BENCHMARKING THE QUANTUM LEAP: AUDITING LLM EVALUATION FOR QUANTUM CODE GENERATION

Anonymous authors
Paper under double-blind review

ABSTRACT

Large language models (LLMs) are increasingly capable of generating quantum programs, but reliable evaluation remains difficult: benchmarks differ in task composition, scoring procedures, and failure handling, while leaderboard rankings can obscure uncertainty and evaluator defects. We evaluate 15 hosted frontier, open-weight, quantum-specialized, and retrieval-augmented models across five primary quantum code-generation suites containing 530 tasks, and 10 models on QuantumBenchEval (QBE), a new benchmark of 96 tasks across six quantum-computing topics that we introduce. We report task-level bootstrap confidence intervals, multiplicity-adjusted paired comparisons, and audits of evaluator behavior, output validity, and shared failures. QiskitHumanEval-Standard is approaching saturation for the strongest models (98.7% pass@1), whereas its Hard variant retains substantial headroom (63.8%); across the recorded primary-suite outcomes, none of the five top-two comparisons remains significant after multiple-comparison correction. Evaluation design can have an even larger effect: removing checks on quantities the prompts never specified in two QuantumBenchEval topics raises GPT-6 Astra's pass@1 from 0% to 94.1% (T1) and from 1.2% to 92.9% (T6) without changing its outputs. Our audit also identifies 17 QuanBench-117 tasks with shared failures associated with benchmark-owned imports, while controlled probes of the current evaluator reveal unexecuted unit tests, leakage of canonical functions into candidate execution, and reference imports that can block otherwise valid candidates. These defects can affect both apparent failures and apparent passes, requiring historical QuanBench outcomes to be revalidated under an isolated evaluator. Together, these results show that quantum code-generation progress cannot be characterized by leaderboard scores alone. We release the evaluation harness, per-sample outcomes, and audit artifacts, and advocate protocols that validate scoring and test execution, isolate candidate and reference code, report uncertainty and output validity, and respect sampling assumptions when estimating pass@k.

1 INTRODUCTION

A programmer asked to prepare a four-qubit W state, or to decompose a CNOT into iSWAP gates, must reason about circuit semantics, measurement statistics, and backend constraints in addition to ordinary concerns of API usage and correctness (Riofrío et al., 2025). Large language models (LLMs) have become capable code generators (Chen et al., 2021), and several systems now target this setting directly, including Qiskit Code Assistant (Dupuis et al., 2024), QSpark (Kheiri et al., 2025), PennyCoder (Basit et al., 2025b), and Agent-Q (Jern et al., 2025). A parallel line of work has produced benchmarks for measuring them, from Qiskit HumanEval (Vishwakarma et al., 2024) and QCircuitNet (Yang et al., 2024) to QHackBench (Basit et al., 2025c), QCoder (Mikuriya et al., 2025), and QuanBench (Guo et al., 2025).

These benchmarks do not agree on what to measure or how. They differ in task composition, execution environment, and output format. Some grade by executable unit tests; others require a model-based judge or a custom scoring function. Three consequences follow for anyone comparing models across them. First, a suite may lose discriminative power as models improve, so that a leaderboard position no longer distinguishes capability. Second, a difference between two models may be smaller than the sampling noise of a few hundred tasks, so that leaderboard order and statistical evidence for

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

separation come apart. Third, a benchmark score depends on the reference solutions, test harness, dependencies, and evaluator assumptions as much as on the generated program, so that a defect in any of these components is misread as a failure, or a success, of the model. Systematic analysis of evaluation validity of this kind exists for classical code generation (Chen et al., 2021) but not, to our knowledge, for quantum code generation.

In this paper, we ask two questions: how well do current LLMs generate quantum code, and how reliably do existing protocols measure that ability? We evaluate 15 models spanning hosted frontier, open-weight, quantum-specialized, and retrieval-augmented systems on five established suites comprising 530 tasks. We also introduce QuantumBenchEval (QBE), a new 96-task benchmark spanning six quantum-computing topics, on which we evaluate 10 models. Because its tasks are newly constructed, they are less likely than long-standing public suites to appear in training data. Across all suites we report uncertainty, statistical separability, shared failures, output validity, and evaluator behavior rather than point estimates alone.

Our contributions are as follows.

- **Unified evaluation.** We build a single harness covering five suites, 530 tasks, and 15 models, and report pass@k with task-level bootstrap confidence intervals, paired comparisons with Holm correction, and bootstrap rank intervals.
- **Saturation and separability.** QiskitHumanEval-Standard is saturating for the strongest models (98.7% pass@1), whereas its Hard variant retains headroom (63.8%). On each of the five primary suites, the top-two model comparison loses significance after multiplicity correction.
- **Evaluator effects.** Removing checks on quantities the prompts never specified in two QuantumBenchEval topics raises GPT-6 Astra from 0% to 94.1% (T1) and from 1.2% to 92.9% (T6) on unchanged outputs. An audit of the QuanBench evaluator identifies 17 QuanBench-117 tasks whose shared failures trace to benchmark-owned imports, and controlled probes show that unit-test cases can go unexecuted, canonical functions can leak into candidate execution, and reference-code imports can block otherwise valid candidates. These defects affect apparent passes as well as apparent failures, so historical QuanBench results require revalidation under an isolated evaluator.
- **Living benchmark.** We release Quantum Arena, a public leaderboard consolidating tasks, per-sample outcomes, and audit artifacts, together with the harness itself, so that new models and tasks can be evaluated under the same protocol after publication.

2 Related Work

2.1 Quantum Code Generation with Large Language Models

Qiskit Code Assistant (Dupuis et al., 2024) was among the first specialized models, trained on the Granite architecture to generate Qiskit code; Dupuis et al. (2025) subsequently introduced quantum verifiable rewards for post-training using circuit-simulation feedback. QSpark (Kheiri et al., 2025) fine-tuned Qwen2.5-Coder with group relative policy optimization, PennyCoder (Basit et al., 2025b) showed that PennyLane-centric training improves generation quality, and Agent-Q (Jern et al., 2025) explored multi-agent iterative circuit refinement. Dataset work includes QCircuitNet (Yang et al., 2024), KetGPT (Apak et al., 2024), and C4Q (Aragonés-Soria & Oriol, 2024).

2.2 Benchmarks for Quantum Code Generation

Qiskit HumanEval (Vishwakarma et al., 2024) adapts HumanEval (Chen et al., 2021) to 151 tasks on Qiskit API usage and basic algorithms. QCoder (Mikuriya et al., 2025) introduces 67 hardware-aware tasks with constraints such as qubit connectivity and restricted gate sets. QHackBench (Basit et al., 2025c) collects PennyLane hackathon challenges, PennyLang (Basit et al., 2025a) offers 264 test cases across PennyLane features, and QuanBench (Guo et al., 2025) targets core algorithm implementations such as Grover's algorithm, the quantum Fourier transform, and VQE.

