Beyond Pass@1: K-Sample Behavioral Equivalence for Code-Agent Evaluation

Ahmad A. Rushdi
rushdi@stanford.edu
Stanford University
Stanford, CA, USA

Abstract

Most code-agent benchmarks are *observational*: each problem is attempted once, scored against a gold test suite, and aggregated into a single pass-rate. But two agents with the same pass-rate can redeploy very differently — one concentrated on correct programs, another spread across wrong-but-plausible variants — and an observational metric cannot tell them apart. We propose an *interventional* evaluation: resample the agent's decoding several times per problem, execute each candidate in a sandbox, and cluster the outcomes by per-test pass/fail signature. The maximum-cluster frequency captures behavioural concentration; Conformal Risk Control on held-out correctness labels picks an acceptance threshold that bounds expected silent-failure risk under exchangeability of calibration and test items. We split the test bank into a probe half (used to compute the behavioural signature) and a disjoint gold half (used to score correctness), so "silent failure" means accepted on the probe half but wrong on the gold half. On HumanEval+ with Qwen2.5-Coder-1.5B-Instruct at eight samples per problem and a 10% silent-failure budget, our wrapper attains 1.65 and 1.94 times the effective pass-rate of single-sample logprob conformal at matched silent failure for moderate temperatures (0.4 and 0.8); at the highest temperature tested (1.2) the advantage reverses, an honest applicability boundary traced to low dominant-cluster correctness. The wrapper also surfaces a *stable-wrong* regime — up to 12% of problems where all sampled solutions agree on a wrong answer, invisible to observational and logprob baselines — and motivates sample-collapse as the natural reward-hacking failure mode.

Keywords

agent evaluation, conformal prediction, abstention, equivalence classes, code generation, interventional evaluation, LLM verifiers

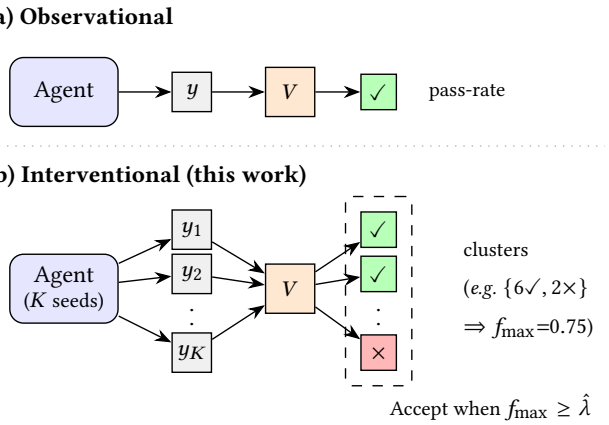


Figure 1: Observational vs. interventional evaluation. (a) One trajectory y, scored by a verifier V, yields a single pass/fail label. (b) Our protocol resamples K trajectories and clusters them by behaviour signature; the example shows a 6:2 split, giving a maximum-cluster frequency $f_{\max}(x)$=0.75. We accept the dominant class when $f_{\max}(x)$ clears an acceptance threshold λ; the smallest λ satisfying the finite-sample silent-failure bound on a held-out split is $\hat{\lambda}$, calibrated by CRC (§2). Both panels show a single verifier V; the method uses a *probe* verifier V_{probe} on one half of the test bank (for clustering) and a disjoint *gold* verifier V_{gold} on the other (for correctness).

1 Introduction

Problem. Most code-agent benchmarks evaluate *observationally*: the agent emits a single trajectory per problem, a verifier scores it, and the results are averaged into pass-rate or pass@k [5, 12]. This answers *"what happened on this run?"* but says nothing about *"what would happen on the next?"*. For deployment, where the same agent is invoked repeatedly under nominally identical conditions, the second question is the one that matters. Two agents with the same pass-rate can fail very differently when redeployed: one concentrates its probability mass on a narrow basin of correct programs, the other spreads it across many incorrect-but-plausible variants — and pass-rate cannot distinguish them. Figure 1 illustrates that gap.

Our approach. We propose an *interventional* evaluation of code agents: at each problem we draw K stochastic perturbations of the agent's decoding process (varying random seed, with fixed temperature), execute each in a sandbox, and read the equivalence-class structure of the resulting outcomes. The intervention is on the agent's sampling distribution rather than on the environment or the prompt, so it is a stochastic counterfactual rather than a strict do-calculus intervention [14]. The reliability signal is the *shape* of the K-sample distribution: if all K candidates execute to the same observable behavior on a held-out test bank, the agent is concentrated on this problem; if they split across multiple behaviors, the agent is uncertain. We summarize concentration with the max cluster frequency $f_{\max} \in [1/K, 1]$ and feed it to a Conformal Risk Control calibrator [3], which converts $f_{\max}$ into an abstention rule that targets a finite-sample bound on *expected* silent-failure risk under exchangeability of calibration and test items.

Scope: a general framing, instantiated on code. The wrapper is agent-agnostic. Treat the agent as a stochastic policy that, given a problem x, emits *trajectories* — sequences of action choices and intermediate observations terminating in an outcome that a verifier

can score. In a multi-step RL setting, a trajectory is a full rollout: action choices, tool calls, and intermediate states under a fixed environment seed. Code generation is the simplest single-step instance of the same object: a trajectory collapses to a sampled decoding sequence whose terminal artifact is a *candidate program* y, with the verifier-observable outcome being the per-test pass/fail signature on running y. The primitive of the wrapper — resample K trajectories under fixed conditions, cluster by any verifier-observable behaviour (final reward, test signature, tool-call pattern), feed the maximum-cluster frequency to CRC — is the same in both regimes. The empirical results in this paper instantiate the single-step code-agent case (HumanEval+ with Qwen2.5-Coder); the abstention guarantee and the stable-wrong diagnostic are written for the general trajectory-space view, and extending the experiments to multi-step RL agents on SWE-Bench [7]-style benchmarks is direct.

