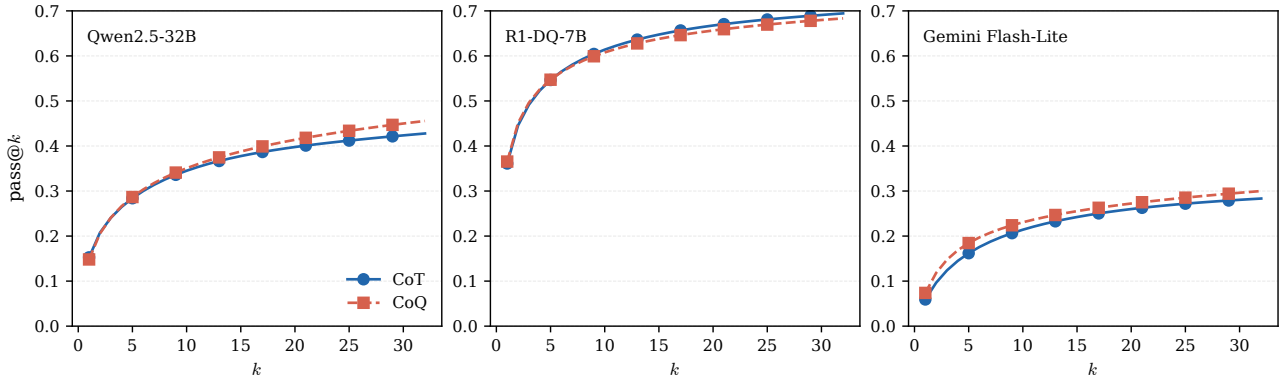

Figure 22. AIME equal-sample pass@k curve bank. These panels preserve the full sample-budget trajectory used to choose compact main-text summaries.

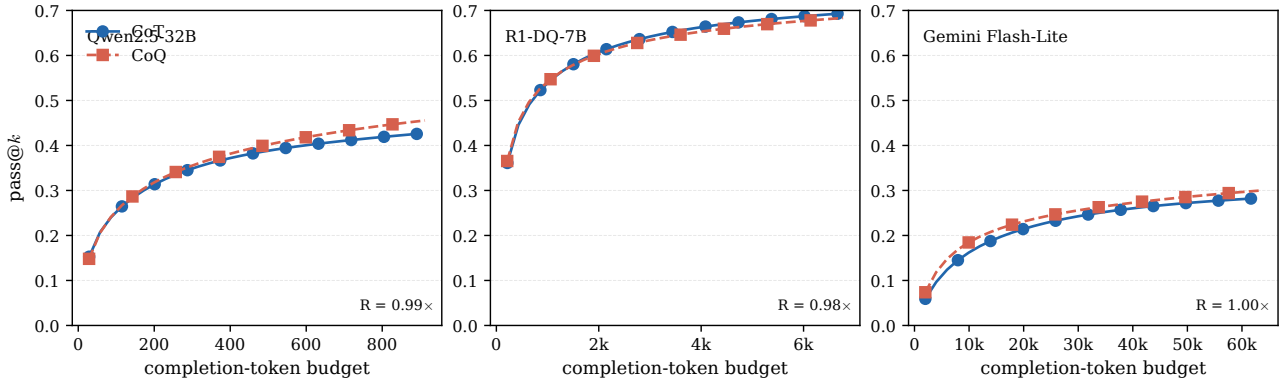

Figure 23. AIME completion-token-normalized curve bank. These panels show how the CoT–CoQ comparison evolves as the shared completion-token budget grows.

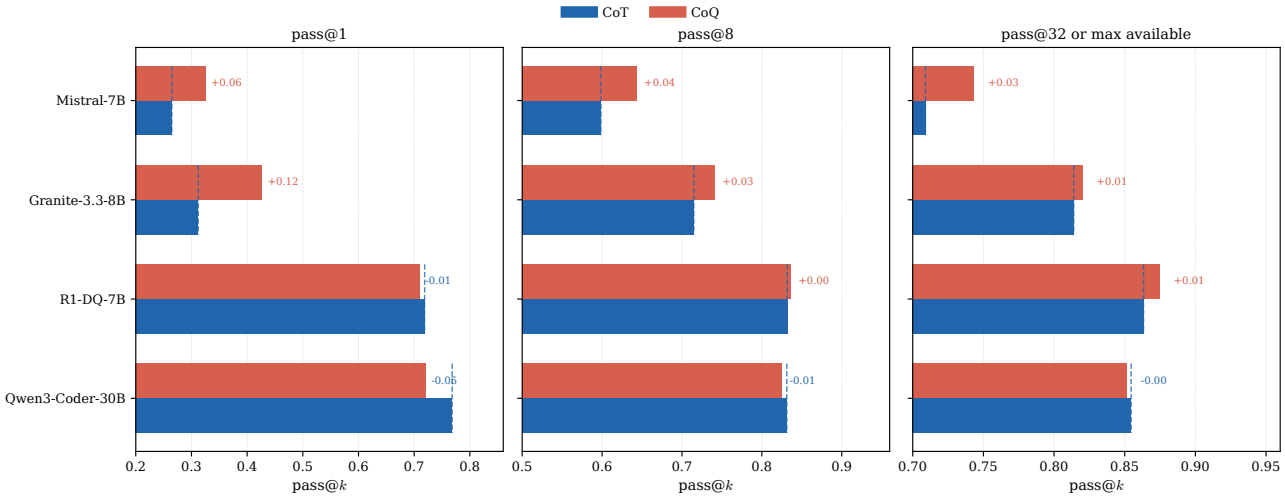

Figure 24. MBPP equal-sample bar chart bank. Panels report pass@1, pass@8, and pass@32 where available, falling back to the largest sampled budget for lanes with smaller grids.

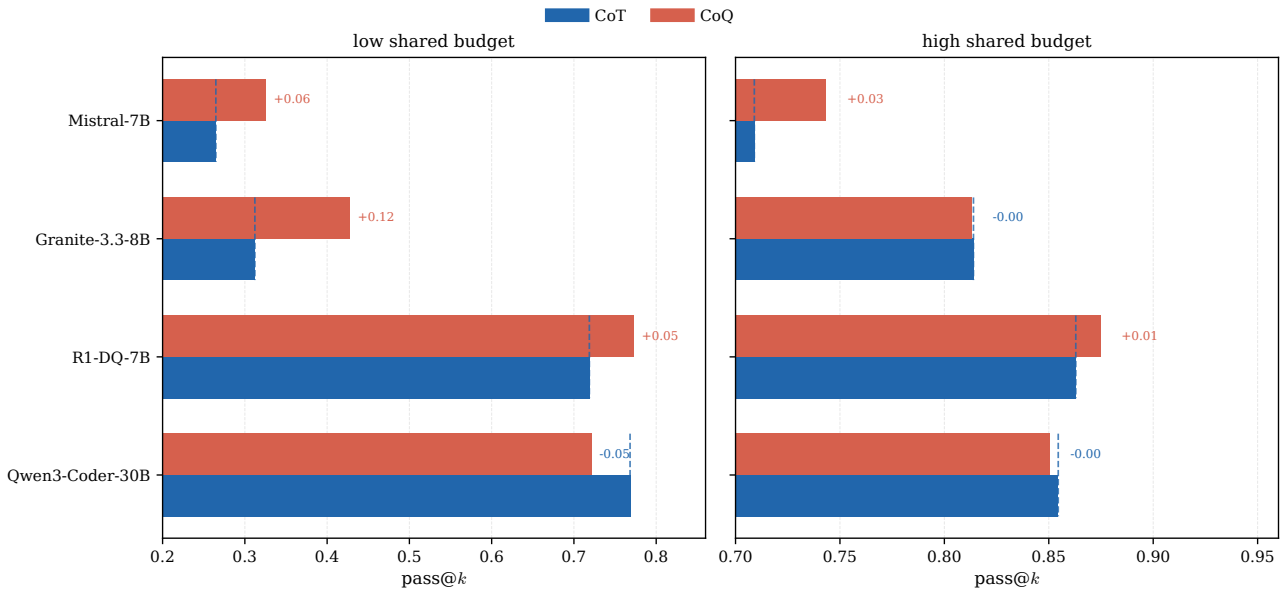

Figure 25. MBPP token-normalized bar chart bank. The low-budget and high-budget panels compare matched CoT and CoQ family means at paired completion-token budgets.