Dataset	# Tasks	Difficulty	Eval Type	Tests?
Qiskit (Standard)	151	Standard	Execution	✓
Qiskit (Hard)	151	Hard	Execution	✓
QuanBench-44	44	Medium	Execution	✓
QuanBench-117	117	Med–Hard	Execution	✓
QCoder	67	Mixed	LLM Judge	✗
QuantumBenchEval (QBE)	96	Mixed	Execution + LLM Judge	✓

Table 1: Summary of quantum code generation benchmarks. Execution-based datasets use unit tests, while QCoder relies on LLM-based evaluation. QBenchEval is execution-graded except for its VQE topic (T3, 15 tasks), which is scored by an LLM rubric judge.

2.3 Evaluation Validity

A benchmark score measures a model only if the evaluator measures the intended capability, and a growing body of work shows that it often does not. Liu et al. (2023) found that HumanEval's unit tests are too sparse to catch many incorrect programs; augmenting them substantially lowers pass@k and reorders models. Benchmark-side defects also cause false failures: SWE-bench (Jimenez et al., 2023) was refined into a human-validated Verified subset after annotators found tasks whose issue descriptions were underspecified or whose tests rejected valid solutions. Contamination is a third threat, which LiveCodeBench (Jain et al., 2025) and MathArena (Balunovic et al., 2026) address by evaluating on problems released after model training; MathArena also reports inflated performance on older, widely circulated competitions. Where no executable oracle exists, LLM judges are a common substitute, but Zheng et al. (2023) document position, verbosity, and self-enhancement biases in such judges. Finally, small benchmarks call for explicit uncertainty: Miller (2024) argues for task-level confidence intervals and paired comparisons in place of raw leaderboard ordering. The quantum code-generation benchmarks above report single-suite leaderboards without auditing their evaluators. We apply these validity concerns to quantum code generation, where each one appears: tests that never execute and reference-side imports that block valid candidates (§4.5), scoring contracts that check undisclosed quantities (§4.4), a judge-scored suite with a non-i.i.d. protocol (§4.6), and top-model differences that do not survive multiplicity correction (§4.2).

3 Methods

3.1 Benchmarking Framework Overview

Our harness samples tasks from each suite, generates $n{=}5$ completions per task with a shared prompt template and no reranking, voting, or post-processing, and scores them with each suite's native evaluator (unit tests, judge, or task-specific scorer), logging every program and outcome. We use temperature 0.8 where the provider exposes it. Decoding is not uniform across providers: most models use temperature 0.8, Claude Fable 5.1 and Muse-Glimmer-30B use 1.0, and QCoder output budgets range from 2,048 to 32,768 tokens (Appendix A.2). We therefore report output validity alongside pass@k and treat cross-model gaps as measured under these configurations.

3.2 Datasets

We evaluate models across five quantum programming benchmarks spanning two evaluation paradigms: execution-based correctness and LLM-based judgment.

QiskitHumanEval (Standard, 151 tasks) is a quantum adaptation of the HumanEval benchmark, consisting of short programming problems covering core quantum computing primitives such as state preparation, circuit construction, and measurement. Correctness is determined via unit tests.

QiskitHumanEval (Hard, 151 tasks) is a difficulty-enhanced variant of the standard benchmark, requiring deeper compositional reasoning and multi-step circuit design.

QuanBench-117 (117 tasks) is a large-scale benchmark designed to cover a broad spectrum of quantum programming concepts, emphasizing generalization across diverse tasks.

QuanBench-44 (44 tasks) is a smaller subset of QuanBench that enables controlled comparison under reduced task diversity.

QCoder (67 tasks) consists of open-ended quantum programming problems without canonical solutions or executable test cases. These tasks emphasize end-to-end reasoning and algorithmic design rather than test-driven correctness.

QuantumBenchEval (QBE, 96 tasks) is a new benchmark we introduce, spanning six quantum-computing topics with 15–17 tasks each (Table 6). Five topics are scored by executing the candidate against task-specific assertions; T3 combines execution with a rubric (below).

Unlike classical code benchmarks, correctness in our setting requires alignment with quantum state behavior, including measurement distributions and circuit semantics, making evaluation fundamentally more challenging than traditional unit-test-based code generation. Table 1 summarizes the datasets used in our evaluation.

Examples of a task from each dataset can be found in Appendix B.1.

3.3 Models

We evaluate 15 models in the primary benchmark collection, grouped by specialization and deployment setting.

Hosted general-purpose models. These comprise GPT-5 via Azure OpenAI (Singh et al., 2025), GPT-6 Astra (OpenAI, 2026), Claude Opus 4.6 (Anthropic, 2026b), Claude Fable 5.1 (Anthropic, 2026a), Gemini 3 Flash (Google DeepMind, 2025), Gemini 3.6 Flash (Google, 2026), and Gemini 3.1 Pro Preview (Google DeepMind, 2026).

General-purpose open-weight models. We include LLaMA-3.1-8B (Grattafiori et al., 2024), Mistral-7B-Instruct-v0.3 (Jiang et al., 2023), Gemma-3-4B-IT (Team et al., 2025), and Muse-Glimmer-30B (Lu et al., 2026). LLaMA-3.1-8B is the base, non-instruction-tuned checkpoint.

Quantum-specialized models. We evaluate Qwen2.5-Coder-14B-Qiskit (IBM Quantum, 2025c), Mistral Small 3.2 24B Qiskit (IBM Quantum, 2025b), and Granite 3.2 8B Qiskit (IBM Quantum, 2025a; Dupuis et al., 2024).

Retrieval-augmented system. We additionally evaluate Quantum RAG (Stapleton et al., 2026), which combines retrieval with a quantum-code-generation model.

3.4 Evaluation Protocol

Pass@k and uncertainty. For a task with $n{=}5$ samples and c successes we use the unbiased estimator $\widehat{\text{pass@}k} = 1 - \binom{n-c}{k} / \binom{n}{k}$, averaged over tasks, reporting $k \in \{1, 3, 5\}$ for the primary suites and $k \in \{1, 5\}$ for assertion-scored QBE topics. We give pointwise 95% intervals from 10,000 task-bootstrap replicates, paired task-level model comparisons with Holm correction within each suite (105 pairs; 78 for QuanBench-44 and QCoder; Appendix A.4), and Spearman and Kendall τ_b rank agreement across suites. Intervals condition on the recorded samples and evaluators and do not account for unresolved scoring defects; affected outcomes are marked as requiring revalidation.