Related work. Several existing methods share primitives with our wrapper but stop short of calibrated abstention from K-sample equivalence-class structure on executable code. *AlphaCode* [11] clusters K candidate programs by behavior on auto-generated tests — the same equivalence primitive — and uses the clustering for *best-of-K selection*; no abstention rule, no finite-sample guarantee. *Self-consistency* [18] majority-votes K chain-of-thought samples on the final textual answer; no executable verifier and no calibration step. *Semantic uncertainty* [8] clusters K NLG samples by entailment for a scalar uncertainty estimate; the equivalence relation is a noisy LLM judgement and the output is not an abstention decision. *LLM-as-a-Verifier* [9] aggregates verifier log-probabilities across scoring granularity and repeated verification for trajectory *ranking*; the verifier-aggregation primitive is the same, the downstream task is different. *Conformal selective generation* [1, 13, 15] applies split-conformal to LM outputs using single-sample log-probability or factuality-based nonconformity scores; the conformal calibration is shared, but a single-sample score cannot see cross-sample disagreement. *Repeated-sampling analysis* [4] catalogues the inference-compute scaling regime our method sits in but does not propose a calibrated decision rule. Our wrapper occupies the intersection: equivalence-class clustering on executable code, repeated sampling, *and* finite-sample-bounded selective abstention.

Contributions. We formalize K-sample interventional evaluation via execution-equivalence-class variance and obtain a finite-sample bound on the expected silent-failure loss $L(\lambda; x) = \mathbb{1}\{c(x) \geq \lambda \wedge \text{accept-wrong}\}$ by feeding the concentration score $c(x) = f_{\max}(x)$ to a Conformal Risk Control calibrator [3]. We analyze the natural adversarial response — sample-collapse, where an agent emits the same program K times to inflate $f_{\max}$ — and propose three structural mitigations (diversity floor, seed perturbation by the grader, cluster-count penalty). We provide a paired experimental protocol that compares our metric against three baselines on identical K=8 samples per problem.

Comparison and metrics. We pair four methods on the same generated samples per HumanEval+ problem: *(i) observational pass@1* as the dominant baseline; *(ii) best-of-K*, the strongest no-abstention K-sample baseline, capturing the AlphaCode-style "some sample passes" framing [11]; *(iii) single-sample logprob conformal*, the closest conformal cousin [1, 15], which shares our calibration pipeline but uses average per-token log-probability as the score; and *(iv) ours*, $f_{\max}$ over K-sample equivalence classes. We omit self-consistency, semantic uncertainty, and LLM-as-a-Verifier as direct baselines: their primitives either degenerate to ours under an executable verifier (semantic uncertainty), require a textual answer that code does not have (self-consistency), or target ranking rather than abstention (LLM-as-a-Verifier). For each method we report five quantities: abstention rate, accuracy on answered items, effective pass-rate (fraction both answered and correct), empirical silent-failure rate at α=0.10 for the CRC methods, and median wall-time per problem. These together separate ability from calibration from cost.

Cost. The wrapper is not K-fold more expensive than single-sample evaluation. Generation scales sublinearly because the K candidates share prefill and are produced in a single batched decode; verifier cost scales linearly in K but each y_k is independent and parallelisable. On Qwen2.5-Coder-1.5B at K=8 over the full 164-problem HumanEval+ sweep at T=0.8 we measure a median wall-time of ∼76 s per problem (gen ∼25 s, exec ∼49 s on the rigorous plus-test bank), against an estimated ∼10 s for observational K=1 — roughly a 7× overhead, dominated by verifier execution. K controls the resolution of $f_{\max} \in \{1/K, \ldots, 1\}$: K=4 already detects a 3:1 split, K=8 doubles that resolution, and marginal value flattens above $K \approx 10$–20 [4]. The CRC risk tolerance α sets the abstention rate and thereby the rate at which flagged items are routed to human review; the wrapper is useful when silent-failure cost dominates review cost, and α exposes that tradeoff as a single tunable knob (Appendices D, E, F).

2 K-Sample Behavioral-Equivalence Wrapper

Setup: probe and gold verifiers. Let $\mathcal{A}$ be an agent that, given problem x, samples $\pi_{\mathcal{A}}(\cdot \mid x)$ over executable artifacts. To avoid using the same evidence for clustering and for correctness scoring, we split the held-out test bank $\mathcal{T}$ into disjoint subsets $\mathcal{T} = \mathcal{T}_{\text{probe}} \sqcup \mathcal{T}_{\text{gold}}$ and use two verifiers (same sandbox, different test indices). The *probe verifier* returns a per-test *behaviour signature* $\phi_{V_{\text{probe}}}(y) = (\mathbb{1}\{y \text{ passes } t\})_{t \in \mathcal{T}_{\text{probe}}}$ that drives clustering. The *gold verifier* V_{gold} defines correctness (y correct iff it passes all $\mathcal{T}_{\text{gold}}$ tests) and is used only for CRC calibration labels and held-out evaluation; the wrapper never sees V_{gold} at acceptance time. Concretely on HumanEval+, $\mathcal{T}_{\text{probe}}$ and $\mathcal{T}_{\text{gold}}$ are the first-half and second-half indices of each problem's plus test bank. Decoding temperature is denoted T in §3; $\mathcal{T}$ is the test bank, distinct. Two artifacts y_1, y_2 are *probe-equivalent* if $\phi_{V_{\text{probe}}}(y_1) = \phi_{V_{\text{probe}}}(y_2)$. We use this per-test bit vector as the primary signature; coarser alternatives (e.g. a (any-base-pass, any-plus-pass) pair) collapse too many distinct programs into the same bucket and are reported as ablation in Appendix C.