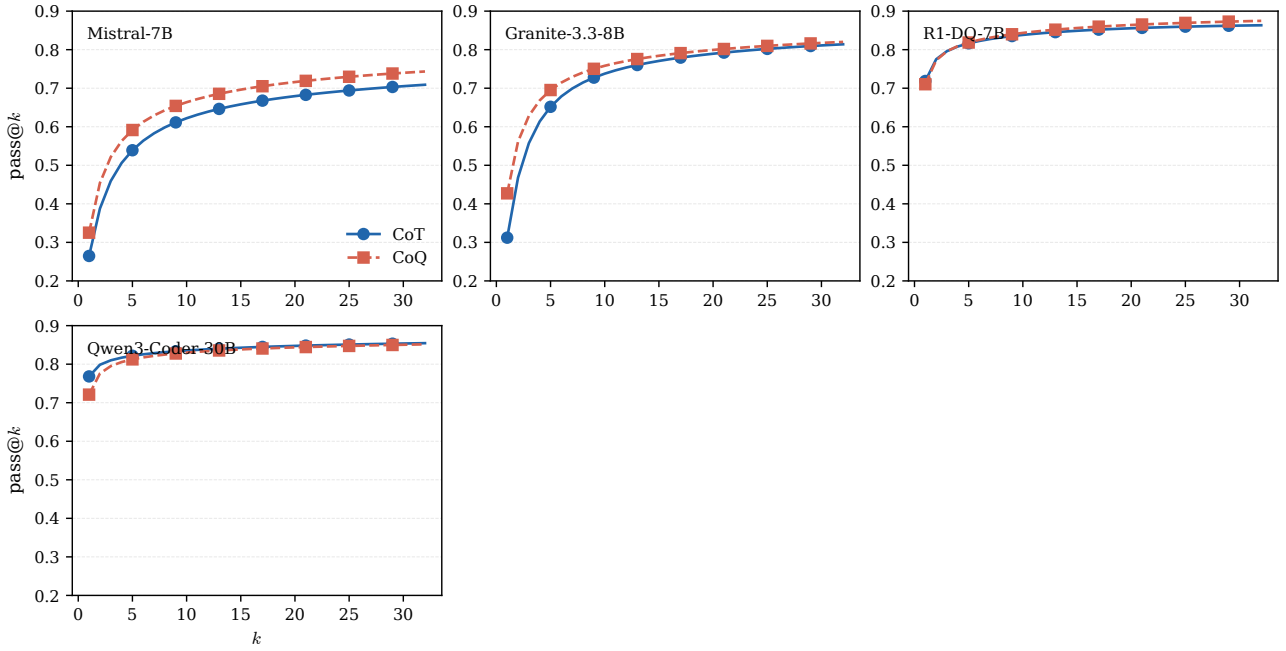

Figure 26. MBPP equal-sample pass@k curve bank. These panels preserve the full sample-budget trajectory used to choose compact main-text summaries.

Table 3. Main cross-benchmark synthesis. Entries are Δ pass@1 = CoQ family mean minus CoT family mean. A dagger marks cells where CoQ completions are more than $2\times$ as long as CoT on average. Missing cells are omitted from the current paper claim.

Family	Model	HumanEval	MATH	GPQA	AIME	MBPP	LCB	ChemIQ
instruct-small	Phi-4-mini	-0.01	-0.06	$+0.02$	–	–	–	$+0.00$
instruct-small	Mistral-7B	$+0.12$	-0.03	$+0.10$	–	$+0.06$	$+0.01$	$+0.00$
instruct-small	Qwen2-7B	-0.04	-0.07	$+0.06$	–	–	–	$+0.00^{\dagger}$
instruct-small	Granite-3.3-8B	$+0.23$	-0.01	$+0.07$	–	$+0.12$	$+0.00$	$+0.00$
instruct-small	Llama-3.1-8B	$+0.05$	-0.03	$+0.04$	–	–	–	$+0.00$
instruct-large	Qwen2.5-32B	-0.01	-0.05	$+0.04$	$+0.00$	-0.01	$+0.01$	$+0.00^{\dagger}$
instruct-large	Llama-3.3-70B	$+0.03$	$+0.04$	$+0.01$	–	-0.04	-0.15	$-0.01^{\dagger}$
reasoning	R1-DQ-7B	-0.01	$+0.01$	$+0.00$	$+0.00$	-0.01	–	$+0.01$
reasoning	QwQ-32B	-0.04	-0.02	$+0.00$	$+0.07$	$+0.00$	–	$+0.00$
reasoning	R1-DQ-32B	-0.01	$+0.01$	$+0.00$	$+0.05$	-0.01	–	$+0.01$
coder-small	Qwen2.5-Coder-7B	$+0.01$	–	–	–	-0.02	$+0.02$	–
coder-small	DeepSeek-Coder-V2-Lite	-0.02	–	–	–	-0.01	$+0.01$	–
coder-large	Qwen3-Coder-30B	-0.03	–	–	–	-0.05	–	–
closed	Gemini Flash-Lite	$+0.25$	–	–	$+0.01$	–	$+0.04$	–
closed	GPT-4o mini	$+0.13$	–	–	–	–	–	–

Table 4. Token-normalization caveats. We include cases where the high-budget equal-sample read and the highest paired equal-completion-token read disagree or materially change magnitude.

Model	Benchmark	Equal-sample read	Token-normalized read	Ratio
QwQ-32B	MATH	-0.01	$+0.10$	$0.95\times$
Qwen2.5-32B	ChemIQ	$+0.04$	-0.03	$4.02\times$
Llama-3.3-70B	ChemIQ	$+0.02$	-0.04	$2.48\times$
Qwen2-7B	ChemIQ	$+0.01$	-0.03	$2.84\times$
Llama-3.3-70B	LCB	-0.27	-0.24	$0.58\times$

C.5. MATH: Budget-Sensitive Math Results

MATH separates three regimes. Llama-3.3-70B is the positive case at the larger instruction scale, while Qwen2.5-32B does not improve under CoQ. Granite-3.3-8B and Llama-3.1-8B are not equal-sample wins, but improve in the token-normalized low-budget view. Reasoning models show the strongest CoQ effect at low sampling budgets rather than in the tail.

C.6. AIME: Selective Math Results

On AIME, the token-normalized improvements are concentrated on three models: Gemini Flash-Lite, Qwen2.5-32B, and R1-Distill-Qwen-7B. Smaller or incomplete lanes are omitted from the figure set.

C.7. GPQA-Diamond: Open-Model Science Results

On GPQA-Diamond, the CoQ effect is concentrated on the smaller open instruction models: Granite-3.3-8B, Llama-3.1-8B, Phi-4-mini, Mistral-7B, and Qwen2-7B all move positively at low k, while the reasoning-tuned models are near the benchmark ceiling.

Table 5. HumanEval cases under raw pass@1 and the highest paired completion-token budget. Entries use family means over matched CoT and CoQ prompt families.

Model	pass@1	equal-token endpoint	Ratio
GPT-4o mini	$0.622 \rightarrow 0.752$	$0.939 \rightarrow 0.943$	$1.22\times$
Llama-3.3-70B	$0.600 \rightarrow 0.632$	$0.894 \rightarrow 0.929$	$1.28\times$
Granite-3.3-8B	$0.210 \rightarrow 0.441$	$0.783 \rightarrow 0.862$	$1.30\times$
Llama-3.1-8B	$0.268 \rightarrow 0.321$	$0.864 \rightarrow 0.875$	$1.17\times$
Mistral-7B	$0.213 \rightarrow 0.333$	$0.750 \rightarrow 0.815$	$1.02\times$

C.8. HumanEval

On HumanEval, CoQ improves over CoT for the closed models and a subset of open instruction models reported here. Coder-large and reasoning-tuned lanes are not included in the figure set.

C.9. MBPP

MBPP shows positive, neutral, and negative cases on the same prompt matrix. We report a compact grid of the cases that most clearly distinguish them.

C.10. LiveCodeBench

On LiveCodeBench, CoQ improves at larger budgets for Gemini Flash-Lite, Qwen2.5-Coder-7B, Granite-3.3-8B, and Mistral-7B. We retain Llama-3.3-70B in the figures as the minor-failure case; reasoning-model lanes are excluded as poor fits for this task type.

C.11. ChemIQ

ChemIQ separates raw accuracy gains from token cost. At the larger instruction scale, CoQ improves under equal sample count but its completions are substantially longer; the token-normalized panels make the tradeoff explicit.

C.12. Latent Questions: Formalism and Connection to Latent Thoughts

This subsection develops the formalism behind the latent-question experiments introduced in Section 5.2, and locates them with respect to Ruan et al. (2025), which directly inspired the setup.

Latent-variable model. Let x be a problem, y its solution, and $q = (q_1, \ldots, q_m)$ a list of *epistemic* questions whose answers would close the reasoning gap from x to y. We posit the joint

$$p_\theta(y, q \mid x) \;=\; p_\theta(q \mid x)\, p_\theta(y \mid x, q), \tag{2}$$

so that the marginal likelihood of the solution under the model factors through the latent decomposition:

$$p_\theta(y \mid x) \;=\; \sum_q p_\theta(q \mid x)\, p_\theta(y \mid x, q). \tag{3}$$

The two factors split the work of reasoning into *asking* (what would help?) and *solving* (given what helps, produce the answer).

Training objective. A stronger teacher T post-hoc synthesizes a latent question list $q^\star \sim T(\cdot \mid x, y)$ from each solved source-task pair (x, y), producing a supervised dataset $D = \{(x_i, q_i^\star, y_i)\}$. The student θ is then trained on the two factors of (2) directly:

$$\mathcal{L}_{\text{lat}}(\theta) = -\mathbb{E}_{(x,q^\star,y^\star)\sim D}[\log p_\theta(q^\star \mid x) + \log p_\theta(y^\star \mid x, q^\star)]\,. \tag{4}$$

The first term trains the asker $p_\theta(q \mid x)$ to imitate the teacher's decomposition; the second trains the solver $p_\theta(y \mid x, q)$ to use a high-quality decomposition when one is provided. At test time, the student generates $\hat{q} \sim p_\theta(\cdot \mid x)$ and answers from $p_\theta(\cdot \mid x, \hat{q})$, on a *target* benchmark distinct from the source.

