INFERENCE-TIME AMPLIFICATION OF WEAK REASONING MODELS

Anonymous authors
Paper under double-blind review

ABSTRACT

How much of the capability of a reasoning model exposed by repeated sampling can an imperfect selector recover? We study this problem as *inference-time amplification*, separating *coverage*—whether a useful candidate is generated—from *identifiability*—whether it can be recognized among the alternatives. We show that better coverage alone does not guarantee better selection, while even imperfect local selection signals can be amplified through repetition. We also characterize proposal-side blind spots that additional sampling cannot overcome. On SWE-bench Verified, a single GPT-5.4 nano trajectory solves 67.0% of tasks, while eight proposals contain a correct patch on 79.0%; critic–comparator selection reaches 76.4%, recovering 78.3% of the oracle-exposed gain. The same pattern appears across proposer families and in mathematical reasoning, competitive programming, and formal proof. On MATH L4–5 and GSM-Plus, our selector recovers 85% and 77% of the oracle gap and substantially outperforms majority vote. Holding proposal coverage fixed, providing the selector with additional repository information also improves recovery. Conversely, when the available selection signal is weak, redundant, or misaligned, amplification can diminish or fail. These results identify selection quality as the key factor governing how much capability exposed by repeated sampling can actually be recovered.

1 INTRODUCTION

A reasoning model may solve a problem on one attempt and fail on the next. Repeated sampling exposes this intermittent capability, but using it requires selecting the correct output. The central question of this paper is therefore:

When can an imperfect selector recover the capability exposed by repeated sampling?
Can we model that mathematically?

On SWE-bench Verified (Jimenez et al., 2024), a single `GPT-5.4 nano` trajectory solves 67.0% of tasks, while among eight independent trajectories at least one correct patch appears on 79.0%.

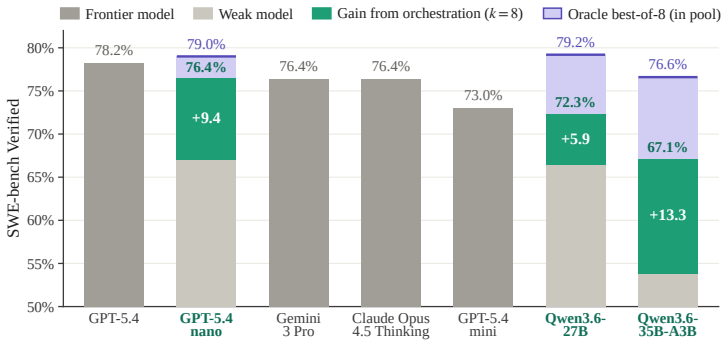


Figure 1: **Inference-time amplification on SWE-bench Verified.** With eight `GPT-5.4 nano` proposals, oracle coverage reaches 79.0% and our selector reaches 76.4%. The 67.0% nano value is a separate single-trajectory reference; stronger models are shown for context.

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.

Our selector reaches 76.4%. This gap exposes substantial *latent proposal-pool capability*: correct solutions are often generated even when a single generation fails, but their presence is not enough. The system must still identify the correct candidate without access to the hidden verifier.

This is the selection problem at the heart of inference-time amplification. Existing methods use majority vote, self-consistency, reward models, learned verifiers, LLM-as-a-judge, or task-specific checks to choose among sampled candidates (Wang et al., 2023; Chen et al., 2023; Cobbe et al., 2021; Lightman et al., 2024; Jiang et al., 2023; Huang et al., 2025). Yet an oracle best-of-k result only shows that a useful candidate exists; it says nothing about whether a deployable selector can recognize it. Likewise, sampling alone does not solve the problem: majority voting can fail when correct answers are diverse or infrequent, and a learned judge may itself be weak or misaligned.

To answer our question, we separate two requirements. *Coverage* asks whether the proposal pool contains a useful candidate; *identifiability* asks whether the available evidence makes that candidate distinguishable from the alternatives. These are logically different resources. More samples can increase coverage without providing any information about which sample is useful. We make this separation formal by constructing task families with arbitrarily high proposal coverage but no uniform positive critic or comparator edge. Thus repeated sampling creates an opportunity for amplification, but cannot by itself realize it.

The positive result is that the selection signal need not be strong. Suppose that, on a useful–unhelpful pair, a comparator chooses the useful candidate with probability at least $1/2 + \sigma$, for some $\sigma > 0$. Repeated independent comparisons amplify this weak local edge; critics can analogously filter candidates using unary evidence. When the proposal mechanism also has nonzero useful mass, repetition therefore amplifies both sides of the problem: sampling increases the chance that a useful candidate appears, while repeated selection increases the chance that it is recovered. We then show how these local guarantees compose along adaptive reasoning trajectories when useful actions make bounded progress toward a verified solution.

This decomposition also identifies the fundamental ceiling. If no useful candidate is generated, no selector can recover one. We characterize such proposal-side blind spots and show that oracle best-of-k converges to a support-limited ceiling rather than necessarily to 100%. At finite k, the oracle gap and the implemented selector together distinguish two failure modes: *proposal failure*, where no correct candidate appears, and *selection failure*, where a correct candidate appears but is not chosen. This makes oracle-gap recovery a natural measure of how much sampled capability a practical selector actually realizes.

Contributions. Our theoretical and empirical results support this picture.

(i) **We characterize the conditions and limits of inference-time amplification.** We separate proposal *coverage*, local *identifiability*, and end-to-end *progress*. Coverage alone is insufficient: even arbitrarily high useful-proposal probability need not provide any information about which proposal is useful. Conversely, any uniformly positive local comparison edge can be amplified by repetition, and bounded progress composes these local guarantees into end-to-end success. We further characterize proposal-side blind spots that impose an irreducible ceiling on amplification.