Judge-scored evaluation. QCoder provides no executable tests, so each candidate receives a binary PASS/FAIL verdict from an LLM judge (per-run judge identities in Appendix A.2). Claude Opus 4.6 is excluded from the QCoder roster to avoid self-evaluation; Claude Fable 5.1 shares a provider with the QCoder and T3 judges and was retained, so its judge-scored results are flagged given documented self-preference effects (Zheng et al., 2023).

QBE T3: open-ended energy estimation. T3 contains 15 tasks requiring Qiskit implementations of variational ground-state energy estimation for molecular and lattice systems. Because multiple implementations can be valid, T3 has no binary outcome. Each judged sample is scored on a fixed 10-point rubric: energy accuracy (3), chemical accuracy (1), required outputs (2), qubit and circuit

specification (2), and scientific reasoning (2). Candidate code runs in an isolated subprocess; energy accuracy is recomputed deterministically from the executed energy against a task-specific reference, and the remaining criteria are scored by Claude Sonnet 4.6 from the candidate code and execution result.

Diagnostics. We report valid-output rates (nonempty and syntactically valid, with total-sample denominators), rule-based classifications of saved evaluation errors, and observed task coverage (empirical pass@5) with pairwise complementarity. Union coverage is an oracle quantity that uses additional generation budget, not a deployable selector. We also compute CodeBLEU (Ren et al., 2020) and its per-sample point-biserial correlation with pass/fail. None of these diagnostics establishes functional or quantum-semantic correctness. Error-label definitions, CodeBLEU scoring details, T1 resource closeness, token usage, and incomplete-run bounds are given in Appendix A.3.

4 Experiments

We evaluate current language models across the five primary quantum code-generation suites described in Section 3, and separately analyze QuantumBenchEval (QBE) as an evaluator-focused benchmark. Unless otherwise stated, results are reported as pass@1 with pointwise 95% task-bootstrap confidence intervals from 10,000 replicates. We emphasize task-level uncertainty and paired comparisons rather than interpreting raw leaderboard ordering as evidence of model separation. QuanBench results are retained as recorded evaluator outcomes but require replay following the evaluator audit described in Section 4.5.

4.1 Overall Performance

Table 2 summarizes pass@1 across the five primary suites. Performance varies substantially across benchmarks, and no single ordering describes all five evaluations. QiskitHumanEval Standard is particularly concentrated at the top: GPT-6 Astra reaches 98.7% pass@1 [96.7–100.0], Gemini 3.1 Pro Preview reaches 97.6% [95.1–99.6], and Muse-Glimmer-30B reaches 95.1% [92.5–97.5]. The corresponding Hard split remains considerably less saturated, with the highest recorded pass@1 at 63.8% [56.4–71.0].

QuanBench exhibits a different ordering. Claude Fable 5.1 records the highest pass@1 on QuanBench-44 at 60.0% [45.9–73.6], while several systems cluster near 84–85% on QuanBench-117. These QuanBench values should not be interpreted as validated capability estimates, however, because our evaluator audit identifies defects affecting both apparent passes and apparent failures (Section 4.5).

QCoder produces another distinct performance profile. Among complete runs, Claude Fable 5.1 records 43.0% pass@1 [33.7–52.2], followed by GPT-6 Astra at 32.8% [25.4–40.6]. Because QCoder conditions later generations on prior attempts, these pass@k values describe the recorded sequential protocol rather than an unbiased independent-sampling estimator.

4.2 Benchmark Saturation and Statistical Separability

The Standard and Hard QiskitHumanEval suites provide a useful contrast in remaining benchmark headroom. Across five attempts per model, only one of the 151 Standard tasks is unsolved by every evaluated model, compared with 31 of 151 Hard tasks. The highest observed pass@1 correspondingly falls from 98.7% on Standard to 63.8% on Hard. All 15 models record lower pass@1 on the Hard split, although we treat this as a benchmark-level contrast rather than a causal estimate of difficulty because task-ID correspondence alone does not establish a controlled paired intervention.

Raw leaderboard ordering also overstates how strongly the evaluated suites separate their highest-scoring systems. Table 3 reports the two highest observed pass@1 values on each suite together with paired task-level differences. After Holm correction within each suite, none of the five top-two gaps is significant. On Hard and QCoder the unadjusted evidence is suggestive (p=0.016 and 0.003; the paired intervals exclude zero), but it does not survive correction for the full set of within-suite comparisons. An observed first place therefore should not be read as evidence that the leading model is reliably better than the runner-up.

Model	Standard	Hard	QB-44‡	QB-117‡	QCoder†
Quantum-specialized models					
Granite-Qiskit	20.5 [15.9–25.6]	11.4 [7.7–15.5]	10.5 [5.9–15.9]	31.3 [26.3–36.2]	7.5 [1.5–14.9]
Qwen-Qiskit	53.8 [47.5–59.9]	22.3 [17.0–27.9]	27.7 [17.3–39.5]	74.5 [67.5–81.0]	4.8 [1.2–9.3]
Mistral-Qiskit	67.2 [60.1–74.2]	23.4 [17.7–29.5]	25.9 [14.5–37.7]	79.8 [73.2–86.3]	4.5 [1.2–9.0]
Retrieval-augmented system					
Quantum RAG	41.3 [35.6–47.2]	8.5 [5.3–11.9]	15.9 [7.7–25.0]	42.1 [35.2–49.1]	9.3 [3.9–15.5]
General-purpose open-weight models					
LLaMA-3.1-8B	5.7 [3.7–7.8]	0.1 [0.0–0.4]	31.4 [22.7–40.0]	28.0 [23.2–32.8]	4.2 [1.2–8.1]
Mistral-7B	9.4 [6.1–13.1]	0.4 [0.0–0.9]	5.0 [0.5–10.9]	11.6 [6.7–16.9]	2.4 [0.3–5.4]
Gemma-3-4B	65.2 [57.6–72.6]	0.1 [0.0–0.4]	11.8 [3.6–21.4]	77.6 [70.8–84.1]	3.9 [0.0–8.7]
Muse-Glimmer-30B	95.1 [92.5–97.5]	28.3 [22.1–34.7]	45.9 [33.2–58.6]	81.9 [75.0–88.0]	–*
Hosted models					
GPT-5	46.5 [39.2–53.9]	24.1 [18.0–30.5]	58.6 [45.5–71.4]	81.0 [74.4–87.2]	7.8 [2.1–14.6]
Gemini 3 Flash	62.8 [55.6–69.8]	32.6 [26.0–39.5]	25.5 [15.9–36.4]	44.4 [37.4–51.8]	2.4 [0.9–4.2]
Gemini 3.6 Flash	64.0 [56.6–71.0]	40.8 [33.5–48.2]	31.8 [20.0–44.5]	50.6 [43.1–58.0]	23.3 [16.4–30.7]
Claude Opus 4.6	71.7 [64.4–78.5]	44.1 [36.6–51.9]	52.3 [38.2–66.4]	84.4 [77.9–90.6]	–
Claude Fable 5.1	70.7 [63.4–78.0]	56.0 [48.6–63.4]	60.0 [45.9–73.6]	84.6 [77.8–90.6]	43.0 [33.7–52.2]
Gemini 3.1 Pro Preview	97.6 [95.1–99.6]	58.1 [51.1–65.3]	57.3 [42.3–70.9]	83.1 [76.4–89.6]	22.1 [14.6–30.1]
GPT-6 Astra	98.7 [96.7–100.0]	63.8 [56.4–71.0]	54.5 [38.6–68.2]	84.6 [77.8–90.6]	32.8 [25.4–40.6]