Connection to Ruan et al. (2025). Ruan et al. (2025) train language models from *latent thoughts*: free-form intermediate text z that is not observed in the data but is treated as a latent variable explaining unlabeled text x. Their model maximizes the marginal likelihood $\log p_\theta(x) = \log \sum_z p_\theta(z)p_\theta(x \mid z)$ via an EM-style bootstrap: an approximate posterior or the model itself proposes z, and the model is updated to fit the data conditioned on the proposed latents. The result is a single language model whose chain-of-thought sampling distribution improves with iteration.

Where we differ. The two setups share the latent-variable factorization in (2), and both treat reasoning as a structured object the model must learn to produce. Beyond that, the differences are load-bearing.

1. **Latent shape.** Ruan et al. (2025) place no structural prior on z: latent thoughts are free-form natural-language traces. Our latent q is constrained to a *list of epistemic questions*—specific, gap-targeting queries that name what is missing rather than narrate what is being thought. This constraint is what lets us judge $p_\theta(q \mid x)$ independently of final-task accuracy, route subquestions across model tiers, and elicit tacit knowledge from human experts in the co-scientist setting.

2. **Source of the latent.** They infer z via approximate posterior or self-bootstrap (the same model produces and consumes z). We *distill* $q^\star$ from a stronger asymmetric teacher with access to (x, y), sidestepping the posterior-collapse and bootstrap-instability concerns that EM-style training brings. The cost is a dependence on teacher quality, which we address separately through the oracle-answer ablation and routing diagnostics.

3. **Training data.** Their data is unlabeled text where the reasoning is unobserved and the objective is language modeling. Our data is solved problems where the *solution* is observed but the decomposition is not. The supervised pair $(x, q^\star, y^\star)$ lets us train both factors of (2) directly, rather than requiring marginalization over q.

4. **Evaluation.** They measure in-distribution language modeling and downstream task performance on the same data distribution. We measure *cross-domain transfer*: a student trained on source benchmark S is evaluated on a target benchmark T unrelated to S under standard prompting. This is a stricter criterion for the claim that $p_\theta(q \mid x)$ has acquired a portable competence rather than a benchmark-specific surface pattern.

5. **What the model is asked to internalize.** Their latent thoughts are intended to make the language modeler more capable on its native task. Our question-asking factor is intended to be *decoupled* from the solver: a trained asker should be useful when paired with any solver, not just with the one it was co-trained with. The downstream test of this is whether the asker improves a vanilla CoT prompt as well as a CoQ prompt on the target benchmark.

Reading the empirical results through the formalism. A positive transfer result—the asker $p_\theta(q \mid x)$ trained on S makes a *different* target-domain solver better—is evidence that the question factor has learned competence not specific to S. A null result under the same evaluation would mean either that the source has no portable decomposition skill to teach or that the teacher's $q^\star$ does not contain the right transferable structure. The empirical results in Section 5.2 support a narrower conclusion: latent question supervision can transfer across domains, but the transfer should be evaluated at the level of explicit source–target pairs rather than assumed to be universal.

C.13. Latent Questions: Strategy-Level Results

Table 6 gives the strategy-level rows behind Figure 4. The main figure shows only the best predeclared CoT and CoQ variants so the transfer effect is legible, but showing all rows matters: the deltas reported in §5.2 are best-in-family comparisons across the predeclared CoT and CoQ prompt variants, not best-of-many over hidden variants.

C.14. Oracle-Answer Ablation: Detailed Conditions

The oracle-answer ablation experiments decompose CoQ into asking, answering, and answer selection. In the base condition, the solver generates questions. In oracle-answer conditions, a stronger model answers all questions or only those that pass a quality judge. Matched random subsets and no-answer scaffolds separate answer value from the mere presence of a question list.

This experiment is crucial because a raw CoQ gain is ambiguous. It could mean that the solver asks better subquestions; it could mean that an external answerer injects missing facts; or it could mean that the question scaffold regularizes the

Table 6. **Strategy-level latent-question transfer results.** Accuracy is pass@1 on the target benchmark for the same transferred Qwen2-7B checkpoint. Bold entries mark the best strategy within each prompt family.

Source	Target	Family	Strategy	Correct / Total	Accuracy
AIME	ChemBench	CoT	v1	109 / 236	46.19%
AIME	ChemBench	CoT	v2	110 / 236	**46.61%**
AIME	ChemBench	CoT	v3	106 / 236	44.92%
AIME	ChemBench	CoQ	v1	104 / 236	44.07%
AIME	ChemBench	CoQ	v2	123 / 236	52.12%
AIME	ChemBench	CoQ	v3	136 / 236	**57.63%**
GPQA-Diamond	ChemIQ	CoT	v1	51 / 160	31.88%
GPQA-Diamond	ChemIQ	CoT	v2	63 / 160	**39.38%**
GPQA-Diamond	ChemIQ	CoT	v3	60 / 160	37.50%
GPQA-Diamond	ChemIQ	CoQ	v1	77 / 160	**48.12%**
GPQA-Diamond	ChemIQ	CoQ	v2	59 / 160	36.88%
GPQA-Diamond	ChemIQ	CoQ	v3	71 / 160	44.38%

final solution even without new information. The ablation therefore translates the paper's conceptual claim into measurable components.

When the diagnostic depends on question quality (the quality-filtered condition), the quality judge has to be vetted with the same care as the final-answer evaluator: a fragile JSON parser or an inconsistent rubric can substitute for a real signal in either direction.

D. Gap-Targeted Question Training

Motivation. Most of the paper treats CoQ as an inference interface: the model writes questions, optionally receives answers, and then solves the task. Adaptive question generation asks a different downstream question. Instead of asking whether a model can *use* a question scaffold at test time, we ask whether a model can be trained to identify a local reasoning gap and ask the question that would best repair it. This matters for the broader paper because it treats question asking not as prompt style, but as a trainable capability.

Task construction. The training examples are built from worked solutions. A solution is segmented into steps, a subset of steps is masked, and the model is shown the problem together with the resulting partial solution. The target is not the final answer but a short set of questions that would help recover the missing reasoning. We contrast *good* questions, which are intended to target the missing reasoning, with intentionally *bad* or low-value questions, which are plausible but generic, tangential, or premature. The contrastive setup is meant to supervise question quality rather than only surface form.

Why this is different from trace imitation. Latent-question transfer asks whether a model benefits from teacher-synthesized decompositions across domains. Gap-targeted question training is more local. It does not ask the student to reproduce an entire reasoning trace; instead, it asks the student to notice what information is missing *at a specific point* in a partial derivation and to formulate the next useful query. That makes it a cleaner match to the paper's central claim that explicit questions are useful when the bottleneck is identifying missing information rather than merely producing more tokens.

Training conditions. The current implemented family supports both supervised fine-tuning and a preference-style objective over good versus bad questions. The pilot reported here uses supervised fine-tuning only. The model is trained to map `problem + partial solution with masked steps` to a short list of gap-targeting questions. The present pilot uses a fixed masking curriculum rather than a fully adaptive one; in other words, the current evidence is best understood as a first gap-targeted-training pilot rather than the final form of the method.

Evaluation protocol. Our revised evaluation criterion follows the logic of the main paper. The key question is not whether a bespoke masked-step inference game can be solved, but whether the trained model becomes more useful for reasoning. We therefore treat two evaluations as primary:

1. **Question-quality evaluation:** given a masked partial solution, can the model generate questions that a stronger judge rates as relevant, specific, gap-targeting, and recovery-supporting?

2. **Reasoning transfer evaluation:** does the trained model help on downstream reasoning benchmarks either as a direct solver under a standard prompt or as a question generator inside a standard CoQ scaffold?

This framing keeps gap-targeted question training aligned with the rest of the paper: it is a way of improving question quality and reasoning behavior, not a separate toy task.

Pilot result. The pilot is centered on AIME-style masked-solution data and evaluates both question quality and downstream solving. On 80 held-out masked AIME examples, the adaptive-trained questioner modestly improves judged quality over the base Qwen2-7B questioner: average score $3.750/5$ versus $3.628/5$, gap-targeting $3.900/5$ versus $3.800/5$, recovery support $3.888/5$ versus $3.775/5$, and 61.25% versus 58.75% of examples rated good. More importantly, when inserted into the standard two-question CoQ scaffold, the trained questioner improves ChemBench pass@1 to 33.33%, compared with 25.00% for the same scaffold with the base questioner and 28.33% for base CoT with the same solver.

Role in the paper. Gap-targeted question training is the training-time companion to CoQ prompting at inference time: instead of asking whether explicit questions help when prompted, it asks whether a model can be trained to produce useful questions in the first place. The pilot supports the main-text claim in Section 5.2, while the appendix records the training construction and diagnostic details.

E. Question-Level Routing

Motivation. If CoQ exposes explicit subquestions, then those subquestions become units that can be budgeted. Some are easy enough that a cheap draft model should answer them locally; others should be escalated to a stronger tier. This turns explicit questioning into a routing problem. The per-question routing family studies whether question-structured reasoning gives us a better object for cost-aware allocation than ordinary CoT steps do.