Concentration scores. Given K samples from $\pi_{\mathcal{A}}(\cdot \mid x)$, partition into equivalence classes $C_1, \ldots, C_m$ and report two summary quantities:

$$f_{\max}(x) = \max_j |C_j|/K \in [1/K, 1],$$
$$f_{\text{pass}}(x) = \tfrac{1}{K}\big|\{k : \phi_V(y_k) \text{ all-pass}\}\big|.$$

$f_{\max}$ measures *concentration* (cluster mass under resampling) and f_{pass} measures *ability* (raw pass-rate at the sample level). We emphasize: $f_{\max}$ does *not* measure correctness. An agent whose K samples all collapse to the same wrong program achieves $f_{\max} = 1$ alongside $f_{\text{pass}} = 0$ — a "stable wrong" regime that any calibration must learn to flag. The wrapper's reliability signal is therefore the joint pair $(f_{\text{pass}}, f_{\max})$ together with the dominant-class correctness indicator $\mathbb{1}\{\text{dom. class is correct}\}$.

Algorithm. Algorithm 1 summarises one application of the wrapper on a single problem.

Algorithm 1: KSampleSignature($\mathcal{A}, x, V_{\text{probe}}, \mathcal{T}_{\text{probe}}, K$)

Input: Agent $\mathcal{A}$, problem x, probe sandbox V_{probe} with probe test indices $\mathcal{T}_{\text{probe}}$, sample budget K
Output: Score pair $(f_{\text{pass}}(x), f_{\max}(x))$ and dominant probe-cluster index $j^\star$

$Y \leftarrow \emptyset$;
for $k = 1$ **to** K **do**
  $y_k \sim \pi_{\mathcal{A}}(\cdot \mid x)$;
  $\sigma_k \leftarrow \phi_{V_{\text{probe}}}(y_k)$;
  $Y \leftarrow Y \cup \{(y_k, \sigma_k)\}$;
end
Partition Y by signature σ into probe clusters $C_1, \ldots, C_J$;
$j^\star \leftarrow \arg\max_j |C_j|$; **return** $(f_{\text{pass}}, f_{\max}, j^\star)$;

Calibrated abstention via Conformal Risk Control. We use $f_{\max}$ directly as a positive concentration score, $c(x) = f_{\max}(x) \in [1/K, 1]$, where larger c means a more behaviorally concentrated — and therefore more reliable-looking — agent on x. Standard split-conformal prediction [2, 10, 17] on a label-free score controls only the marginal rate at which c falls below the threshold, not the joint silent-failure event we actually care about. We therefore use Conformal Risk Control [3]. Define the silent-failure loss

$$L(\lambda; x) \;=\; \mathbb{1}\big\{c(x) \geq \lambda \;\wedge\; \text{dominant cluster fails } V_{\text{gold}}\big\},$$

i.e. L fires when the wrapper accepts x at acceptance threshold λ and the dominant probe-cluster's representative is wrong under the disjoint gold verifier. On a held-out calibration set $\mathcal{D}_{\text{cal}}$ of size n with known correctness, we choose

$$\hat{\lambda} \;=\; \min\Big\{\lambda : \tfrac{1}{n} \sum_{x_i \in \mathcal{D}_{\text{cal}}} L(\lambda; x_i) \;\leq\; \alpha - 1/n\Big\}.$$

At test time we accept x if $c(x) \geq \hat{\lambda}$, otherwise abstain. Under exchangeability of $\mathcal{D}_{\text{cal}}$ and $\mathcal{D}_{\text{test}}$, CRC gives a finite-sample bound on the *unconditional* silent-failure rate:

$$\mathbb{E}\big[L(\hat{\lambda}; x_{\text{test}})\big] \;=\; \Pr\big(\text{accept}(x_{\text{test}}) \;\wedge\; \text{wrong}(x_{\text{test}})\big) \;\leq\; \alpha. \qquad (1)$$

Note that (1) bounds the joint event of acceptance *and* error. It does *not* guarantee a bound on the *conditional* error rate among accepted items Pr(wrong | accept) unless one calibrates a different loss; we report conditional accuracy empirically alongside the CRC-controlled silent-failure rate. Figure 2 sketches the full pipeline.

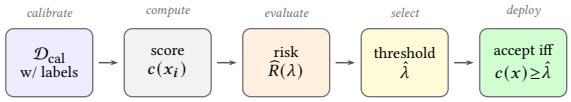


Figure 2: CRC pipeline (formulas in §2): from a labelled calibration set $\mathcal{D}_{\text{cal}}$, compute the concentration score $c(x_i) = f_{\max}(x_i)$ per problem, form the empirical silent-failure risk $\widehat{R}(\lambda)$, select the smallest $\hat{\lambda}$ satisfying the CRC bound $\widehat{R}(\hat{\lambda}) \leq \alpha - 1/n$, and at test time accept x iff $c(x) \geq \hat{\lambda}$.

3 Empirical Results

Benchmark and method scope. HumanEval+ [12] extends HumanEval [5] (164 Python problems) by augmenting each test suite with ∼80× automatically generated adversarial inputs that catch *fragile-correct* outputs. The wrapper itself is *benchmark-agnostic* — it needs only a deterministic verifier V and a signature function ϕ_V, and applies equally to math agents (proof checker; remaining-goals signature), web agents (goal-state checker; DOM-state hash), or LLM-judged QA (entailment-class indicator). We focus on code because executable verification gives the lowest-noise instantiation.