Table 2: Pass@1 (%) with pointwise 95% task-bootstrap confidence intervals across the five primary suites. †QCoder is judge-scored and uses a sequential, non-i.i.d. generation protocol. ‡QuanBench values are recorded evaluator outcomes requiring revalidation under the corrected isolated evaluator. *Muse QCoder is incomplete (45/67 tasks) and is therefore excluded from full-dataset comparison. Missing results are shown as "–" rather than imputed as zero.

Suite	Highest observed	P@1 [95% CI]	Runner-up	Paired gap, pp [95% CI]	p	Holm p
Standard	GPT-6 Astra	98.7 [96.7–100.0]	Gemini 3.1 Pro	1.1 [0.1–2.6]	0.128	0.90
Hard	GPT-6 Astra	63.8 [56.4–71.0]	Gemini 3.1 Pro	5.7 [1.3–10.3]	0.016	0.23
QuanBench-44‡	Claude Fable 5.1	60.0 [45.9–73.6]	GPT-5◇	1.4 [−8.6–10.9]	0.860	1.00
QuanBench-117‡	Claude Fable 5.1	84.6 [77.8–90.6]	GPT-6 Astra (tie)	0.0 [0.0–0.0]	1.000	1.00
QCoder†	Claude Fable 5.1§	43.0 [33.7–52.2]	GPT-6 Astra	10.1 [3.9–16.7]	0.003	0.13

Table 3: Top-two pass@1 comparison per suite. The paired gap is the mean per-task difference between the two models, with a pointwise task-bootstrap interval; p is a two-sided paired permutation test (10,000 sign flips). Holm adjustment is applied within each suite over all model pairs (105; 78 for QuanBench-44 and QCoder). Neither the intervals nor the unadjusted p values are multiplicity-adjusted. †Judge-scored, sequential protocol. ‡Recorded evaluator outcomes pending replay. §Judge shares a provider with the evaluated model. ◇High empty-output rate (Appendix D.4).

Suite choice also changes the ordering itself. Pass@1 rank agreement is high among most execution-scored suites (Spearman ρ 0.78–0.89) but only ρ=0.54 between QiskitHumanEval-Standard and QCoder (Appendix A.4), so no single suite recovers the cross-suite ranking. Additional samples do not separate the leaders either: the strongest models gain at most about 6 pp from pass@1 to pass@5 on the execution-scored suites (zero for GPT-6 Astra on three of four), whereas quantum-specialized models gain up to 38 pp and reorder the lower half of the leaderboard (Appendix D.2).

Observed coverage reinforces the difference between the suites without implying that any task is intrinsically unsolvable. Across the available roster, the number of tasks unsolved in all five recorded attempts is 1/151 for Standard, 31/151 for Hard, 9/44 for QuanBench-44, 17/117 for QuanBench-117, and 12/67 for QCoder. These are empirical coverage statistics, equivalent to observed pass@5 coverage across the roster, rather than estimates of a fundamental capability ceiling.

4.3 Category-Level Performance

Figure 1 reports pass@1 across the ten shared QiskitHumanEval task categories for Standard and Hard. Standard is broadly saturated among the strongest models: GPT-6 Astra reaches 100% pass@1 in eight of ten categories, with the remaining two at 96.0% and 95.2%.

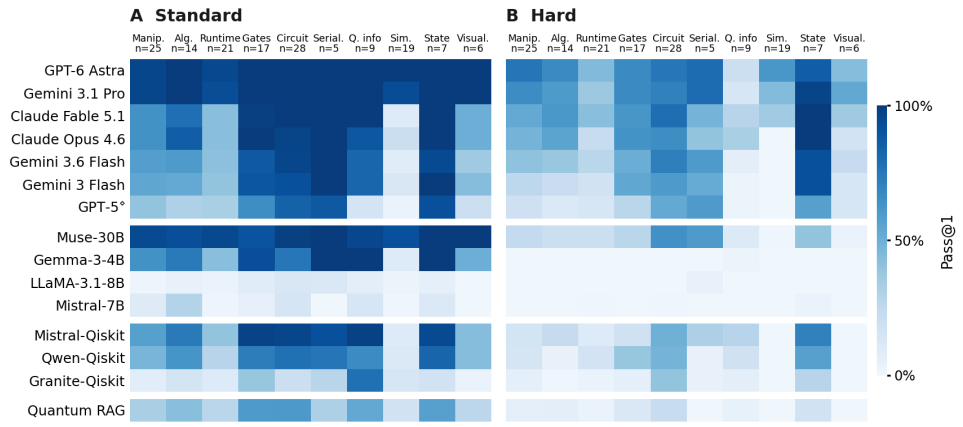

Figure 1: **Category-level pass@1 on QiskitHumanEval Standard and Hard.** Rows are models, grouped by family; columns are the ten task categories, with per-category task counts (n) in the headers. Standard is near saturation for the strongest models across most categories, while Hard shows larger differences across task types. Categories with fewer than ten tasks (e.g., Serialization, n=5) have wide intervals, and a single-category difference should not be read as a model ranking. Per-category 95% task-bootstrap intervals for every model are given in Appendix B.2.