Routing conditions. We compare six completed inference-only conditions:

- **COQ-ALL-DRAFT:** all subquestions are answered by the cheap draft model.
- **COQ-ALL-TIER4:** every subquestion is escalated to the strongest tier.
- **COQ-CONFIDENCE:** the draft model proposes answers, a judge checks consistency, and uncertain units are escalated.
- **COT-ALL-DRAFT:** the analogous draft-only baseline for CoT reasoning units.
- **COT-ALL-TIER4:** the analogous all-strong-tier baseline for CoT units.
- **COT-CONFIDENCE:** confidence-based routing over CoT units.

The learned-routing extension then trains the draft model itself to emit either a local answer marker or a route marker to a stronger tier. This is where the paper's question-centric framing matters most: if questions are a cleaner representation of missing information, then learned routing over CoQ units should be more effective than learned routing over raw CoT steps.

Table 7 summarizes the completed inference-only suite The pattern is deliberately mixed, which is scientifically useful. AIME is the most favorable benchmark for CoQ-style routing in the current suite: the strongest completed condition is `COQ-ALL-TIER4` at pass@1 $= 0.1667$, compared with 0.10 for the best completed CoT conditions. GPQA-Diamond currently favors CoT in the all-strong-tier setting, with `COT-ALL-TIER4` reaching pass@1 of 0.28.

Interpretation. The pattern is consistent with the body of the paper: explicit questions are most useful when the bottleneck is deciding what to resolve next. On AIME, the question decomposition picks out useful missing sub-tasks; on GPQA-Diamond, the inference-only result is less favorable to CoQ, which suggests that either the drafted questions are not yet the right routing units, or the tier stack and confidence policy are better matched to CoT-style derivations on this benchmark.

Table 7. Inference-only per-question routing conditions for the benchmarks reported here. The result is favorable to CoQ on AIME and unfavorable on GPQA-Diamond

Benchmark	Best CoQ condition	Best CoQ pass@1	Best CoT pass@1
AIME	`COQ-ALL-TIER4`	0.1667	0.1000
GPQA-Diamond	`COQ-ALL-TIER4` / `COQ-CONFIDENCE`	0.1800	0.2800

Toward learned routing (future work). The inference-only result here uses a confidence-based router. A natural extension is to train the draft model itself to emit either a local answer or a route marker such as `[ROUTE:Tier4]`, with a route-cost penalty λ that biases the policy toward cheaper tiers unless the data justify escalation. Route-supervision data can be collected by labeling each unit with the cheapest tier whose answer matches an oracle-quality reference. We do not report learned-routing results in this paper; the inference-only confidence routing in Section 5.3 establishes that question structure is a useful unit for cost-aware allocation, and we leave the learned-router extension to future work.

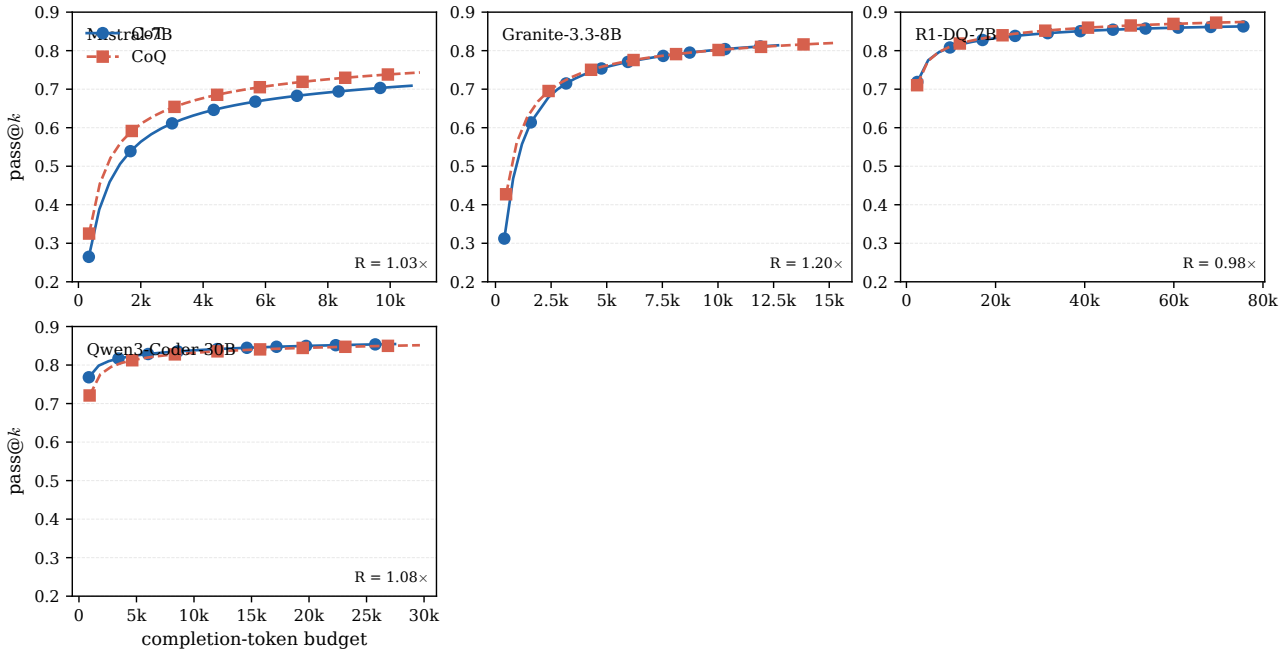

Figure 27. MBPP completion-token-normalized curve bank. These panels show how the CoT–CoQ comparison evolves as the shared completion-token budget grows.

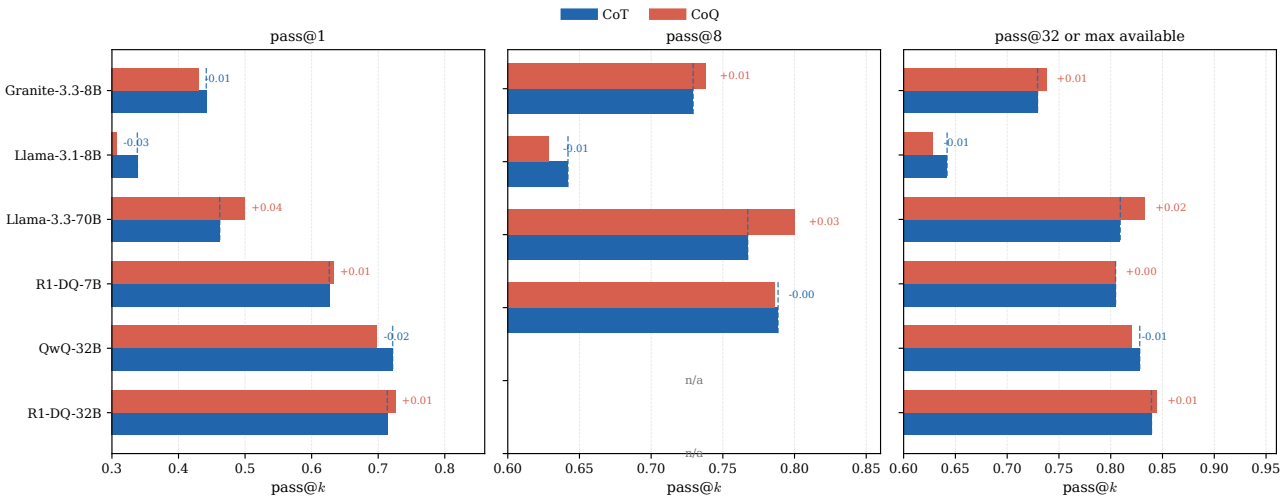

Figure 28. MATH equal-sample bar chart bank. Panels report pass@1, pass@8, and the largest available sampled budget up to pass@32. Llama-3.3-70B is the positive case at the larger instruction scale, Qwen2.5-32B is the same-scale counterexample, and reasoning models concentrate their CoQ effect at low k.

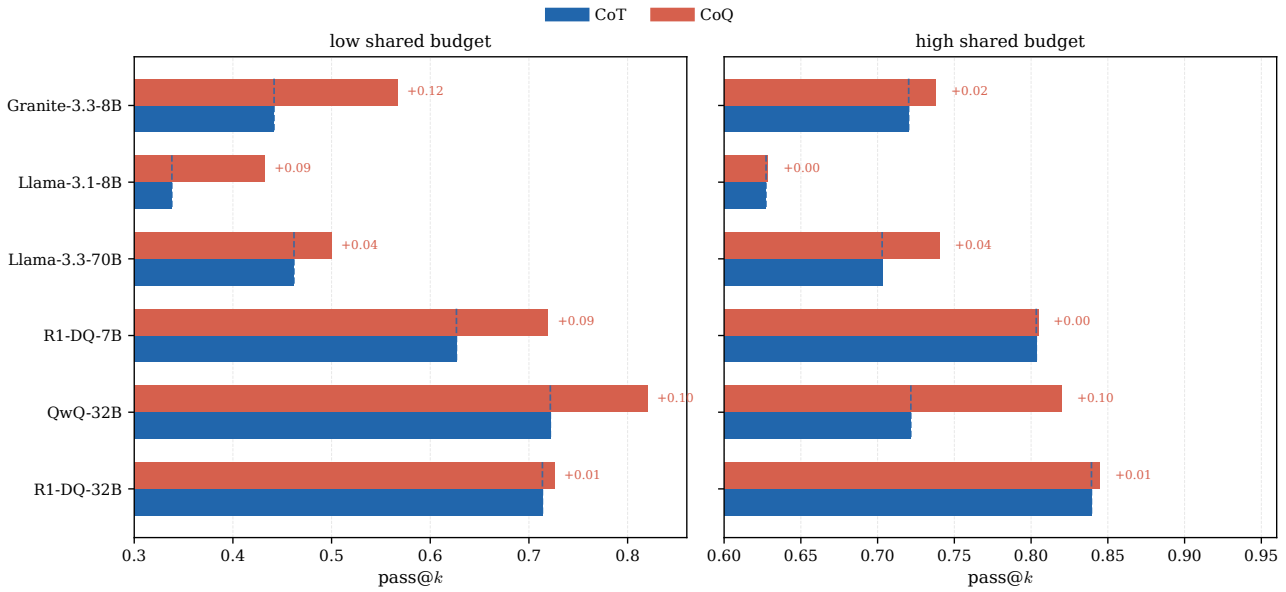

Figure 29. MATH token-normalized bar chart bank. The low-budget panel highlights where Granite-3.3-8B, Llama-3.1-8B, and the reasoning models look most favorable to CoQ; the high-budget panel shows which gains persist at the largest paired completion-token budget.