(ii) **We show that repeated sampling exposes substantial capability that practical selectors can recover.** On SWE-bench Verified, eight `GPT-5.4 nano` proposals raise the oracle from 67.0% pass@1 to 79.0%, while critic–comparator selection reaches 76.4%, recovering 78.3% of the exposed gain. On MATH L4–5 and GSM-Plus, selection recovers 85% and 77% of the oracle gap and substantially outperforms majority vote. A controlled GSM-Plus scaling study further shows that pairwise comparison continues to recover roughly 77–80% of the gain exposed as the proposal pool grows.

(iii) **We show that identifiability determines whether exposed capability becomes usable.** Across domains, stronger measured comparison edges coincide with greater oracle-gap recovery. More directly, holding a Qwen3.6-27B SWE-bench proposal pool fixed, repository context raises the comparator edge from 0.04 to 0.09 and recovery from 24% to 40%. Conversely, direct verification can make learned selection unnecessary, simple heuristics can outperform a weak selector, and a misaligned comparator can reduce performance. Thus a large oracle gap creates an opportunity for amplification, not a guarantee.

2 Related Work

Boosting and inference-time amplification. Classical boosting converts weak predictive edges into strong predictors under supervised feedback (Schapire, 1990; Freund & Schapire, 1995; 1997). Related work studies LLMs as weak learners, weak-to-strong generalization, and boosting-style model combinations (Manikandan et al., 2023; Burns et al., 2024; Chen et al., 2025). Our setting is different: model weights remain fixed, and amplification must arise entirely at inference time from repeated proposal and imperfect local selection.

Sampling and test-time scaling. Self-consistency, rationale ensembles, multi-agent voting, and repeated sampling improve performance by generating multiple candidates (Wang et al., 2023; 2022; Chen et al., 2023; Li et al., 2024; Brown et al., 2024). Recent work studies scaling with additional model calls and best-of-N inference (Chen et al., 2024; Huang et al., 2025; Zhang et al., 2025). We separate the capability exposed by sampling from the ability to recover it: a correct candidate can be present in the pool without being identifiable by the available selector.

Selection, verifiers, and judges. Learned verifiers, process reward models, pairwise rankers, verifier ensembles, LLM judges, and tool-backed checking provide increasingly sophisticated ways to select among candidate solutions (Cobbe et al., 2021; Lightman et al., 2024; Wang et al., 2024; Jiang et al., 2023; Lifshitz et al., 2025; Saad-Falcon et al., 2025a; Zheng et al., 2023; Lambert et al., 2025; Gou et al., 2024; Dhuliawala et al., 2024). Rather than proposing another scoring rule, we ask when an *imperfect* selection signal is sufficient for amplification. Our analysis formalizes this through local identifiability: a weak positive comparison edge can be amplified by repetition, whereas proposal coverage alone cannot create such an edge.

Search, agents, and verifier-backed reasoning. Tree of Thoughts, RAP, LATS, ReAct, Reflexion, Self-Refine, and related compound or multi-agent systems use search, feedback, planning, and tool interaction over intermediate states (Yao et al., 2023a; Hao et al., 2023; Zhou et al., 2024; Yao et al., 2023b; Shinn et al., 2023; Madaan et al., 2023; Li et al., 2023; Wu et al., 2024; Hong et al., 2024; Wang et al., 2025; Saad-Falcon et al., 2025b; Guha et al., 2024). Code and formal reasoning make selection especially measurable because tests, execution, types, or proof checkers can distinguish candidate artifacts (Li et al., 2022; Chen et al., 2021; Jimenez et al., 2024; Yang et al., 2024; Zhang et al., 2024; Xia et al., 2025). Related work on debate, prover–verifier games, and scalable oversight also uses weaker local judgments to support harder decisions (Irving et al., 2018; Christiano et al., 2018; Du et al., 2024; Liang et al., 2024; Brown-Cohen et al., 2024; Anil et al., 2021; Kirchner et al., 2025; Kenton et al., 2024). We isolate a narrower mechanism: how proposal coverage, local identification edges, and bounded progress combine to make inference-time amplification possible.

3 Verifier-Backed Committee Search

We formalize inference-time amplification as a selection problem. Repeated proposal calls may produce a useful next action, but the system must identify it without access to the final hidden verifier. The theory therefore separates *proposal coverage* from *local identifiability*.

We model reasoning as search over partial solutions. For an input x, a state records the current partial solution, an action transforms that state, and a terminal state produces an output checked by a final verifier.

Definition 1 (Valid-state system)**.** *For an input x, let $\mathcal{S}_x$ be a countable state space with initial state $s_0(x)$, valid states $\mathrm{Valid}_x \subseteq \mathcal{S}_x$, terminal states $\mathrm{Term}_x \subseteq \mathrm{Valid}_x$, action sets $\mathcal{A}_x(s)$, and transition map $\Phi_x(s,a)$. Let $\mathrm{Out}_x : \mathrm{Term}_x \to \mathcal{Y}_x$ map terminal states to outputs, and let $R_x : \mathcal{Y}_x \to \{0,1\}$ be the terminal verifier.*

A rank function $d_x : \mathrm{Valid}_x \to \{0,1,\dots,L_x\}$ satisfies $d_x(s) = 0$ on terminal states and $d_x(s) \geq 1$ on valid nonterminal states. We require $s_0(x) \in \mathrm{Valid}_x$ and $R_x(\mathrm{Out}_x(s)) = 1$ for every $s \in \mathrm{Term}_x$.