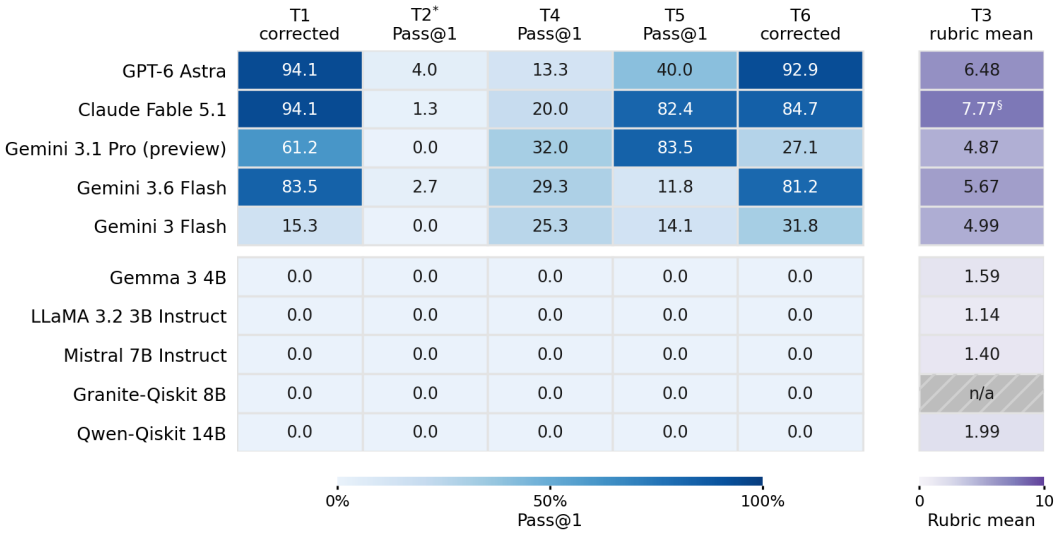

Figure 2: **QuantumBenchEval results by topic.** Pass@1 (%) for T1, T2, T4, T5 and T6 (blue, shared scale). T3 (purple, separate scale) is an LLM-judged rubric mean over judged samples only; it is not a pass rate and should not be compared with the other columns. n/a: no judged samples. T1 and T6 use corrected scoring that removes checks on quantities the prompts did not specify, applied to the same saved outputs. This raises GPT-6 Astra from 0% to 94.1% on T1 and from 1.2% to 92.9% on T6. *T2 is reported as recorded (Appendix C.4). §The judge shares a provider with the evaluated model. The horizontal gap separates hosted frontier models from small open-weight and quantum-specialized models.

Hard reveals substantially greater category-level variation. For Astra, pass@1 ranges from 20.0% on QUANTUM INFORMATION and 44.8% on BACKEND AND RUNTIME to 75.0% on QUANTUM CIRCUIT GENERATION and 85.7% on STATE PREPARATION AND ANALYSIS; similar heterogeneity appears across other strong models. Because category sizes range from 5 to 28 tasks, these results are interpreted descriptively rather than as a secondary model ranking.

4.4 QUANTUMBENCHEVAL: CHALLENGING TASKS AND EVALUATION SENSITIVITY

QuantumBenchEval (QBE) complements the established suites with 96 new tasks spanning six quantum-computing topics. Instead of short API-completion problems, its tasks are multi-step scientific workflows that require reasoning about algorithm design, resource use, optimization, simulation, and scientific outputs. Each task is scored by its own evaluator, and because the tasks are newly constructed, they are less likely than long-standing public suites to appear in model training data.

Figure 2 shows substantial variation across both models and topics. The hosted frontier models score above zero on several topics, while the smaller open-weight and quantum-specialized systems score zero pass@1 on every assertion-scored topic. We do not read these zeros as an absence of quantum knowledge. The same models receive nonzero T3 rubric scores (1.1–2.0 of 10), so the zeros more plausibly reflect failure to produce complete outputs in the required structure than failure to engage with the task. Among the stronger systems, performance is highly topic dependent. GPT-6 Astra and Claude Fable 5.1 both reach 94.1% pass@1 on corrected T1, yet score 40.0% [22.4–58.8] and 82.4% [69.4–94.1] on T5. Gemini 3.1 Pro Preview reaches 83.5% [76.5–90.6] on T5, while no model exceeds 32% on T4. The six topics therefore probe different capabilities, and no single model ordering holds across them.

We applied the same audit to our own evaluators, checking whether each scoring rule matches the information given to the model. It flagged three topics. T1's evaluator checked exact resource quantities the prompts never specified, and T6's required a scalar `reproducibility` value whose form the prompts never stated. Removing only these checks, while keeping every correctness check and rescoring the saved outputs, raises GPT-6 Astra from 0% to 94.1% on T1 and from 1.2% to 92.9% on T6; on T6 this moves it from last to first among the frontier models. All 32 reference solutions pass under both evaluators, and no replayed sample changes from pass to fail (Appendix C.4). T2 also checks undisclosed evaluation counts, but removing them changes little (at most 13.3% pass@1 on the 12 internally consistent tasks), and three T2 tasks contain data that contradict their required answers, so we report T2 as recorded. T3 is scored differently again, as a rubric mean over judged samples rather than pass@1.

4.5 QUANBENCH EVALUATOR AUDIT

The 17 QuanBench-117 tasks unsolved across the observed roster initially appear to form a shared performance ceiling. Inspection instead identifies a common benchmark-side obstruction: for each of the 17 tasks, the dominant recorded failure corresponds to an import in benchmark-owned reference or test code, including dependencies from `qiskit_algorithms`, `qiskit_optimization`, `qiskit.primitives`, and `qiskit_machine_learning`.This does not imply that every candidate on these tasks is correct (some outputs fail before reaching the shared obstruction), but it prevents some outputs fail before reaching the shared obstruction, but it prevents the observed ceiling from being interpreted as a pure model-capability limit.

More importantly, source inspection and controlled synthetic probes identify three defects in the current QuanBench evaluator. First, the evaluator executes test scripts without explicitly loading and running their `unittest` test cases; all 117 current QuanBench-117 scripts contain test classes and no top-level expression calls. A deliberately incorrect candidate therefore passes when the failing assertion is confined to such a test class. Second, canonical code is executed in the candidate namespace before generated code, allowing an empty candidate to satisfy an explicit assertion by inheriting the canonical function. Third, a valid synthetic candidate can fail before execution when the canonical solution imports a missing dependency. The false-pass mechanism is not only hypothetical: a recorded QuanBench-117 pass (Task 110, Claude Opus 4.6) returns a circuit that is not equivalent to CNOT up to global phase (Appendix E).

These controlled probes demonstrate mechanisms for both false passes and false failures. Historical evaluator source was not immutably captured, so they do not prove that every saved QuanBench outcome was produced by the exact inspected revision. We therefore retain the historical scores as *recorded evaluator outcomes* and require replay of unchanged candidates under a versioned, isolated evaluator before interpreting either successes or failures as validated capability estimates. QuanBench-44 has a different test-script structure, so the class-only finding for QuanBench-117 is not automatically extended to every QuanBench-44 task.