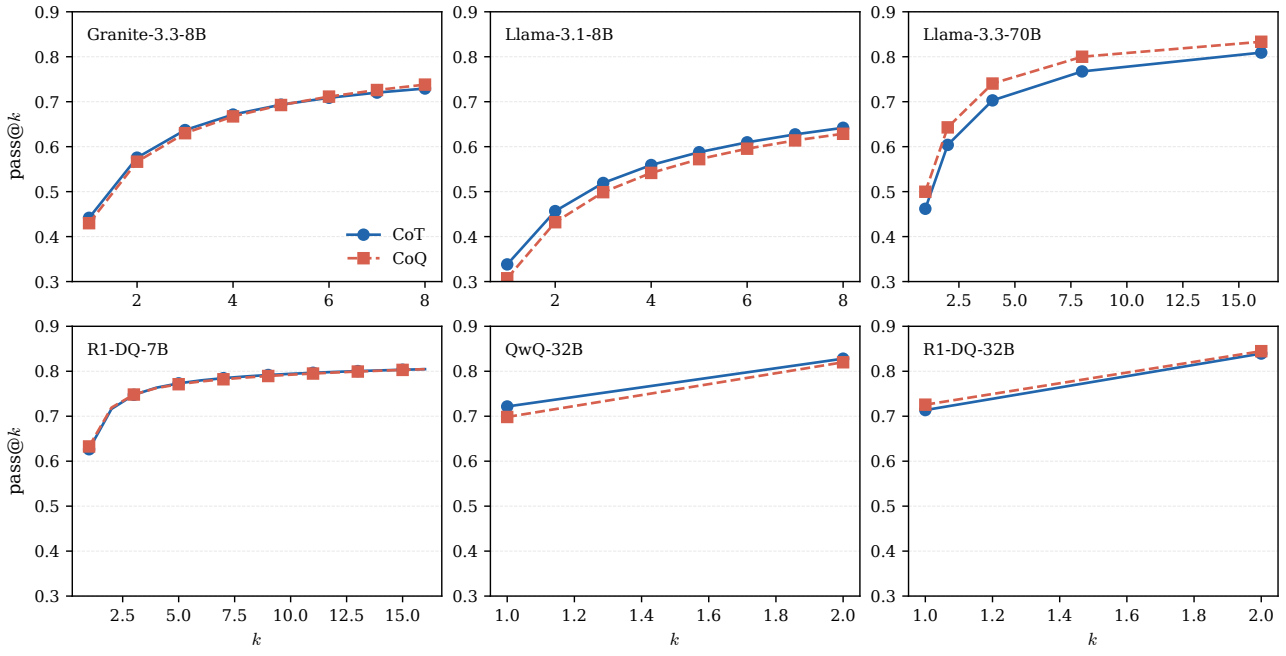

Figure 30. MATH equal-sample pass@k curves. These keep the larger-instruction contrast, the mostly non-positive smaller-instruction cases, and the low-budget reasoning-model behavior visible in one view.

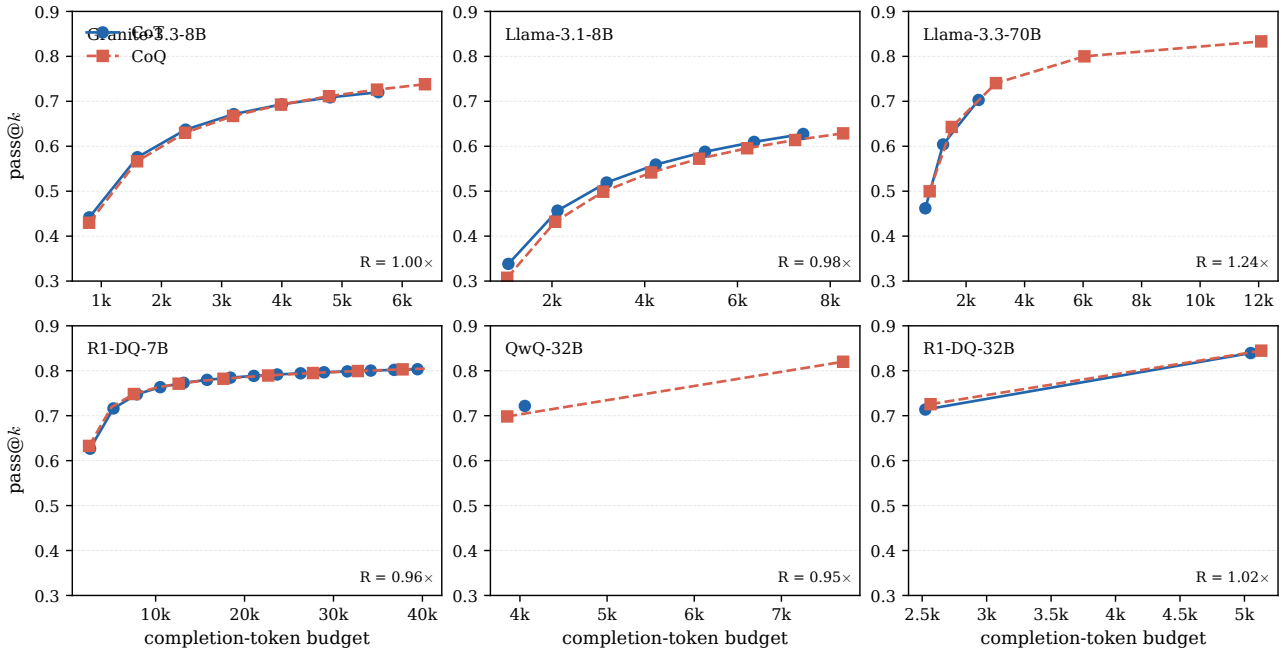

Figure 31. MATH token-normalized pass@k curves as completion budget increases. This view separates raw extra-sample benefits from cases where CoQ remains competitive under a matched completion-token budget.

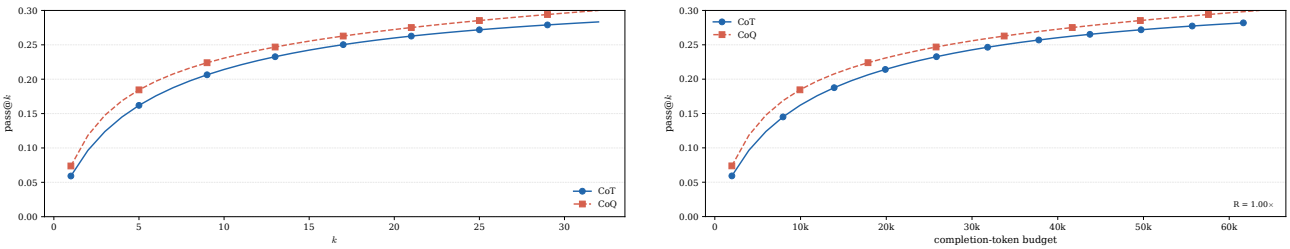

Figure 32. AIME, Gemini Flash-Lite. The token-normalized view is the one used for the reported result; smaller-scale lanes are omitted from the figure set.

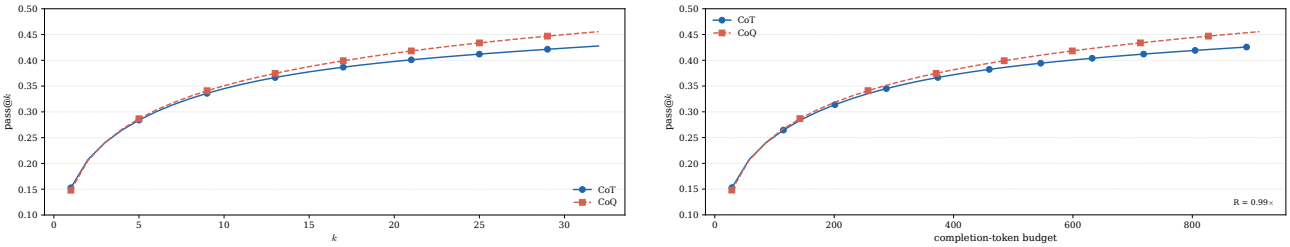

Figure 33. AIME, Qwen2.5-32B. CoQ is not better at pass@1, but improves at larger sampled budgets, and the advantage persists under equal-token-budget because this lane has near-equal average completion length for CoT and CoQ.