Setup. We sweep all 164 HumanEval+ problems at three decoding temperatures $T \in \{0.4, 0.8, 1.2\}$ with Qwen2.5-Coder-1.5B-Instruct, K=8 samples per problem, top-p 0.95, 512 max-new-tokens, seed 42. The evalplus 0.3.1 sandbox is applied to each plus test bank, partitioned by index into the disjoint $\mathcal{T}_{\text{probe}} \sqcup \mathcal{T}_{\text{gold}}$ of §2: $\phi_{V_{\text{probe}}}$ is the per-test bit vector on the probe half, correctness is scored only on the gold half (both truncated at the first failing test under `fast_check=True`). All four methods are paired on identical samples within each temperature; CRC uses a 60/40 cal/test split, seed 42 (n_{cal}=98, n_{test}=66); full hyperparameters in Appendix A.

Risk-coverage curve. Sweeping λ traces out the *risk-coverage curve* (Eq. (2)–(3)) on the held-out test split $\mathcal{D}_{\text{test}}$ of size n_{test}:

$$\widehat{\text{Cov}}(\lambda) = \tfrac{1}{n_{\text{test}}} \textstyle\sum_{x_i \in \mathcal{D}_{\text{test}}} \mathbb{1}\{c(x_i) \geq \lambda\}, \qquad (2)$$

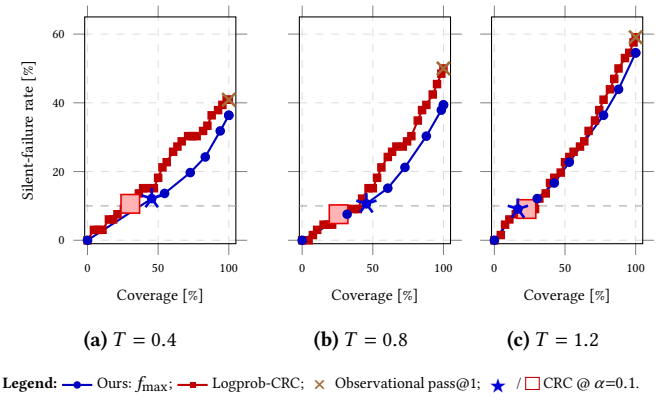
$$\widehat{R}(\lambda) = \tfrac{1}{n_{\text{test}}} \textstyle\sum_{x_i \in \mathcal{D}_{\text{test}}} \mathbb{1}\{c(x_i) \geq \lambda \;\wedge\; \text{wrong}(x_i)\}. \qquad (3)$$

$\widehat{\text{Cov}}$ is the answer rate; $\widehat{R}$ is the empirical silent-failure rate that CRC bounds at $\alpha + O(1/n_{\text{cal}})$. wrong($x_i$) is method-specific: for the wrapper, the dominant cluster's representative fails V_{gold}; for Logprob-CRC, sample 0 fails.

Four-way method comparison. Table 1 and Figure 3 summarise the comparison. Both CRC methods land within finite-sample fluctuation of the α=0.10 bound (CRC controls *expected* silent failure under exchangeability, not every realised n_{test}=66 split). At moderate temperatures $T \in \{0.4, 0.8\}$, our wrapper attains 1.65× and 1.94× the effective pass-rate of logprob CRC and largely Pareto-dominates it across the observed threshold sweep. At T=1.2 the dominant-cluster-correctness rate drops to 45% and the advantage reverses: logprob CRC's single-sample score becomes more informative for the small fraction of items that genuinely deserve acceptance. When accepting, the wrapper deploys the dominant probe-cluster's representative, *not* oracle-aided best-of-K — the target is redeployment behaviour under the agent's sampling distribution.

Table 1: Four-way comparison on HumanEval+, paired on identical K=8 samples per problem (n_{test}=66 per T; CRC at α=0.10, 60/40 split). Columns: abstention rate (Abst), effective pass-rate (Eff = answered *and* correct), silent-failure rate (Sil = Pr(accept ∧ wrong)); all in %. Green Sil cells lie within the α=0.10 risk budget (finite-sample fluctuation); red Sil cells exceed 2α. Bold Eff marks the highest effective pass-rate among the two calibrated methods.

	T=0.4			T=0.8			T=1.2		
Method	Abst	Eff	Sil	Abst	Eff	Sil	Abst	Eff	Sil
Obs. pass@1	0	59	41	0	50	50	0	41	59
Best-of-K	0	77	23	0	83	17	0	79	21
Logprob-CRC	70	20	11	74	18	8	77	**14**	9
Ours: K-sample CRC	55	**33**	12	55	**35**	11	83	8	9



(a) $T = 0.4$ **(b)** $T = 0.8$ **(c)** $T = 1.2$

Figure 3: Risk-coverage curves at three decoding temperatures (HumanEval+, Qwen2.5-Coder-1.5B-Instruct, K=8, n_{test}=66 per panel). x: coverage = answer rate [%]; y: silent-failure rate Pr(accept ∧ wrong) [%]. Solid lines trace the threshold sweep; large markers show the CRC-selected operating point at α=0.10. At $T \in \{0.4, 0.8\}$, the ours-curve largely Pareto-dominates logprob across the observed threshold sweep (lower silent failure at comparable coverage); at T=1.2, the advantage reverses (see §3).

Cross-temperature behavioural diversity. Probe-cluster structure varies monotonically with T (Appendix B, Table 2): the ≥ 2-cluster rate at K=8 grows from 55% to 68% to 83%, and dominant-cluster correctness falls from 64% to 61% to 45%. The wrapper's advantage peaks at T=0.8 (diverse enough to be informative, dominant mode still usually right) and reverses at T=1.2 (dominant mode more often *wrong* than right): wrapper is most useful at moderate diversity.

Stable-wrong diagnostic. 8–12% of problems are "stable-wrong" across T ($f_{\max}$=1 with the dominant cluster wrong on V_{gold}). $f_{\max}$ alone cannot distinguish these from stable-correct problems; CRC instead *prices them into the aggregate threshold* — stable-wrong calibration examples raise the estimated risk of high-$f_{\max}$ acceptance, forcing more abstention. This is aggregate-risk control, not per-problem rescue, but it is the right behaviour for a calibrated primitive.