4.6 JUDGE-SCORED QCODER

QCoder evaluates open-ended programs with a model-based judge rather than executable unit tests. Among complete evaluations, Claude Fable 5.1[§] records 43.0% pass@1 [33.7–52.2] and 67.2% observed pass@5 [55.2–77.6], while GPT-6 Astra records 32.8% pass@1 [25.4–40.6] and the same 67.2% observed pass@5. Equal coverage does not mean identical behavior: each model solves 45/67 tasks within five attempts, with 40 shared successes and five unique successes each, so their oracle union covers 50/67 tasks (74.6%). This shows complementary observed coverage, not a demonstrated ensemble, since combining the systems doubles the generation budget and relies on oracle knowledge of which candidate succeeds.

These results are also conditioned on the generation configuration. GPT-6 Astra, Claude Fable 5.1, and Muse-Glimmer-30B produced no usable code under a 2,048-token output limit, so their runs used 32,768 tokens. Earlier runs, including GPT-5, used 2,048, and GPT-5's 63% empty-output rate on QCoder shows that the cap was binding for it as well (Appendix D.4). The judge also changed between the two groups (`claude-sonnet-4` before April 2026, `claude-sonnet-4-6` after), and neither group was re-judged with the other judge. Gaps between the earlier and later runs therefore reflect model capability together with output budget and judge version, and the within-group comparison of Fable and Astra is the most controlled one QCoder provides.

Muse-Glimmer-30B is provisional: it has completed 45/67 tasks (completed-subset pass@1 53.3%; full-dataset bounds 35.8–68.7%, obtained by assigning every unfinished task failure or success) and is excluded from QCoder rankings.

5 DISCUSSION AND CONCLUSION

Our results show that measured quantum code-generation performance depends as much on how correctness is operationalized as on the model evaluated. Correcting one QBE evaluator changes two models' pass@1 by 94.1 pp on unchanged outputs; the current QuanBench evaluator admits both false passes and false failures; and no primary suite separates its top two models after multiplicity correction. Meanwhile QiskitHumanEval-Standard is near saturation, so headroom now lies in harder or less-exposed tasks such as the Hard split and QBE.

Implications for model comparison. These effects compound. A leaderboard entry for QuanBench-117 reflects the model, its output budget, whether the evaluator runs its tests, and whether reference code leaks into the candidate namespace; a QCoder entry additionally reflects the judge version. Comparisons are most trustworthy within a single configuration and evaluator revision, as with Claude Fable 5.1 and GPT-6 Astra on QCoder. Across configurations, differences smaller than the evaluator effects documented here should not be interpreted as capability differences.

Recommendations. Quantum code benchmarks should (i) validate reference solutions and confirm that tests execute under the released harness; (ii) isolate candidate code from reference code; (iii) disclose every quantity the scorer checks; (iv) report valid-output rates alongside pass@k; (v) report uncertainty and corrected paired comparisons rather than raw ordering; and (vi) use independent sampling whenever pass@k is estimated.

Limitations. QuanBench outcomes await replay under an isolated evaluator. QBE T2 contracts are corrected only in the release, and T1/T6 rescoring used replay on a different platform (5 of 1,600 samples differ). QCoder uses a non-i.i.d. protocol, its judge changed between runs, and all judges share a provider with one evaluated model. Decoding settings and output budgets differ across models, and our roster reflects one point in time.

AI USE STATEMENT

In this work, we used generative AI tools to assist with implementing portions of the initial analysis and evaluation code and with developing and refining the statistical analysis framework, including bootstrap confidence intervals, paired comparisons, and multiple-comparison correction. We did not use generative AI tools to generate synthetic datasets, formulate or prove mathematical claims, clean

or reformat datasets, perform qualitative or thematic data analysis, or translate research materials; the remaining required-disclosure tasks were not applicable to this work.

Additionally, we used generative AI tools to draft and edit portions of the manuscript for clarity, organization, and readability, and to assist with creating and editing research software. All AI-assisted code and statistical analyses were reviewed and tested by the authors, and reported results were verified against the underlying experimental outputs and evaluation artifacts. All AI-assisted manuscript text was reviewed and revised by the authors. We take responsibility for the final content of this work, including text, claims, code, and artifacts produced with the aid of generative AI.

Ethics statement

This work evaluates language models on quantum code-generation benchmarks and does not involve human subjects, personally identifiable information, or sensitive personal data. We use benchmark data and model outputs solely for research evaluation and report known limitations, missing results, and evaluator defects rather than presenting provisional measurements as validated capability estimates. Released benchmark materials, model outputs, evaluation artifacts, and code will be distributed in accordance with applicable licenses and terms of use. We are not aware of additional ethical concerns arising from the experiments presented in this work.

Reproducibility statement

All code, evaluation scripts, model configurations, prompts, and experimental results will be made publicly available upon acceptance. The datasets and models evaluated in this work are publicly available, and the paper and supplementary materials describe the benchmark construction, evaluation protocol, and experimental settings required to reproduce our results. We will additionally maintain a public version of the Quantum Arena containing links to the datasets, models, benchmark tasks, and detailed evaluation results. The arena is intended to support continued evaluation beyond this work, and we welcome additional model submissions to maintain the benchmark as a long-term, continuously updated quantum-computing leaderboard.

Author Contributions

Contributions have been withheld for double blind anonymity.

Acknowledgments

All acknowledgments have been withheld for double blind anonymity.

References

Anthropic. Claude Fable 5.1. Claude Platform Documentation, 2026a. Released September 1, 2026.

Anthropic. Claude Opus 4.6. Claude Platform Documentation, 2026b. Released February 5, 2026.

Boran Apak, Medina Bandic, Aritra Sarkar, and Sebastian Feld. Ketgpt–dataset augmentation of quantum circuits using transformers, 2024.

Yaiza Aragonés-Soria and Manuel Oriol. C4q: A chatbot for quantum, 2024.

Mislav Balunovic, Jasper Dekoninck, Ivo Petrov, Nikola Jovanović, and Martin Vechev. Matharena: Evaluating llms on uncontaminated math competitions. *Advances in Neural Information Processing Systems*, 38, 2026.

Abdul Basit, Nouhaila Innan, Muhammad Haider Asif, Minghao Shao, Muhammad Kashif, Alberto Marchisio, and Muhammad Shafique. Pennylang: Pioneering llm-based quantum code generation with a novel pennylane-centric dataset, 2025a.