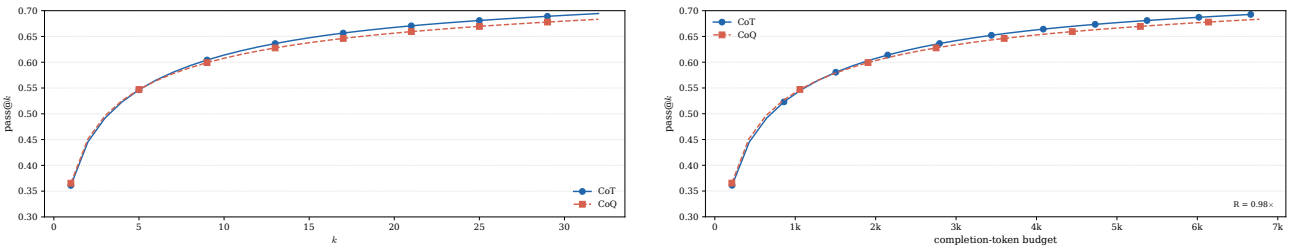

Figure 34. AIME, R1-Distill-Qwen-7B. The CoQ effect is at the low-token end rather than the high-budget tail; larger reasoning-model greedy-only lanes are omitted from the figure set.

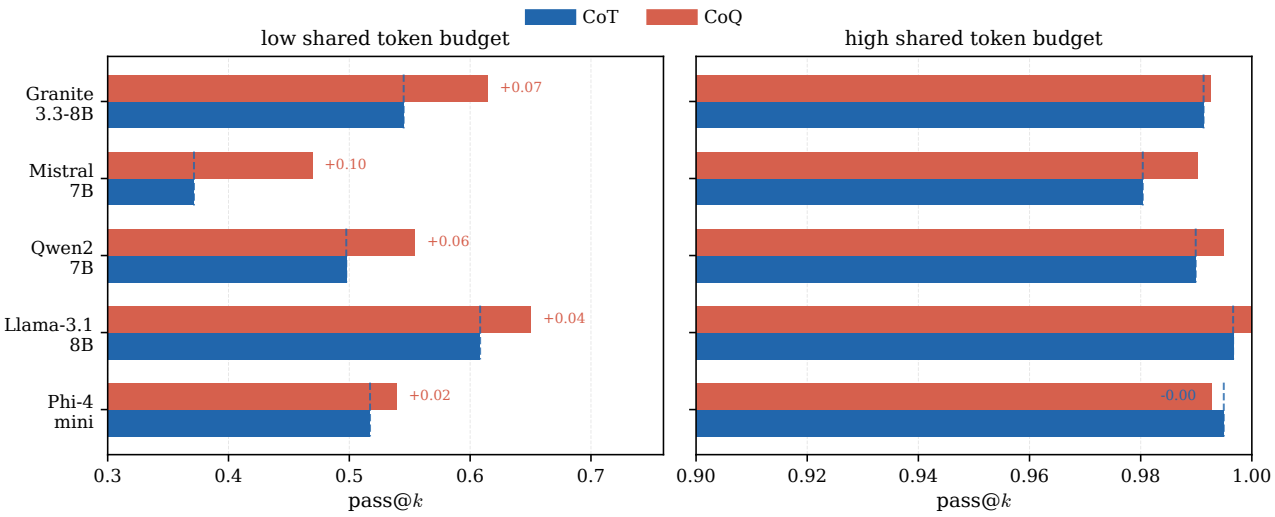

Figure 35. GPQA-Diamond token-normalized comparison for the five models that favor CoQ. The left panel is the low-token regime where all five improve; the right panel is the high-budget regime, where the advantage mostly shrinks as models approach ceiling.

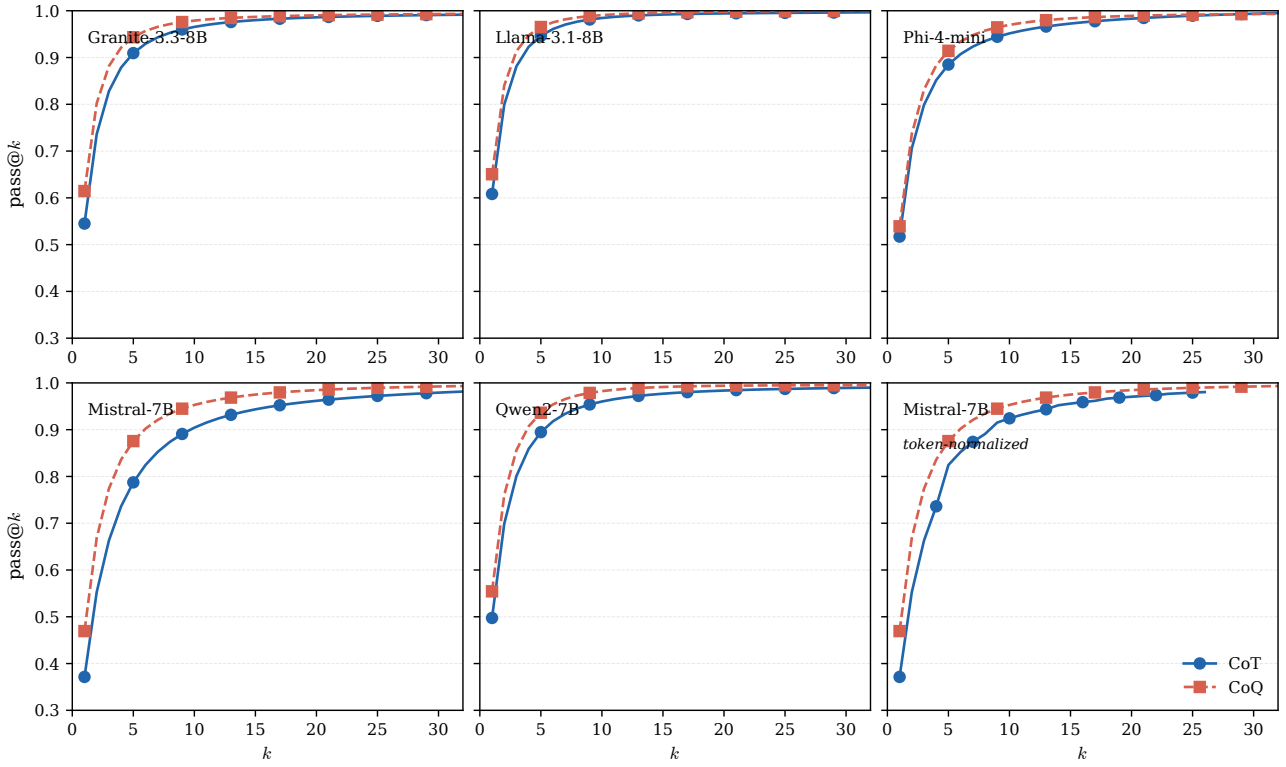

Figure 36. GPQA-Diamond. The first five panels show equal-sample pass@k for Granite-3.3-8B, Llama-3.1-8B, Phi-4-mini, Mistral-7B, and Qwen2-7B. The final panel shows Mistral-7B under token normalization, where the gain is smaller than in the raw bootstrap view.

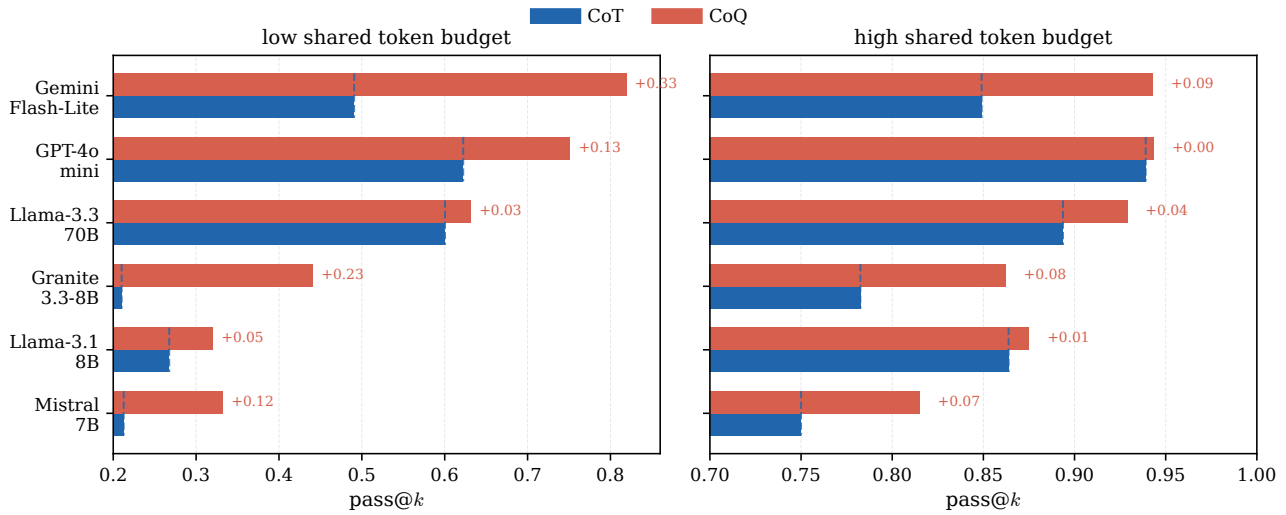

Figure 37. HumanEval token-normalized family means for selected positive cases. The left panel is the first paired equal-completion-token budget and the right panel is the largest paired equal-completion-token budget. Dashed vertical segments mark the CoT family mean; CoQ deltas are annotated beside the CoQ bars.