4 Discussion

Reward hacking: the sample-collapse attack. A reward-hacking [16] agent can trivially maximise $f_{\max}$ by emitting the same program K times (decoding at T=0, or copying sample 1): $f_{\max}$=1 regardless of correctness, and a collapsed agent gets accepted on every problem. This is a structural property of any K-sample concentration score.

The collapse is detectable by the grader. Three grader-side mitigations close the loop without re-architecting the wrapper: *(i) diversity floor* — reject bundles whose artifact edit-distance entropy is below a threshold; *(ii) seed perturbation by the grader* — require behaviour under grader-supplied seeds before computing $f_{\max}$, applying the intervention to the grader rather than just the agent; *(iii) collapsed-cluster penalty* — combine $f_{\max}$ with the raw cluster count and reject single-cluster bundles of high pairwise similarity before CRC sees them.

Scope: agent evaluation, not RL-environment design. Our intervention is on the agent's decoding distribution, not on environment transitions: we set the agent's seed K times and read the resulting outcome distribution — a stochastic counterfactual, not a do-calculus intervention [14]. The K-sample reading generalises directly to multi-step trajectories (SWE-Bench [7] with action-selection sweeps), but that extension is out of scope here.

Equivalence-class granularity and verifier dependence. The behaviour signature ϕ_V controls which differences are treated as meaningful. Appendix C shows empirically that the per-test bit vector is necessary, not optional: under the coarse (ANY-BASE-PASS, ANY-PLUS-PASS) signature, the wrapper degenerates to 100% abstention at every T because too many distinct wrong programs collapse into the same all-fail bucket, raising CRC's calibration risk past α. Beyond signature choice, the wrapper needs a deterministic, fast V; code benchmarks satisfy this natively, while LLM-judged benchmarks add noise to ϕ_V and would need an entailment-based equivalence relation.

Future work. (a) Lift single-shot decoding to multi-step trajectories (SWE-Agent on SWE-Bench [7]) by intervening on action selection at each step — a per-decision audit; (b) study $f_{\max}$ as a reinforcement-learning objective (expected to surface new sample-collapse modes); (c) cross-benchmark transfer of $\hat{\lambda}$ via adaptive conformal recalibration [6].

5 Conclusions

K-sample behavioural equivalence is a calibrated evaluation primitive: cluster K probe-verifier signatures, feed the maximum cluster frequency $f_{\max}$ to Conformal Risk Control on gold-verifier labels, and bound expected silent-failure risk under exchangeability. On HumanEval+ at K=8 the wrapper attains 1.65–1.94× the effective pass-rate of logprob CRC at matched silent failure for moderate temperatures and surfaces a stable-wrong regime (8–12%) that single-sample baselines accept; the advantage reverses at the highest temperature tested. The contribution is *utility at matched risk*: CRC fixes the silent-failure budget; behavioural equivalence improves coverage under it.

References

[1] Yasin Abbasi Yadkori, Ilja Kuzborskij, David Stutz, András György, Adam Fisch, Arnaud Doucet, Iuliya Beloshapka, Wei-Hung Weng, Yao-Yuan Yang, Csaba Szepesvári, Ali Taylan Cemgil, and Nenad Tomasev. 2024. Mitigating LLM Hallucinations via Conformal Abstention. arXiv:2405.01563 [cs.LG] arXiv preprint arXiv:2405.01563.

[2] Anastasios N. Angelopoulos and Stephen Bates. 2021. A Gentle Introduction to Conformal Prediction and Distribution-Free Uncertainty Quantification. arXiv:2107.07511 [cs.LG] arXiv preprint arXiv:2107.07511.

[3] Anastasios N. Angelopoulos, Stephen Bates, Adam Fisch, Lihua Lei, and Tal Schuster. 2022. Conformal Risk Control. arXiv:2208.02814 [cs.LG] arXiv preprint arXiv:2208.02814.

[4] Bradley Brown, Jordan Juravsky, Ryan Ehrlich, Ronald Clark, Quoc V. Le, Christopher Ré, and Azalia Mirhoseini. 2024. Large Language Monkeys: Scaling Inference Compute with Repeated Sampling. arXiv:2407.21787 [cs.LG] arXiv preprint arXiv:2407.21787.

[5] Mark Chen, Jerry Tworek, Heewoo Jun, Qiming Yuan, Henrique Ponde de Oliveira Pinto, Jared Kaplan, et al. 2021. Evaluating Large Language Models Trained on Code. arXiv:2107.03374 [cs.LG] arXiv preprint arXiv:2107.03374.

[6] Isaac Gibbs and Emmanuel J. Candès. 2021. Adaptive Conformal Inference under Distribution Shift. In *Advances in Neural Information Processing Systems*.

[7] Carlos E. Jimenez, John Yang, Alexander Wettig, Shunyu Yao, Kexin Pei, Ofir Press, and Karthik Narasimhan. 2024. SWE-bench: Can Language Models Resolve Real-World GitHub Issues?. In *International Conference on Learning Representations (ICLR)*.

[8] Lorenz Kuhn, Yarin Gal, and Sebastian Farquhar. 2023. Semantic Uncertainty: Linguistic Invariances for Uncertainty Estimation in Natural Language Generation. In *International Conference on Learning Representations (ICLR)*.

[9] Jacky Kwok, Shulu Li, Pranav Atreya, Yuejiang Liu, Marco Pavone, Ion Stoica, and Azalia Mirhoseini. 2026. LLM-as-a-Verifier: A General-Purpose Verification Framework. Notion Blog.