A state is *reachable* if it can be obtained from $s_0(x)$ by a finite sequence of actions. At a reachable valid nonterminal state s, call an action *progressing-sound* if it preserves validity and strictly decreases the rank:

$$\mathrm{Sound}_x(s) := \{a \in \mathcal{A}_x(s) : \Phi_x(s,a) \in \mathrm{Valid}_x,\ d_x(\Phi_x(s,a)) < d_x(s)\}\,.$$

Thus any trajectory of progressing-sound actions reaches a terminal state within at most L_x steps.

This notion separates local usefulness from final verification. In SWE-bench, for example, a progressing-sound edit need only preserve the existence of a hidden-test-passing completion while making progress toward it. Visible tests, type checks, linters, or learned judgments may provide local evidence without certifying final correctness.

Committee protocol. At each reachable valid nonterminal state, $\Pi_{k,m,r}$ draws k candidate actions, applies m independent critic calls to each, and removes any candidate rejected by a critic. If none survives, the protocol fails. Otherwise, each surviving pair receives r comparator votes; majority vote orients each pair, and a Copeland winner is selected and applied. The process repeats until a terminal state is reached.

At termination the protocol returns the corresponding output under Out_x. We write $\perp$ for the no-survivor failure and define $R_x(\perp) = 0$.

Local success requires two conditions: a progressing-sound action must appear in the pool, and the selection signal must distinguish it from unsound alternatives. We formalize these as coverage and identifiability.

Assumption 1 (Per-state local coverage)**.** *For each problem size N, there is a proposer portfolio P_N of polynomial size such that, for every reachable valid nonterminal state s of every size-N input, at least one proposer $p_s \in P_N$ generates a progressing-sound action with probability at least some constant $\alpha_0 > 0$.*

Assumption 1 guarantees that useful actions have nontrivial probability of appearing, but says nothing about which proposal is useful. Sampling can therefore amplify coverage without solving selection.

Assumption 2 (Efficient local identifiability)**.** *At every reachable valid nonterminal state s, there are polynomial-time randomized procedures Crit_s and Comp_s, and constants $\beta_0, \sigma_0 > 0$, such that:*

(i) if $a \in \mathrm{Sound}_x(s)$, then $\mathsf{Crit}_s(a)$ never rejects;
(ii) if $a \notin \mathrm{Sound}_x(s)$, then $\mathbb{P}(\mathsf{Crit}_s(a) = \textsc{reject}) \geq \beta_0$; and
(iii) if $a \in \mathrm{Sound}_x(s)$ and $b \notin \mathrm{Sound}_x(s)$, then $\mathbb{P}(\mathsf{Comp}_s(a, b) = a) \geq \frac{1}{2} + \sigma_0$.

Assumption 2 supplies the information needed for selection. In particular, the comparator need only have a positive edge σ_0 on progressing-sound–unsound pairs; repetition will amplify this weak preference.

Assumption 1 concerns generation; Assumption 2 concerns selection. Neither is enough on its own. The one-sided critic is a clean sufficient condition rather than a claim about practical learned critics; Appendix A.2 allows false rejections. We next show why additional sampling cannot create an identification signal, and then show how any positive signal can be amplified.

4 Coverage and Identifiability

More samples can make a useful action increasingly likely to appear without making it any easier to recognize. Coverage and identifiability are therefore separate resources: the former is supplied by the proposer, while the latter requires information about candidate quality beyond their mere occurrence.

Proposition 1 (Coverage does not imply local identifiability)**.** *For every $M \geq 2$, there is a one-step task family with action set $\mathcal{A} = \{1, \ldots, M\}$ and hidden parameter $\theta \in \{1, \ldots, M\}$ such that the proposer is $\mathrm{Unif}(\mathcal{A})$ in every world and $\mathrm{Sound}_\theta = \mathcal{A} \setminus \{\theta\}$. Even with arbitrarily many samples from this proposer, a selector that receives no information about θ cannot have a uniform positive critic or comparator edge over all worlds.*

The construction isolates the obstruction. A proposal is sound with probability $1 - 1/M$, so useful actions are abundant, yet the proposal distribution is identical in every hidden world. Additional samples therefore increase neither the information available to a critic nor the edge of a comparator. Reliable selection requires an additional signal—for example execution, tests, proof checking, type information, constraints, or a learned reviewer with task-relevant information.

Sampling can nevertheless amplify the part of the problem it controls: coverage.

Theorem 1 (Coverage amplification from a proposer portfolio)**.** *Fix a reachable valid nonterminal state s and suppose Assumption 1 holds. Query the proposers in P_N in round-robin order with fresh independent randomness. After k calls, the probability that the candidate pool contains a progressing-sound action is at least $1 - (1 - \alpha_0)^{\lfloor k/|P_N| \rfloor}$. In particular, this probability is at least $1 - \delta_{\text{prop}}$ whenever $k \geq |P_N| \lceil \alpha_0^{-1} \log(1/\delta_{\text{prop}}) \rceil$.*

Theorem 1 says precisely what repeated sampling can guarantee: a useful candidate becomes increasingly likely to appear. It says nothing about whether the system can recognize that candidate once it is present. That is the separate role of identifiability.

Local verification as a source of identifiability. The selection signal need not come from a stronger learned model. A local verifier can provide it directly. The important distinction is that a verifier of completed outputs is not automatically useful for choosing among intermediate candidates; it must provide information about the candidates being compared.