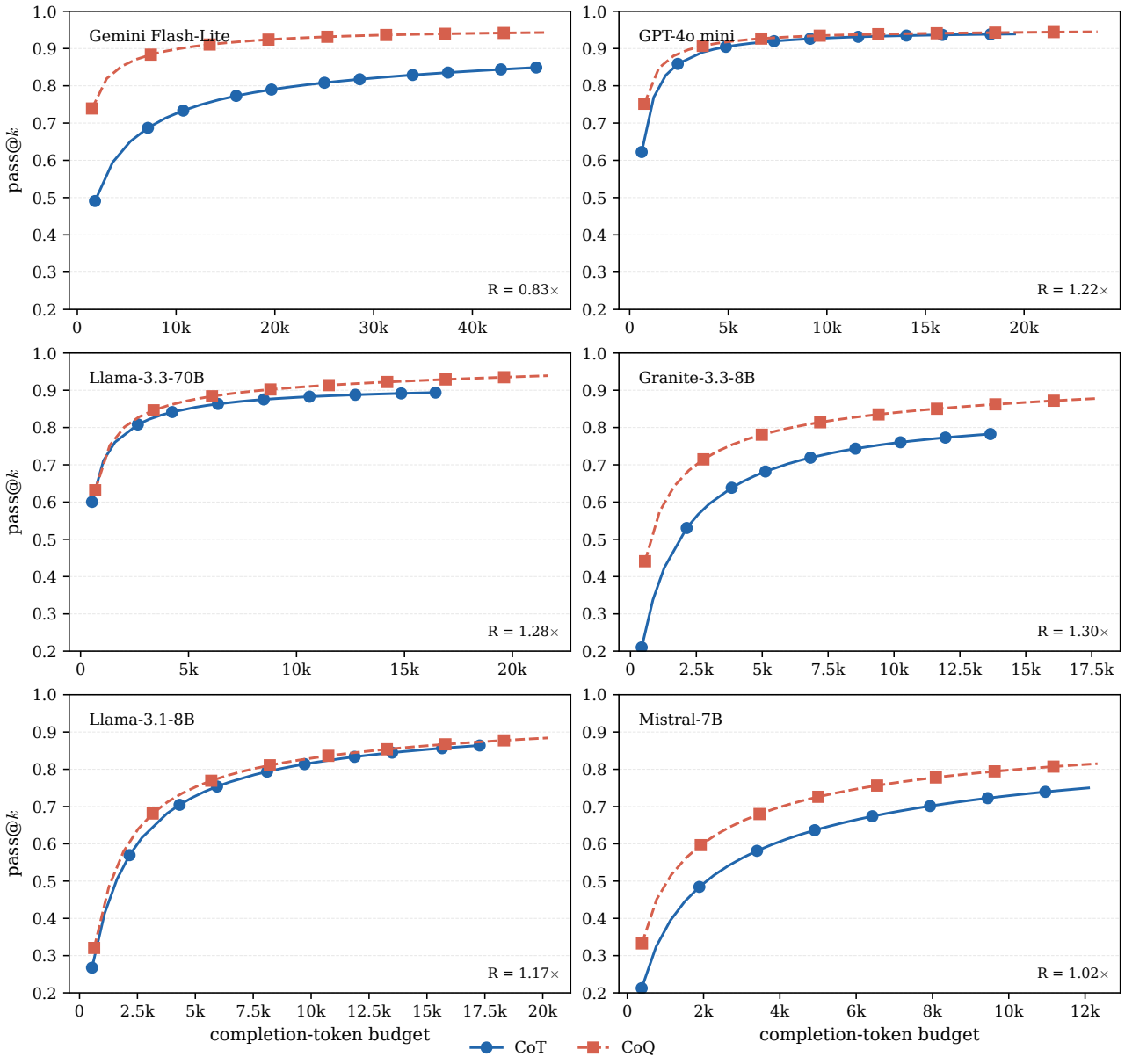

Figure 38. HumanEval token-normalized comparisons for the models reporting positive CoQ effects: the closed models Gemini Flash-Lite and GPT-4o mini, the larger open instruction model Llama-3.3-70B, and the smaller open instruction models Granite-3.3-8B, Llama-3.1-8B, and Mistral-7B.

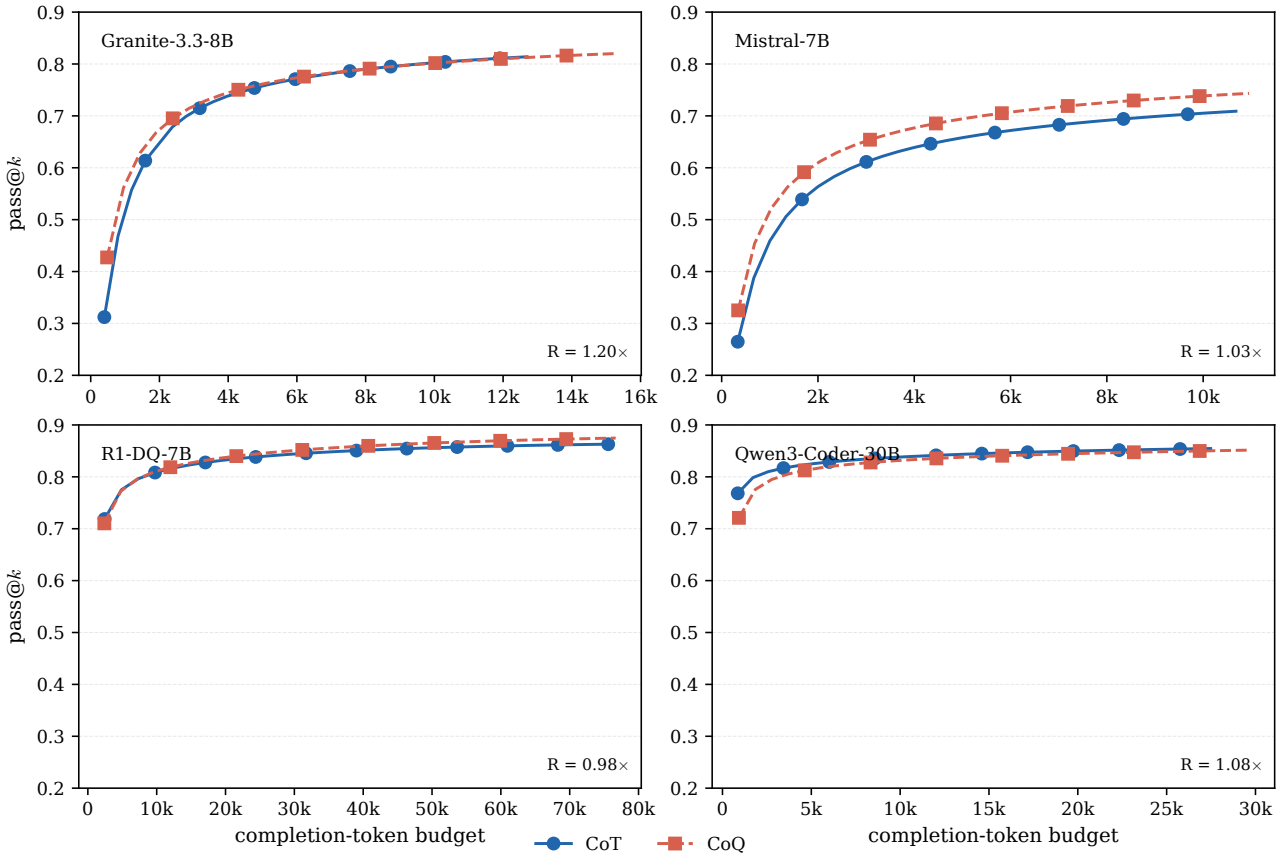

Figure 39. MBPP token-normalized selected cases. Granite-3.3-8B, Mistral-7B, and R1-Distill-Qwen-7B improve under CoQ; Qwen3-Coder-30B is the negative contrast.