[10] Jing Lei, Max G'Sell, Alessandro Rinaldo, Ryan J. Tibshirani, and Larry Wasserman. 2018. Distribution-Free Predictive Inference for Regression. *J. Amer. Statist. Assoc.* 113, 523 (2018), 1094–1111.

[11] Yujia Li, David Choi, Junyoung Chung, Nate Kushman, Julian Schrittwieser, Rémi Leblond, Tom Eccles, James Keeling, Felix Gimeno, Agustin Dal Lago, Thomas Hubert, Peter Choy, Cyprien de Masson d'Autume, Igor Babuschkin, Xinyun Chen, Po-Sen Huang, Johannes Welbl, Sven Gowal, Alexey Cherepanov, James Molloy, Daniel J. Mankowitz, Esme Sutherland Robson, Pushmeet Kohli, Nando de Freitas, Koray Kavukcuoglu, and Oriol Vinyals. 2022. Competition-Level Code Generation with AlphaCode. *Science* 378, 6624 (2022), 1092–1097.

[12] Jiawei Liu, Chunqiu Steven Xia, Yuyao Wang, and Lingming Zhang. 2023. Is Your Code Generated by ChatGPT Really Correct? Rigorous Evaluation of Large Language Models for Code Generation. In *Advances in Neural Information Processing Systems*.

[13] Christopher Mohri and Tatsunori Hashimoto. 2024. Language Models with Conformal Factuality Guarantees. In *Proceedings of the 41st International Conference on Machine Learning (ICML) (Proceedings of Machine Learning Research, Vol. 235)*. 36029–36047.

[14] Judea Pearl. 2009. *Causality: Models, Reasoning, and Inference* (2nd ed.). Cambridge University Press.

[15] Victor Quach, Adam Fisch, Tal Schuster, Adam Yala, Jae Ho Sohn, Tommi S. Jaakkola, and Regina Barzilay. 2024. Conformal Language Modeling. In *International Conference on Learning Representations (ICLR)*.

[16] Joar Skalse, Nikolaus H. R. Howe, Dmitrii Krasheninnikov, and David Krueger. 2022. Defining and Characterizing Reward Hacking. In *Advances in Neural Information Processing Systems*.

[17] Vladimir Vovk, Alexander Gammerman, and Glenn Shafer. 2005. *Algorithmic Learning in a Random World*. Springer.

[18] Xuezhi Wang, Jason Wei, Dale Schuurmans, Quoc Le, Ed Chi, Sharan Narang, Aakanksha Chowdhery, and Denny Zhou. 2023. Self-Consistency Improves Chain of Thought Reasoning in Language Models. In *International Conference on Learning Representations (ICLR)*.

A Reproducibility Setup

We list every hyperparameter and configuration choice needed to reproduce Table 1 and Figure 3.

Method note: decoding and sampling

- Model: Qwen2.5-Coder-1.5B-Instruct (fp16, MPS backend)
- Hardware: Apple M4 Pro, 24 GB unified memory
- Sampling: K=8 candidate completions per problem, `do_sample=True`, `top_p`=0.95, temperature $T \in \{0.4, 0.8, 1.2\}$
- Max new tokens: 512; PyTorch seed: 42

Method note: verifier configuration

- Sandbox: evalplus 0.3.1 [12]
- `fast_check=True` (early-exit on first failing test)
- `min_time_limit=0.2` s, `gt_time_limit_factor=2.0`
- `EVALPLUS_MAX_MEMORY_BYTES=-1` (skip macOS-incompatible `setrlimit(RLIMIT_AS)`)

Method note: CRC calibration

- Score: $c(x)=f_{\max}(x)$ for our wrapper; average per-token log-probability for Logprob-CRC baseline
- Loss: $L(\lambda; x) = \mathbb{1}\{c(x) \geq \lambda \wedge \text{wrong}(x)\}$
- Threshold: $\hat{\lambda} = \min\{\lambda : \widehat{R}(\lambda) \leq \alpha - 1/n\}$ swept over the unique calibration scores; finite-sample tolerance correction is exact [3]
- Cal/test split: 60% / 40% random permutation with seed 42 (n_{cal}=98, n_{test}=66)
- Default $\alpha = 0.10$ (sensitivity in Appendix E)

Compute budget. The full 3-temperature sweep (3 configs × 164 problems × 8 samples) consumed ∼9.5 h of wall time on the single MPS device end-to-end. Per-config breakdown: T=0.4 ∼4.6 h (inflated by a one-time multi-process incident; clean single-process estimate ∼3 h), T=0.8 ∼3.2 h, T=1.2 ∼1.7 h (faster because more wrong samples short-circuit via `fast_check`; see Appendix D).

B Per-Temperature K-Sample Diagnostics

Table 2 extends the headline Table 1 with the per-problem K-sample structural quantities cited in §3 ("Cross-temperature behavioural diversity" and "Stable-wrong diagnostic"). All rates are computed on the held-out test split (n_{test}=66) of the corresponding sweep.

Table 2: Per-temperature K-sample diagnostics on HumanEval+, K=8. *Split rate*: fraction of problems with ≥ 2 behavioural equivalence classes. *Mean* $f_{\max}$: average over test items of the largest cluster's relative frequency. *Mean clusters*: average number of distinct behavioural signatures per problem. *Dom-correct*: fraction of test items where the dominant cluster is the correct one. *Stable-wrong*: $\Pr(f_{\max}=1 \wedge$ dom-class wrong). *Fragile-correct*: fraction of problems with at least one (base-pass, plus-fail) cluster. *Wall-time* medians are clean single-process measurements (T=0.4 excluded for multi-process inflation; see Appendix A).