Assumption 3 (One-sided local verifier)**.** *At every reachable valid nonterminal state s, there is a polynomial-time randomized verifier $V_x(s, a) \in \{\text{ACCEPT}, \text{REJECT}\}$ that accepts every $a \in \text{Sound}_x(s)$ almost surely and rejects every $a \notin \text{Sound}_x(s)$ with probability at least $1 - \nu$.*

Such a verifier immediately supplies both forms of local evidence. It can be used directly as a critic, and it induces a comparator by verifying both candidates, choosing the unrejected candidate when exactly one is rejected, and otherwise breaking the tie uniformly.

Corollary 1 (Local verification implies identifiability)**.** *Under Assumption 3, Assumption 2 holds with $\beta_0 = 1 - \nu$ and $\sigma_0 = (1 - \nu)/2$.*

The verifier can be used directly as a critic. It also induces a comparator: verify both candidates, choose the unrejected one when exactly one is rejected, and otherwise break the tie uniformly. On a sound–unsound pair this gives comparison success at least $1/2 + (1 - \nu)/2$. Thus amplification requires an informative signal, not necessarily a powerful learned selector. A weak positive signal is enough for repetition to help.

Appendix A.1 gives the formal proof and shows why a verifier that checks only completed outputs need not provide the same local information.

5 Local-to-Global Amplification

We now give the positive amplification result. A local decision can fail for two reasons: no progressing-sound action is proposed, or such an action is proposed but the selector fails to recover it. The first is a coverage error; the second is an identification error. Repetition suppresses these two errors through different mechanisms.

Fix a reachable valid nonterminal state s. Let $\varepsilon_{\text{prop}}(k; s)$ denote the probability that none of the k proposals lies in $\text{Sound}_x(s)$, and let $\varepsilon_{\text{loc}}(s)$ denote the probability of local failure, conditional on the protocol being at s: either no candidate survives or the selected action lies outside $\text{Sound}_x(s)$.

Theorem 2 (Local error decomposition)**.** *Suppose the local-identifiability conditions hold at s with critic edge β and comparator edge σ, and let r be odd. Conditional on the current state and the proposed candidates, assume that repeated critic calls are independent, that repeated comparator votes for each pair are independent, and that the comparator randomness is fresh relative to the critic randomness. Then*

$$\varepsilon_{\text{loc}}(s) \leq \varepsilon_{\text{prop}}(k; s) + k^2(1 - \beta)^m e^{-2r\sigma^2} \leq \varepsilon_{\text{prop}}(k; s) + k^2 e^{-\beta m - 2r\sigma^2}.$$

If, in addition, the k proposal calls are conditionally independent and each is progressing-sound with probability at least α, then $\varepsilon_{\text{prop}}(k; s) \leq (1 - \alpha)^k \leq e^{-\alpha k}$.

The bound makes the amplification mechanism explicit. Increasing k reduces the probability that no useful action appears. Once one is present, critic repetition suppresses unsound survivors as $e^{-\beta m}$, while repeated comparison amplifies even a weak positive edge $\sigma > 0$, with error decaying as $e^{-2r\sigma^2}$. Thus the selector need not be highly accurate in a single comparison: a uniformly

better-than-chance local preference can be amplified by repetition. The factor k^2 comes from a conservative union bound over sound–unsound candidate pairs.

This establishes reliable selection at one state. To obtain an end-to-end guarantee, these local decisions must compose. If every selected action is progressing-sound, validity is preserved and the integer-valued rank decreases strictly, so the protocol reaches a terminal state within at most L_x steps.

Define the end-to-end failure probability $\mathrm{err}_x(k, m, r) := \mathbb{P}(R_x(\Pi_{k,m,r}(x)) = 0)$.

Lemma 1 (Adaptive cumulative-error bound)**.** *Let $\mathcal{F}_t$ denote the history available before the decision at step t. Suppose that, whenever the protocol is at a reachable valid nonterminal state, the conditional probability of local failure at step t is at most ε_t. Then $\mathrm{err}_x(k, m, r) \le \mathbb{E}[\sum_{t=0}^{L_x-1} \varepsilon_t]$. In particular, if $\varepsilon_t \le \varepsilon$ uniformly along the trajectory, then $\mathrm{err}_x(k, m, r) \le L_x\varepsilon$.*

No independence across reasoning steps is required: the state, proposal distribution, and local error rate may depend arbitrarily on the preceding trajectory. It is enough that the conditional failure probability is controlled at each state actually reached.

Corollary 2 (Global amplification bound)**.** *Suppose $\varepsilon_{\mathrm{prop}}(k; s) \le \varepsilon_{\mathrm{prop}}(k)$ uniformly over reachable valid nonterminal states, and suppose the critic and comparator edges are uniformly bounded below by β and σ. Then*

$$\mathrm{err}_x(k, m, r) \le L_x\left(\varepsilon_{\mathrm{prop}}(k) + k^2 e^{-\beta m - 2r\sigma^2}\right).$$

Consequently, the protocol succeeds with probability at least $1 - \delta$ whenever the local proposal-plus-selection error in parentheses is at most δ/L_x.

The corollary formalizes the central claim: inference-time amplification requires both useful proposals and an informative selector. Proposal width k controls whether a useful action appears; critic and comparator budgets m, r control whether it is recovered; and trajectory depth L_x determines how local errors accumulate. Explicit polynomial-resource choices appear in Appendix B.

6 Blind Spots and Oracle Ceilings

Selection can recover only capability that the proposer exposes. Even a perfect selector cannot help when the proposal mechanism assigns zero probability to a useful action. We therefore ask how far repeated sampling alone can reduce the coverage term $\varepsilon_{\mathrm{prop}}(k; s)$.