Abdul Basit, Minghao Shao, M. Asif, Nouhaila Innan, Muhammad Kashif, Alberto Marchisio, and Muhammad Shafique. Pennycoder: Efficient domain-specific llms for pennylane-based quantum code generation, 2025b.

Abdul Basit, Minghao Shao, Muhammad Haider Asif, Nouhaila Innan, Muhammad Kashif, Alberto Marchisio, and Muhammad Shafique. Qhackbench: Benchmarking large language models for quantum code generation using pennylane hackathon challenges, 2025c.

Mark Chen, Jerry Tworek, Heewoo Jun, Qiming Yuan, Henrique Ponde de Oliveira Pinto, Jared Kaplan, Harri Edwards, Yuri Burda, Nicholas Joseph, Greg Brockman, Alex Ray, Raul Puri, Gretchen Krueger, Michael Petrov, Heidy Khlaaf, Girish Sastry, Pamela Mishkin, Brooke Chan, Scott Gray, Nick Ryder, Mikhail Pavlov, Alethea Power, Lukasz Kaiser, Mohammad Bavarian, Clemens Winter, Philippe Tillet, Felipe Petroski Such, Dave Cummings, Matthias Plappert, Fotios Chantzis, Elizabeth Barnes, Ariel Herbert-Voss, William Hebgen Guss, Alex Nichol, Alex Paino, Nikolas Tezak, Jie Tang, Igor Babuschkin, Suchir Balaji, Shantanu Jain, William Saunders, Christopher Hesse, Andrew N. Carr, Jan Leike, Josh Achiam, Vedant Misra, Evan Morikawa, Alec Radford, Matthew Knight, Miles Brundage, Mira Murati, Katie Mayer, Peter Welinder, Bob McGrew, Dario Amodei, Sam McCandlish, Ilya Sutskever, and Wojciech Zaremba. Evaluating large language models trained on code, 2021.

Nicolas Dupuis, Luca Buratti, Sanjay Vishwakarma, Aitana Viudes Forrat, David Kremer, Ismael Faro, Ruchir Puri, and Juan Cruz-Benito. Qiskit code assistant: Training llms for generating quantum computing code, 2024.

Nicolas Dupuis, Adarsh Tiwari, Youssef Mroueh, David Kremer, Ismael Faro, and Juan Cruz-Benito. Quantum verifiable rewards for post-training qiskit code assistant, 2025.

Google. Gemini 3.6 Flash. Google AI, 2026. Released July 21, 2026.

Google DeepMind. Gemini 3 Flash. Gemini Model Documentation, 2025.

Google DeepMind. Gemini 3.1 Pro. Gemini Model Card, 2026. Published February 19, 2026.

Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Alex Vaughan, et al. The llama 3 herd of models. *arXiv preprint arXiv:2407.21783*, 2024.

Xiaoyu Guo, Minggu Wang, and Jianjun Zhao. Quanbench: Benchmarking quantum code generation with large language models. In *2025 40th IEEE/ACM International Conference on Automated Software Engineering (ASE)*, pp. 2657–2669. IEEE, 2025.

IBM Quantum. Granite-3.2-8B-Qiskit. Hugging Face Model Card, 2025a. Released June 3, 2025.

IBM Quantum. Mistral-Small-3.2-24B-Qiskit. Hugging Face Model Card, 2025b.

IBM Quantum. Qwen2.5-Coder-14B-Qiskit. Hugging Face Model Card, 2025c.

Naman Jain, Alex Gu, Wen-Ding Li, Fanjia Yan, Tianjun Zhang, Sida Wang, Armando Solar-Lezama, Koushik Sen, and Ion Stoica. Livecodebench: Holistic and contamination free evaluation of large language models for code. In *International Conference on Learning Representations*, volume 2025, pp. 58791–58831, 2025.

Linus Jern, Valter Uotila, Cong Yu, and Bo Zhao. Agent-q: Fine-tuning large language models for quantum circuit generation and optimization, 2025.

Albert Q. Jiang, Alexandre Sablayrolles, Arthur Mensch, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Florian Bressand, Gianna Lengyel, Guillaume Lample, Lucile Saulnier, Lélio Renard Lavaud, Marie-Anne Lachaux, Pierre Stock, Teven Le Scao, Thibaut Lavril, Thomas Wang, Timothée Lacroix, and William El Sayed. Mistral 7b, 2023. URL `https://arxiv.org/abs/2310.06825`.

Carlos E Jimenez, John Yang, Alexander Wettig, Shunyu Yao, Kexin Pei, Ofir Press, and Karthik Narasimhan. Swe-bench: Can language models resolve real-world github issues?, 2024. *URL https://arxiv. org/abs/2310.06770*, 7, 2023.

Kiana Kheiri, Aamna Aamir, Andriy Miranskyy, and Chen Ding. Qspark: Towards reliable qiskit code generation, 2025.

Jiawei Liu, Chunqiu Steven Xia, Yuyao Wang, and Lingming Zhang. Is your code generated by chatgpt really correct? rigorous evaluation of large language models for code generation. *Advances in neural information processing systems*, 36:21558–21572, 2023.

Jianglin Lu, Hailing Wang, Xu Ma, Qihua Dong, Mingyuan Zhang, Yizhou Wang, and Yun Fu. Muse: A unified agentic harness for mllms, 2026. URL https://arxiv.org/abs/2606.03005.

Taku Mikuriya, Tatsuya Ishigaki, Masayuki Kawarada, Shunya Minami, Tadashi Kadowaki, Yohichi Suzuki, Soshun Naito, Shunya Takada, Takumi Kato, Tamotsu Basseda, et al. Qcoder benchmark: Bridging language generation and quantum hardware through simulator-based feedback, 2025.

Evan Miller. Adding error bars to evals: A statistical approach to language model evaluations. *arXiv preprint arXiv:2411.00640*, 2024.

OpenAI. GPT-6 Astra. OpenAI API Documentation, 2026. Accessed 2026.

Shuo Ren, Daya Guo, Shuai Lu, Long Zhou, Shujie Liu, Duyu Tang, Neel Sundaresan, Ming Zhou, Ambrosio Blanco, and Shuai Ma. Codebleu: a method for automatic evaluation of code synthesis, 2020. URL https://arxiv.org/abs/2009.10297.

Carlos A. Riofrío, Johannes Klepsch, Jernej Rudi Finžgar, Florian Kiwit, Leonhard Hölscher, Marvin Erdmann, Lukas Müller, Chandan Kumar, Youssef Achari Berrada, and Andre Luckow. Quantum computing for automotive applications, 2025. URL https://arxiv.org/abs/2409.14183.