Diagnostic	T=0.4	T=0.8	T=1.2
Split rate (≥ 2 clusters)	55%	68%	**83%**
Mean # clusters	1.86	2.27	**2.89**
Mean $f_{\max}$	0.81	0.75	0.63
Dominant-class-correct rate	64%	61%	45%
Stable-wrong rate	**12%**	8%	9%
Fragile-correct rate	9%	15%	16%
Mean f_{pass}	0.59	0.54	0.40
Median wall-time per problem (s)	–	75.6	78.0
of which gen. (s)	–	25.3	29.6
of which exec. (s)	–	49.4	48.2

Finding: behavioural diversity scales monotonically with temperature

The fraction of problems with ≥ 2 probe-clusters grows from 55% at T=0.4 to 83% at T=1.2, and the mean number of clusters from 1.9 to 2.9. The dominant-cluster-correctness rate falls in the opposite direction (64% → 45%), bounding where the wrapper adds value.

Finding: stable-wrong appears across all temperatures

8–12% of HumanEval+ problems have $f_{\max}$=1 with the dominant probe-cluster wrong on V_{gold} — all K=8 resampled solutions agree on a gold-incorrect answer. The rate is highest at T=0.4 (12%, where the model is most prone to confident collapse) and remains 8–9% at higher T. $f_{\max}$ alone cannot distinguish these from stable-correct problems with identical concentration; CRC controls aggregate risk by raising the threshold when such examples are frequent in $\mathcal{D}_{\text{cal}}$.

C Equivalence-Class Signature: Per-Test vs. Coarse Ablation

Section 2 treats the per-test pass/fail bit-vector ($\mathbb{1}\{y \text{ passes } t_i\}$ for each test t_i in $\mathcal{T}_{\text{probe}}$) as the primary equivalence signature. An alternative coarse signature is the (ANY-BASE-PASS, ANY-PLUS-PASS) pair — four buckets total. Table 3 reports our wrapper's performance under both signatures at α=0.10.

Table 3: Per-test vs. coarse signature ablation at α=0.10. The coarse signature collapses many distinct wrong programs into the same all-fail bucket, inflating $f_{\max}$ for problems where the dominant cluster is incorrect. CRC reads the resulting calibration risk as already-too-high at every λ and abstains on every test item. The per-test signature preserves the distinction between "fails at test i" and "fails at test j", giving CRC a useful threshold.

	per-test signature (primary)			coarse (b, p) signature		
T	Abst	Eff. pass	Silent fail.	Abst	Eff. pass	Silent fail.
0.4	55%	**33%**	12%	100%	0%	0%
0.8	55%	**35%**	11%	100%	0%	0%
1.2	83%	8%	9%	100%	0%	0%

Finding: the per-test signature is load-bearing

Under the coarse (ANY-BASE-PASS, ANY-PLUS-PASS) signature, our wrapper collapses to 100% abstention at every T. The wrapper's empirical advantage in Table 1 depends critically on the per-test bit-vector preserving the distinguishability of wrong programs that fail at different test positions. This validates the colleague-reviewer's recommendation to use per-test vectors as the primary signature.

D Case Study: The Spike Region at T=0.8

During the sweep we observed that HumanEval problems 54–57 ran ~16 min each at T=0.8, while the same problems ran in ~50 s at T=0.4 and ~2 s at T=1.2. Table 4 reports the per-problem exec-time and outcome structure across the three T values.

Caveat: moderate temperature is a runtime trap

On heavy plus-test problems, T=0.8 produces correct-but-slow algorithm variants whose tests pass slowly rather than fail fast. `fast_check=True` cannot rescue this case because solutions *are* correct. This is unrelated to our wrapper's evaluation claims but a useful empirical observation for practitioners running extensive sweeps with rigorous test banks: budget for a long-tail of 10–15× slower problems at moderate temperatures, or apply a hard per-problem wall-clock cap to bound sweep runtime.

E Sensitivity to the CRC Tolerance α

Table 5 reports the four-way comparison at T=0.8 across $\alpha \in \{0.05, 0.10, 0.20, 0.30\}$. CRC stays within finite-sample fluctuation of the target at every α; the wrapper's advantage over logprob CRC is greatest at moderate α.

Table 4: Exec wall-time on the four spike problems across T values. At T=0.4 the model collapses to a single (often wrong) solution that `fast_check` truncates after a few failing tests (~ 50 s). At T=1.2 the model produces high-diversity wrong solutions that each fail at distinct early positions (~ 2 s). At T=0.8 the model finds the right *idea* but introduces algorithmic inefficiency (e.g., $O(n^2)$ where $O(n \log n)$ is the canonical), so each sample runs the full heavy plus-test bank (800–1,000 tests with large inputs).

Problem	T=0.4 exec	T=0.8 exec	T=1.2 exec	T=0.4 correctness
HumanEval/54	51 s	**986 s**	2.0 s	0/8 pass (stable wrong)
HumanEval/55	52 s	**989 s**	2.5 s	6/8 pass
HumanEval/56	54 s	**952 s**	1.9 s	8/8 pass
HumanEval/57	51 s	**753 s**	2.3 s	3/8 pass

Table 5: Effective pass-rate and empirical silent-failure rate as a function of the CRC tolerance α, on HumanEval+ at T=0.8. At very strict α=0.05, neither method can find a non-trivial acceptance threshold within the bound (both refuse $\geq$ 88% of items). At lenient α=0.30 the methods converge as abstention becomes cheap and logprob's lower coverage cost no longer matters. Our advantage is highest at the standard α=0.10.