Lemma 2 (Oracle miss and blind-spot floor)**.** *Fix a reachable valid nonterminal state s. Let Z capture latent heterogeneity shared across proposal calls. Conditional on Z, suppose the k calls are independent and each proposes a progressing-sound action with probability $q_s(Z)$. Then*

$$\varepsilon_{\mathrm{prop}}(k; s) = \mathbb{E}[(1 - q_s(Z))^k] = B_s + R_k(s),$$

where $B_s := \mathbb{P}(q_s(Z) = 0)$ and $R_k(s) := \mathbb{E}[(1 - q_s(Z))^k \mathbf{1}\{q_s(Z) > 0\}]$. Moreover, $R_k(s) \to 0$, so $\varepsilon_{\mathrm{prop}}(k; s) \to B_s$.

Here $R_k(s)$ is finite-sample error: a useful action has positive probability but happens not to appear in the k draws. It vanishes as k grows. By contrast, B_s represents cases on which the proposer has zero probability of producing a useful action. More samples from the same proposer cannot fix those cases. Consequently, finite-k oracle failure upper-bounds rather than identifies the asymptotic blind-spot floor. Heterogeneous portfolios and finite-k rates appear in Appendix A.3.

Let $B := \sup_s B_s$ and $R_k := \sup_s R_k(s)$, where the supremum ranges over reachable valid nonterminal states. Combining Lemma 2 with Corollary 2 gives

$$\mathrm{err}_x(k, m, r) \le L_x\Bigg[\underbrace{B}_{\text{blind-spot floor}} + \underbrace{R_k}_{\text{finite-width proposal error}} + \underbrace{k^2 e^{-\beta m - 2r\sigma^2}}_{\text{selection error}}\Bigg].$$

This yields the full decomposition. More proposal samples reduce finite-width coverage error; more critic or comparator effort reduces selection error; and neither can remove a support-level blind spot.

Once selection error is small, further improvement must come from proposal coverage rather than additional judging.

Terminal oracle curves. The same decomposition has a direct task-level analogue in our experiments. Let $X \sim \mathcal{D}$ and $q_P(x) := \mathbb{P}_{Y \sim P(\cdot|x)}(R_x(Y) = 1)$. Then pass@1 is $p_1(P) = \mathbb{E}[q_P(X)]$, while oracle best-of-$k$ is $p_{\text{oracle}}(k; P) = \mathbb{E}[1 - (1 - q_P(X))^k]$. As $k \to \infty$, the oracle converges to $1 - \mathbb{P}(q_P(X) = 0)$.

Pass@1 therefore measures one-shot proposal success, whereas oracle best-of-k measures how often a correct output is present in a k-candidate pool. Their gap is the capability exposed by additional sampling under perfect selection. At finite k, however, oracle failure still combines true blind spots with ordinary finite-sampling misses; even an observed plateau is evidence of saturation rather than a direct estimate of the asymptotic blind-spot mass.

Oracle-gap recovery. For a selector evaluated on the same k-candidate pool, let $p_{\text{system}}(k, m, r; P)$ denote its success probability. Whenever $p_{\text{oracle}}(k; P) > p_1(P)$, define

$$\text{Rec}(k, m, r; P) := \frac{p_{\text{system}}(k, m, r; P) - p_1(P)}{p_{\text{oracle}}(k; P) - p_1(P)}.$$

The denominator measures the capability exposed by sampling under perfect selection; the numerator measures how much of that gain the implemented selector realizes. Oracle-gap recovery therefore directly connects the theory to the experiments: a large oracle gap with low recovery identifies a selection bottleneck, whereas high recovery shifts the remaining bottleneck toward proposal coverage.

7 Experiments

Our experiments measure separately whether repeated sampling produces a correct candidate and whether the selector can recover it. Oracle best-of-k is used only as a hindsight measure of proposal-pool coverage and is never available to the selector.

7.1 Setup

SWE-bench Verified contains 500 repository-level software-engineering tasks scored by held-out tests. In the headline experiment, we generate $k = 8$ independent `GPT-5.4 nano` Mini-SWE-agent trajectories per task and select among their patches using binary critics followed by pairwise comparators. Hidden-test outcomes are used only for final evaluation and oracle diagnostics; they are never exposed to the selector.

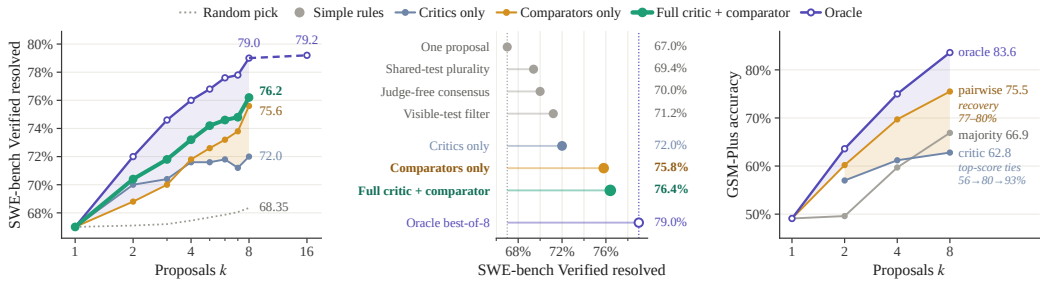
A single headline trajectory resolves 67.0% of tasks. Full critic–comparator orchestration reaches 76.4%, while an oracle that chooses a passing patch whenever one is present reaches 79.0%. The implemented system therefore recovers 78.3% of the improvement exposed by eight proposals. Among the tasks the system still misses, 89.0% contain no passing patch in the eight-proposal pool and 11.0% are oracle-reachable selection misses. Thus, at this budget, the dominant residual is proposal coverage rather than selection.