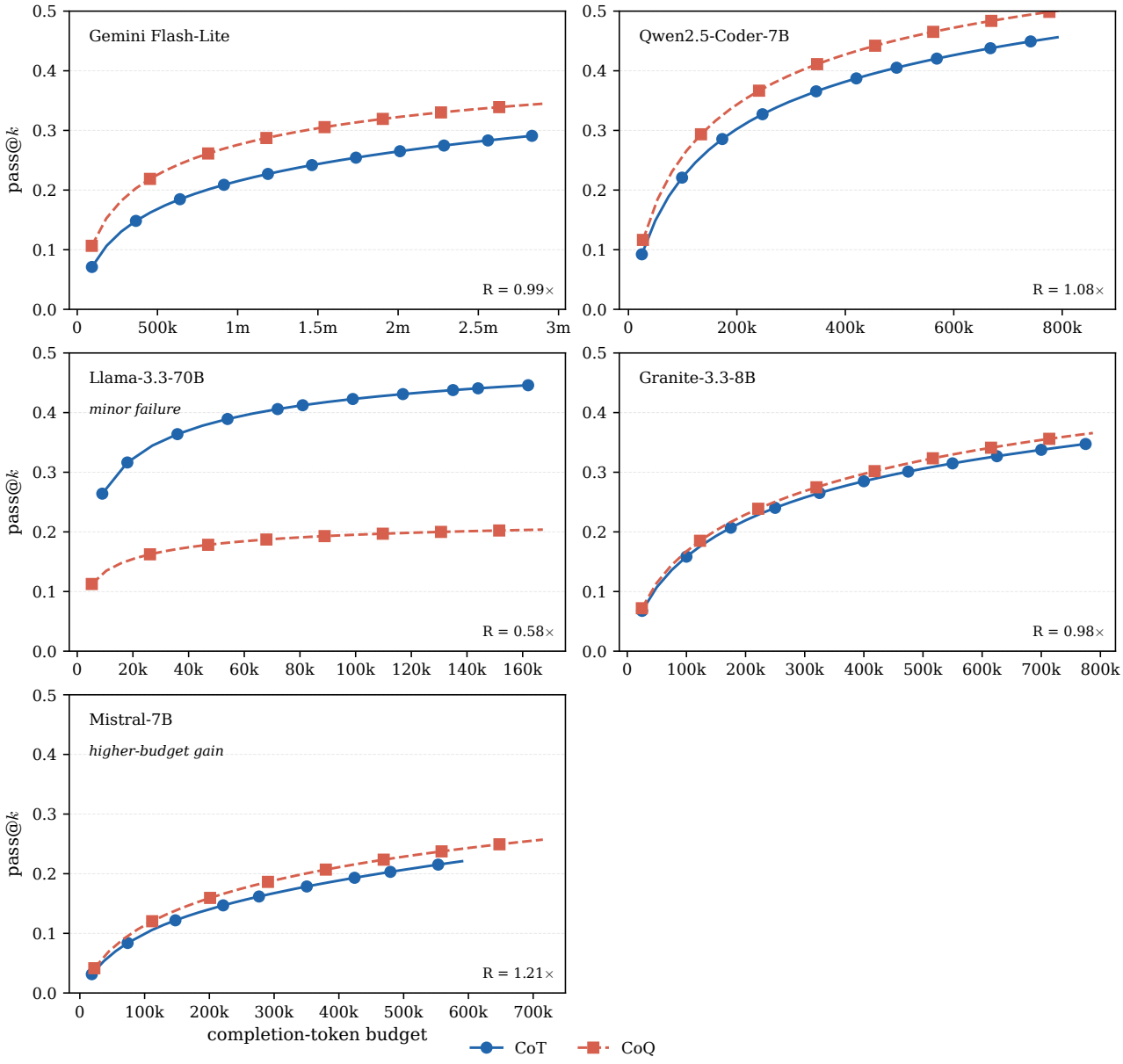

Figure 40. LiveCodeBench token-normalized selected cases. Gemini Flash-Lite is the closed-model positive; Qwen2.5-Coder-7B is the coder-small positive; Granite-3.3-8B improves; Mistral-7B improves at larger budgets; Llama-3.3-70B is included as the minor-failure contrast.

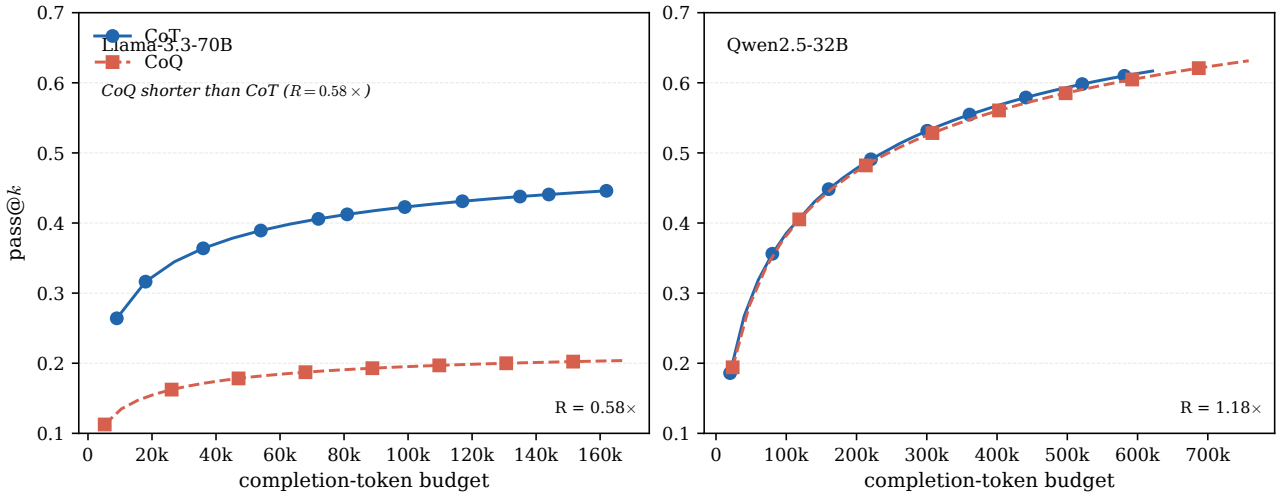

Figure 41. LiveCodeBench at the larger instruction scale. Llama-3.3-70B does not improve under CoQ after token normalization; Qwen2.5-32B is neutral despite a small raw pass@k advantage.

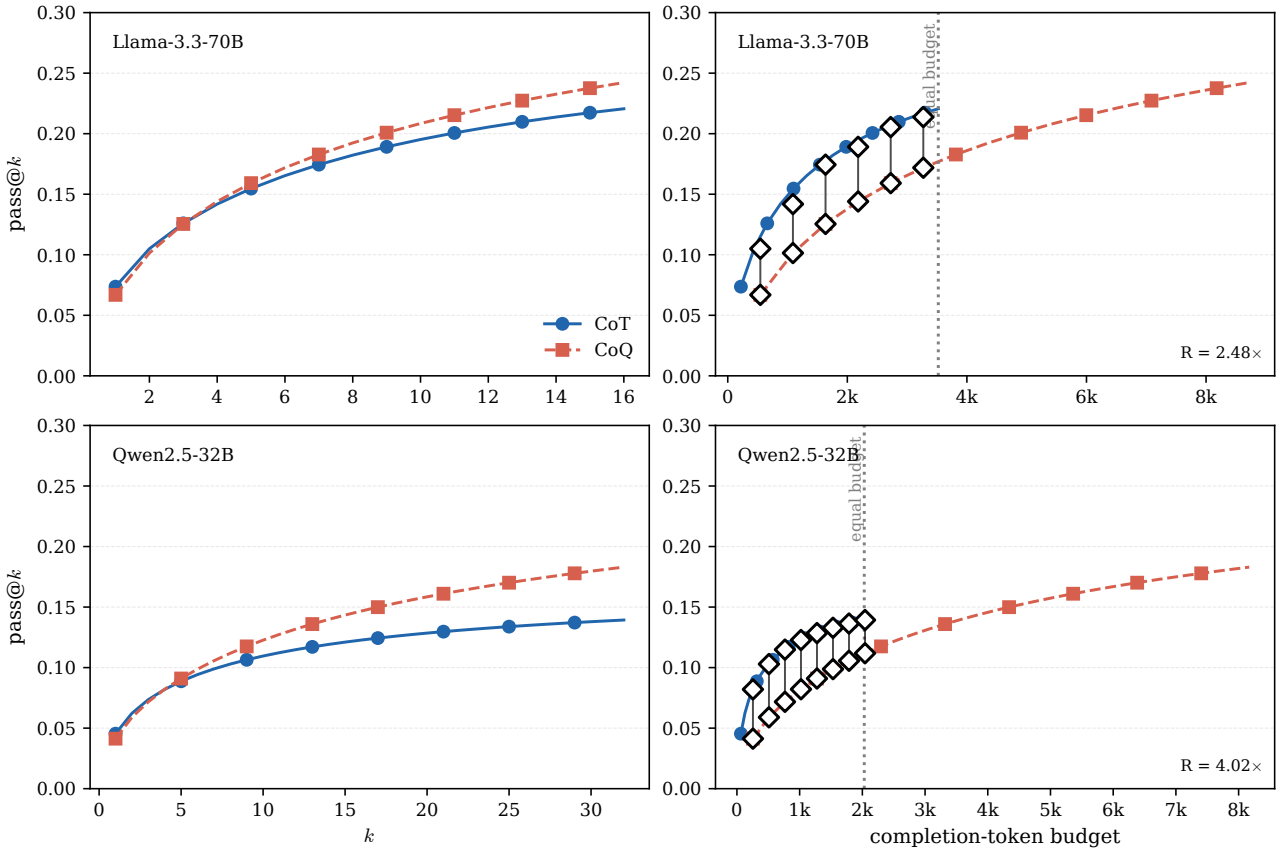

Figure 42. ChemIQ at the larger instruction scale. Rows are Llama-3.3-70B and Qwen2.5-32B; columns are equal-sample pass@k and completion-token-normalized pass@k. CoQ improves under equal samples but pays a substantial token cost, especially for Qwen2.5-32B.