Aaditya Singh, Adam Fry, Adam Perelman, Adam Tart, Adi Ganesh, Ahmed El-Kishky, Aidan McLaughlin, Aiden Low, AJ Ostrow, Akhila Ananthram, et al. Openai gpt-5 system card. *arXiv preprint arXiv:2601.03267*, 2025.

Nicholas Stapleton, Sharmin Afrose, Ethan Seefried, Vicente Leyton Ortega, Travis S. Humble, and Tirthankar Ghosal. Quantum-rag: Retrieval-augmented generation framework for quantum circuit code generation. In *2026 IEEE 19th Dallas Circuits and Systems Conference (DCAS)*, pp. 1–5, 2026. doi: 10.1109/DCAS69364.2026.11544968.

Gemma Team, Aishwarya Kamath, Johan Ferret, Shreya Pathak, Nino Vieillard, Ramona Merhej, Sarah Perrin, Tatiana Matejovicova, Alexandre Ramé, Morgane Rivière, Louis Rouillard, Thomas Mesnard, Geoffrey Cideron, Jean bastien Grill, Sabela Ramos, Edouard Yvinec, Michelle Casbon, Etienne Pot, Ivo Penchev, Gaël Liu, Francesco Visin, Kathleen Kenealy, Lucas Beyer, Xiaohai Zhai, Anton Tsitsulin, Robert Busa-Fekete, Alex Feng, Noveen Sachdeva, Benjamin Coleman, Yi Gao, Basil Mustafa, Iain Barr, Emilio Parisotto, David Tian, Matan Eyal, Colin Cherry, Jan-Thorsten Peter, Danila Sinopalnikov, Surya Bhupatiraju, Rishabh Agarwal, Mehran Kazemi, Dan Malkin, Ravin Kumar, David Vilar, Idan Brusilovsky, Jiaming Luo, Andreas Steiner, Abe Friesen, Abhanshu Sharma, Abheesht Sharma, Adi Mayrav Gilady, Adrian Goedeckemeyer, Alaa Saade, Alex Feng, Alexander Kolesnikov, Alexei Bendebury, Alvin Abdagic, Amit Vadi, András György, André Susano Pinto, Anil Das, Ankur Bapna, Antoine Miech, Antoine Yang, Antonia Paterson, Ashish Shenoy, Ayan Chakrabarti, Bilal Piot, Bo Wu, Bobak Shahriari, Bryce Petrini, Charlie Chen, Charline Le Lan, Christopher A. Choquette-Choo, CJ Carey, Cormac Brick, Daniel Deutsch, Danielle Eisenbud, Dee Cattle, Derek Cheng, Dimitris Paparas, Divyashree Shivakumar Sreepathihalli, Doug Reid, Dustin Tran, Dustin Zelle, Eric Noland, Erwin Huizenga, Eugene Kharitonov, Frederick Liu, Gagik Amirkhanyan, Glenn Cameron, Hadi Hashemi, Hanna Klimczak-Plucińska, Harman Singh, Harsh Mehta, Harshal Tushar Lehri, Hussein Hazimeh, Ian Ballantyne, Idan Szpektor, Ivan Nardini, Jean Pouget-Abadie, Jetha Chan, Joe Stanton, John Wieting, Jonathan Lai, Jordi Orbay, Joseph Fernandez, Josh Newlan, Ju yeong Ji, Jyotinder Singh, Kat Black, Kathy Yu, Kevin Hui, Kiran Vodrahalli, Klaus Greff, Linhai Qiu, Marcella Valentine, Marina Coelho, Marvin Ritter, Matt Hoffman, Matthew Watson, Mayank Chaturvedi, Michael Moynihan, Min Ma, Nabila Babar, Natasha Noy, Nathan Byrd, Nick Roy, Nikola Momchev, Nilay Chauhan, Noveen Sachdeva, Oskar Bunyan, Pankil Botarda, Paul Caron, Paul Kishan Rubenstein, Phil Culliton, Philipp Schmid, Pier Giuseppe Sessa, Pingmei Xu, Piotr Stanczyk, Pouya Tafti, Rakesh Shivanna, Renjie Wu, Renke Pan, Reza Rokni, Rob Willoughby, Rohith Vallu, Ryan Mullins, Sammy Jerome, Sara Smoot, Sertan Girgin, Shariq Iqbal, Shashir Reddy, Shruti Sheth, Siim Põder, Sijal Bhatnagar, Sindhu Raghuram Panyam, Sivan Eiger, Susan Zhang, Tianqi Liu, Trevor Yacovone, Tyler Liechty, Uday Kalra, Utku Evci, Vedant Misra, Vincent Roseberry, Vlad Feinberg, Vlad Kolesnikov,

Woohyun Han, Woosuk Kwon, Xi Chen, Yinlam Chow, Yuvein Zhu, Zichuan Wei, Zoltan Egyed, Victor Cotruta, Minh Giang, Phoebe Kirk, Anand Rao, Kat Black, Nabila Babar, Jessica Lo, Erica Moreira, Luiz Gustavo Martins, Omar Sanseviero, Lucas Gonzalez, Zach Gleicher, Tris Warkentin, Vahab Mirrokni, Evan Senter, Eli Collins, Joelle Barral, Zoubin Ghahramani, Raia Hadsell, Yossi Matias, D. Sculley, Slav Petrov, Noah Fiedel, Noam Shazeer, Oriol Vinyals, Jeff Dean, Demis Hassabis, Koray Kavukcuoglu, Clement Farabet, Elena Buchatskaya, Jean-Baptiste Alayrac, Rohan Anil, Dmitry, Lepikhin, Sebastian Borgeaud, Olivier Bachem, Armand Joulin, Alek Andreev, Cassidy Hardin, Robert Dadashi, and Léonard Hussenot. Gemma 3 technical report, 2025. URL https://arxiv.org/abs/2503.19786.

Sanjay Vishwakarma, Francis Harkins, Siddharth Golecha, Vishal Sharathchandra Bajpe, Nicolas Dupuis, Luca Buratti, David Kremer, Ismael Faro, Ruchir Puri, and Juan Cruz-Benito. Qiskit humaneval: An evaluation benchmark for quantum code generative models, 2024.

Rui Yang, Yuntian Gu, Ziruo Wang, Yitao Liang, and Tongyang Li. Qcircuitnet: A large-scale hierarchical dataset for quantum algorithm design, 2024.

Lianmin Zheng, Wei-Lin Chiang, Ying Sheng, Siyuan Zhuang, Zhanghao Wu, Yonghao Zhuang, Zi Lin, Zhuohan Li, Dacheng Li, Eric Xing, et al. Judging llm-as-a-judge with mt-bench and chatbot arena. *Advances in neural information processing systems*, 36:46595–46623, 2023.