Additional analyses reuse fixed archived proposal and judgment pools; matched baseline definitions, cache provenance, prompts, and implementation details appear in Appendix C.

7.2 Sampling Creates Opportunity; Selection Determines Recovery

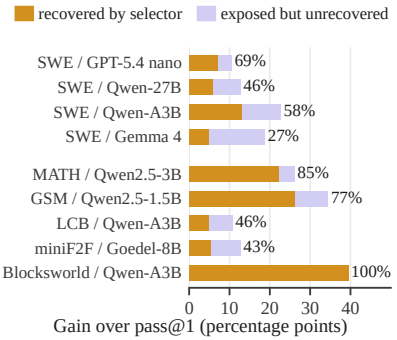
Figure 2 shows the coverage–selection decomposition directly. On SWE-bench, increasing k expands oracle coverage, but selectors exploit that opportunity differently: pointwise criticism saturates earlier, while pairwise comparison continues to improve. Simple rules also leave substantial oracle headroom at $k = 8$. Doubling the proposal budget from $k = 8$ to 16 raises the oracle only from 79.0% to 79.2%, showing sharply diminishing finite-width returns without identifying the asymptotic blind-spot floor.

On GSM-Plus, pairwise comparison recovers roughly 77–80$\%$ of the oracle-exposed gain across $k = 2, 4, 8$, whereas critic-only recovery falls as the top-score tie rate rises from 56% to 93%. This loss of score resolution explains why pointwise criticism benefits less from larger proposal pools. Full subset-averaged results and bootstrap intervals appear in Appendix D.

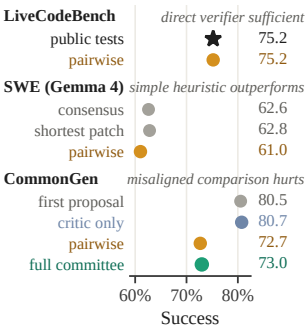


(a) SWE-bench scaling (b) SWE-bench at $k = 8$ (c) GSM-Plus scaling

Figure 2: **Sampling exposes capability; selection determines how much is recovered.** (a) On SWE-bench Verified, increasing proposal width expands oracle coverage, but selectors differ substantially in how well they exploit the larger pool. (b) At $k = 8$, simple selection rules leave substantial oracle-exposed capability unused, while pairwise comparison closes most of the remaining gap. (c) On GSM-Plus, pairwise selection continues to recover a nearly constant fraction of the capability exposed as the proposal pool grows, while pointwise criticism increasingly loses resolution.



(a) Exposed and recovered gain



(b) Boundary cases

	Patch only	+ Repo.	Δ
Comparator edge $\hat{\sigma}$	0.04	0.09	+0.05
Solve rate	67.0%	69.6%	+2.6 pp
Oracle-gap recovery	24%	40%	+16 pp

Proposal pool and oracle held fixed
Qwen3.6-27B; oracle = 79.2%

(c) Fixed-coverage intervention

Figure 3: **Identifiability governs recovery of sampled capability.** (a) Across proposer–task configurations, repeated sampling can expose substantial additional capability, but selectors recover widely varying fractions of that opportunity. (b) Boundary cases show that learned selection can be redundant, inferior to a simple heuristic, or harmful when its signal is misaligned with correctness. (c) Holding both the SWE-bench proposal pool and oracle coverage fixed, providing repository context improves the comparator edge, end-to-end solve rate, and oracle-gap recovery.

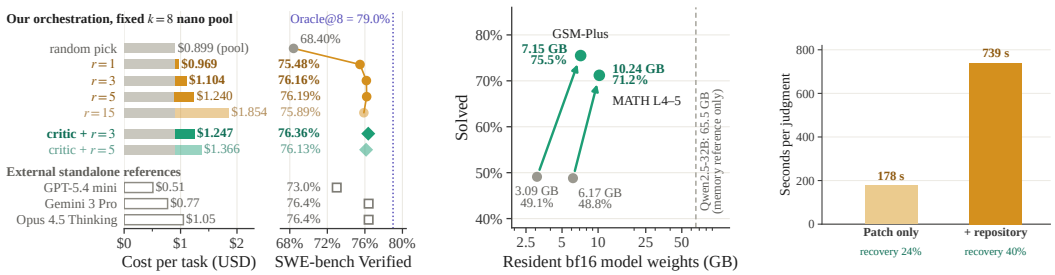
7.3 Identifiability Governs Recovery

For empirical comparator diagnostics, let τ be the observed tie rate on correct–incorrect candidate pairs and let a be accuracy conditional on a decisive judgment. We report $\hat{\sigma} := (1 - \tau)(a - \frac{1}{2})$; equivalently, assigning ties half credit gives overall pairwise accuracy $\frac{1}{2} + \hat{\sigma}$, matching the comparator edge in Assumption 2.

A large oracle gap is an opportunity for amplification, not a guarantee. Figure 3a shows that the fraction of sampled capability recovered by selection varies widely across proposer–task configurations: MATH and GSM recover most of the exposed gain, several SWE-bench pools recover far less, and task-aligned verification on Blocksworld recovers essentially all of it. Full cross-model curves and controls appear in Appendix D.

Figure 3b shows three ways this mechanism can break down. Direct verification can make learned judging unnecessary; a simple heuristic can outperform a weak judge; and a misaligned comparator can make selection worse than doing nothing.