	Logprob-CRC		Ours		ratio (eff)
α	Eff	Silent	Eff	Silent	ours / logprob
0.05	8%	5%	0%	0%	–
0.10	18%	8%	**35%**	11%	**1.94×**
0.20	30%	12%	**45%**	15%	1.50×
0.30	44%	27%	**52%**	21%	1.18×

Finding: the wrapper's advantage is highest at moderate α

At α=0.05 both methods are forced to refuse almost everything (logprob: 88% abstain; ours: 100%). At α=0.30 the methods converge (1.18× ratio) because abstention budgets are large enough that logprob's weaker score still covers acceptable items. The wrapper's value is greatest where α is small enough to demand selective deployment but large enough to permit a non-degenerate acceptance threshold.

F Cost-Utility Analysis

A natural reviewer reaction is: *"what is the wrapper's compute cost relative to its reliability gain?"* This appendix reports empirical wall-time per problem, cost per correctly-answered item (Table 6), and an empirical cost-vs-silent-failure Pareto (Figure 4) across all four methods and three temperatures.

What scales with K, and what does not. Of the four methods compared in Table 1, two require K=1 trajectory (observational pass@1 and Logprob-CRC, which only uses sample 0's average log-probability) and two require the full K=8 trajectories (Best-of-K and our wrapper). Generation scales *sub*-linearly in K because the K candidates share the prefill pass and are produced by a single batched decode (see §1); verifier execution scales linearly in K but each y_k is independent and parallelisable. Empirically, on Qwen2.5-Coder-1.5B at K=8 over the clean single-process T=0.8 run, median wall-time per problem is 75.6 s with a K=8 trajectory

Table 6: Cost-utility comparison across methods and decoding temperatures. "Wall" is median wall-time per problem (s). "Eff" is effective pass-rate (% test items answered and correct). "$/correct" is the implied compute cost per correctly answered item = Wall / Eff (s per correct). "Silent" is empirical $\Pr(\text{accept} \wedge \text{wrong})$ on the test split. The T=0.4 wall-time uses the clean single-process T=0.8 figure to avoid the multi-process inflation noted in Appendix A. Shading follows Table 1: green Sil cells lie within the α=0.10 risk budget, red Sil cells exceed 2α; bold Eff marks the highest effective pass-rate among the two calibrated methods (darker green).

	T=0.4				T=0.8				T=1.2			
Method	Wall	Eff	$/cor	Sil	Wall	Eff	$/cor	Sil	Wall	Eff	$/cor	Sil
Obs. pass@1	12.7	59	22	41	12.7	50	25	50	13.4	41	33	59
Best-of-K	75.6	77	98	23	75.6	83	91	17	78.0	79	99	21
Logprob-CRC	12.7	20	64	11	12.7	18	71	8	13.4	**14**	96	9
Ours: K-sample CRC	75.6	**33**	229	12	75.6	**35**	216	11	78.0	8	972	9

and an estimated 12.7 s with K=1 — a 5.9× overhead rather than 8×.

Diminishing returns above K≈10. We use K=8 because (i) $f_{\max} \in \{1/K, 2/K, \ldots, 1\}$ already has 8 levels of resolution at K=8, enough to detect a 3:1 split and finer; (ii) marginal information from additional samples flattens above K≈10–20 in the repeated-sampling analyses of Brown et al. [4]; and (iii) compute scales linearly in the verifier dimension, which dominates the wall-time (Table 2). Pushing K further would yield smaller returns at proportionally higher cost — a reviewer's "K=10^{10}" would not change our finite-sample silent-failure risk (CRC already stays within fluctuation of the α target at K=8) and would not meaningfully improve effective pass-rate beyond what saturates by K≈10.

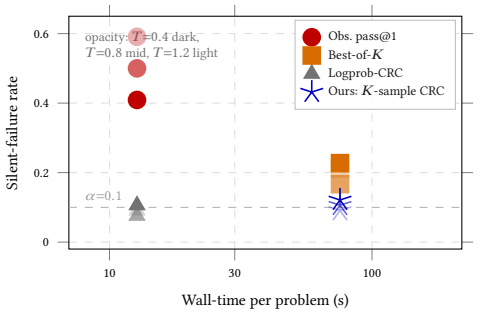


Figure 4: Empirical cost-vs-silent-failure Pareto across 4 methods × 3 temperatures (marker opacity: darker = lower T). Lower-right is the deployment frontier: low silent failures *and* low cost. Observational pass@1 is cheap but has the highest silent-failure rate. Best-of-K pays the K=8 cost without abstention; its silent rate (17–23%) is below pass@1 but well above α=0.10. Logprob-CRC stays within finite-sample fluctuation of the α-bound at K=1 cost (cheapest reliability-correct method). Ours achieves the same silent-failure target at K=8 cost — buying additional *effective pass-rate* (Table 6), not lower silent failure.

Finding: at moderate T, ours buys utility at matched risk

At α=0.10 and $T \in \{0.4, 0.8\}$, ours and Logprob-CRC achieve comparable silent-failure rate (within finite-sample fluctuation), but ours answers 1.65–1.94× more items correctly per problem than Logprob-CRC — so the higher per-correct compute cost is spent producing more correct outputs at the same risk target, not extra reliability over the α bound. At T=1.2, where the dominant-cluster-correctness rate falls to 45%, the wrapper's eff-pass drops to 8% and its cost-per-correct rises sharply — a limitation noted in Table 1. The cost-utility framing is *utility at matched risk*: CRC fixes the silent-failure budget; behavioural equivalence improves coverage under that budget when clustering is informative.

Caveat: when ours is *not* the right pick

If silent failures are cheap (e.g., a developer reviews every output anyway) and abstention is expensive (review cost dominates), then observational pass@1 is the cost-optimal choice and our wrapper is overkill. If silent failures are catastrophic (high-stakes deployment) and review is cheap, then ours dominates — the K=8 overhead is amortised against the avoided silent-failure cost. The wrapper's value depends on the application's $C_{\text{silent}}/C_{\text{review}}$ ratio; we provide the calibrated α knob so practitioners can dial that tradeoff.