Figure 3c gives a cleaner test of the role of identifiability. We hold the Qwen3.6-27B proposal pool and its 79.2% oracle fixed and change only the information available to the comparator. Repository



(a) Cost–accuracy tradeoff (b) Resident model footprint (c) Cost of richer evidence

Figure 4: **Capability–resource tradeoffs.** (a) Judge cost shows diminishing returns on a fixed SWE-bench pool; external systems are contextual, unmatched references. (b) Small proposer–judge combinations improve accuracy on MATH and GSM-Plus with smaller resident model footprints. (c) Repository access improves selection at higher judgment latency.

access raises $\hat{\sigma}$ from 0.04 to 0.09, solve rate from 67.0% to 69.6%, and oracle-gap recovery from 24% to 40%. Better information therefore improves selection without changing what was generated.

The effect is not specific to software repair. On MATH L4–5, selection reaches 71.2% from 48.8% pass@1 against a 75.0% oracle. On miniF2F-hard it reaches 25.0% from 19.5%, toward a 32.3% oracle. The latter result is suggestive rather than conclusive ($p = 0.064$).

7.4 Resource and Deployment Tradeoffs

Amplification uses resources in different ways. More proposals increase the chance that a correct solution appears, while judging and additional task context help identify it. Orchestration can also substitute additional inference-time computation for a larger resident model.

On the fixed SWE-bench proposal pool, most of the selection gain appears at low marginal cost. One comparator vote per pair reaches 75.48% at $\$0.97$ per task, and three votes reach 76.16% at $\$1.10$; additional judging brings little further improvement. For context, the external Claude Opus 4.5 Thinking reference reports 76.4% at $\$1.05$. This is not a controlled cost comparison—provider settings, caching, and inference budgets differ—but it shows that the amplified nano system operates in a similar reported cost range rather than a qualitatively different one.

The other panels show two different deployment tradeoffs. On MATH and GSM, a small proposer–judge combination produces large gains while using much less resident model memory than the 32B reference. Repository access improves selection on SWE-bench, but also makes each judgment substantially slower.

Wall-clock cost need not scale with logical call count. Proposals and many judgments are independent and can run in parallel; with open-weight models they can also be batched across candidates or tasks. We did not have the same batching control for the primary `GPT-5.4 nano` experiments, so the reported nano costs do not assume hypothetical batching gains. Full token, call, latency, and cost accounting appears in Appendix D.

8 Discussion and Limitations

One-shot performance can understate what is present in a model's proposal pool: repeated sampling may expose correct solutions that selection must recover. Our theory formalizes this distinction and its multi-step composition, while the experiments primarily measure terminal selection.

The remaining failures point to different remedies. If a correct candidate is present but not selected, better judging can help. If no correct candidate is generated, better judging cannot help; the proposal mechanism itself must change. In the headline SWE-bench experiment, most remaining failures are of this second kind. Our empirical results mostly test terminal selection rather than multi-step composition, finite-k oracle failure only upper-bounds the asymptotic blind-spot floor, and practical deployment must also account for compute, latency, and system complexity.

AI USAGE STATEMENT

In this work, we used generative AI tools for language editing, rewriting, and retrieval and discovery, including finding related work. We also used generative AI tools for research ideation or execution and for assistance in proving mathematical claims. We did not use generative AI tools to draft sections of the paper or to generate synthetic datasets. We reviewed all AI-assisted work, independently checked mathematical arguments and citations, and verified experimental results against the underlying runs and records. We take responsibility for the final content of this work, including text, claims, and artifacts produced with the aid of generative AI.

ETHICS STATEMENT

This work is foundational research on inference-time reasoning and model selection. The experiments use public benchmarks and do not involve human subjects, personal data, or deployment in high-stakes settings. We do not identify any ethics concerns specific to the methods studied beyond the general risks associated with improvements in AI system capabilities.

REPRODUCIBILITY STATEMENT

We provide the theoretical assumptions, theorem statements, and proofs in the main text and Appendix A. The experimental appendix specifies the proposal and selection protocols, prompts, aggregation rules, benchmark sizes, model pairings, oracle definitions, and resource accounting. For fixed-pool experiments, proposal generation is completed before selector evaluation and the same cached candidates are reused across ablations, allowing changes in selection quality to be separated from proposal-generation variation. Exact values underlying the main scaling, identifiability, and resource results are reported in the appendix.

Some experiments use hosted models whose implementations, serving infrastructure, and behavior may change over time. Reported dollar costs also depend on provider pricing and caching policies at the time of evaluation. We therefore report model versions, token usage, call counts, and relevant protocol details in addition to cost. Experiments using open-weight models are specified by model name and evaluation protocol to facilitate independent replication.

REFERENCES

Cem Anil, Guodong Zhang, Yuhuai Wu, and Roger Grosse. Learning to give checkable answers with prover-verifier games, 2021. URL https://arxiv.org/abs/2108.12099.

Leo Breiman. Bagging predictors. *Machine Learning*, 24(2):123–140, 1996. doi: 10.1007/BF00058655. URL https://doi.org/10.1007/BF00058655.

Bradley Brown, Jordan Juravsky, Ryan Ehrlich, Ronald Clark, Quoc V. Le, Christopher Ré, and Azalia Mirhoseini. Large language monkeys: Scaling inference compute with repeated sampling, 2024. URL https://arxiv.org/abs/2407.21787.

Jonah Brown-Cohen, Geoffrey Irving, and Georgios Piliouras. Scalable AI safety via doubly-efficient debate. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pp. 4585–4602. PMLR, 2024. URL https://proceedings.mlr.press/v235/brown-cohen24a.html.

Collin Burns, Pavel Izmailov, Jan Hendrik Kirchner, Bowen Baker, Leo Gao, Leopold Aschenbrenner, Yining Chen, Adrien Ecoffet, Manas Joglekar, Jan Leike, Ilya Sutskever, and Jeffrey Wu. Weak-to-strong generalization: Eliciting strong capabilities with weak supervision. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pp. 4971–5012. PMLR, 2024. URL https://proceedings.mlr.press/v235/burns24b.html.

Lingjiao Chen, Jared Quincy Davis, Boris Hanin, Peter Bailis, Ion Stoica, Matei A. Zaharia, and James Y. Zou. Are more LLM calls all you need? towards the scaling

properties of compound AI systems. In *Advances in Neural Information Processing Systems*, volume 37, pp. 45767–45790, 2024. doi: 10.52202/079017-1455. URL https://proceedings.neurips.cc/paper_files/paper/2024/hash/51173cf34c5faac9796a47dc2fdd3a71-Abstract-Conference.html.

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. URL https://arxiv.org/abs/2107.03374.

Xinyun Chen, Renat Aksitov, Uri Alon, Jie Ren, Kefan Xiao, Pengcheng Yin, Sushant Prakash, Charles Sutton, Xuezhi Wang, and Denny Zhou. Universal self-consistency for large language model generation, 2023. URL https://arxiv.org/abs/2311.17311.

Zehao Chen, Tianxiang Ai, Yifei Li, Gongxun Li, Yuyang Wei, Wang Zhou, Guanghui Li, Bin Yu, Zhijun Chen, Hailong Sun, Fuzhen Zhuang, Jianxin Li, Deqing Wang, and Yikun Ban. LLM-Boost: Make large language models stronger with boosting, 2025. URL https://arxiv.org/abs/2512.22309. Preprint; also submitted to ICLR 2026 on OpenReview.

Paul Christiano, Buck Shlegeris, and Dario Amodei. Supervising strong learners by amplifying weak experts, 2018. URL https://arxiv.org/abs/1810.08575.

Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, Christopher Hesse, and John Schulman. Training verifiers to solve math word problems, 2021. URL https://arxiv.org/abs/2110.14168.

Shehzaad Dhuliawala, Mojtaba Komeili, Jing Xu, Roberta Raileanu, Xian Li, Asli Celikyilmaz, and Jason Weston. Chain-of-verification reduces hallucination in large language models. In *Findings of the Association for Computational Linguistics: ACL 2024*, pp. 3563–3578, Bangkok, Thailand, August 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.findings-acl.212. URL https://aclanthology.org/2024.findings-acl.212/.

Thomas G. Dietterich. Ensemble methods in machine learning. In *Multiple Classifier Systems*, volume 1857 of *Lecture Notes in Computer Science*, pp. 1–15. Springer, 2000. doi: 10.1007/3-540-45014-9_1. URL https://doi.org/10.1007/3-540-45014-9_1.

Yilun Du, Shuang Li, Antonio Torralba, Joshua B. Tenenbaum, and Igor Mordatch. Improving factuality and reasoning in language models through multiagent debate. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pp. 11733–11763. PMLR, 2024. URL https://proceedings.mlr.press/v235/du24e.html.

Yoav Freund and Robert E. Schapire. A decision-theoretic generalization of on-line learning and an application to boosting. In *Computational Learning Theory: Second European Conference, EuroCOLT 1995*, volume 904 of *Lecture Notes in Computer Science*, pp. 23–37. Springer, 1995. doi: 10.1007/3-540-59119-2_166. URL https://doi.org/10.1007/3-540-59119-2_166.

Yoav Freund and Robert E. Schapire. A decision-theoretic generalization of on-line learning and an application to boosting. *Journal of Computer and System Sciences*, 55(1):119–139, 1997. doi: 10.1006/jcss.1997.1504. URL https://doi.org/10.1006/jcss.1997.1504.

Zhibin Gou, Zhihong Shao, Yeyun Gong, Yelong Shen, Yujiu Yang, Nan Duan, and Weizhu Chen. CRITIC: Large language models can self-correct with tool-interactive critiquing. In *International*

Conference on Learning Representations, 2024. URL https://openreview.net/forum?id=Sx038qxjek.

Jiawei Gu, Xuhui Jiang, Zhichao Shi, Hexiang Tan, Xuehao Zhai, Chengjin Xu, Wei Li, Yinghan Shen, Shengjie Ma, Honghao Liu, Saizhuo Wang, Kun Zhang, Zhouchi Lin, Bowen Zhang, Lionel Ni, Wen Gao, Yuanzhuo Wang, and Jian Guo. A survey on LLM-as-a-judge. *The Innovation*, 7 (6):101253, 2026. doi: 10.1016/j.xinn.2025.101253. URL https://doi.org/10.1016/j.xinn.2025.101253.

Neel Guha, Mayee F. Chen, Trevor Chow, Ishan S. Khare, and Christopher Ré. Smoothie: Label free language model routing. In *Advances in Neural Information Processing Systems*, volume 37, 2024. doi: 10.52202/079017-4053. URL https://proceedings.neurips.cc/paper_files/paper/2024/hash/e6b57a990462df5afa58d64ce2709db9-Abstract-Conference.html.

Shibo Hao, Yi Gu, Haodi Ma, Joshua Jiahua Hong, Zhen Wang, Daisy Zhe Wang, and Zhiting Hu. Reasoning with language model is planning with world model. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pp. 8154–8173, Singapore, 2023. Association for Computational Linguistics. doi: 10.18653/v1/2023.emnlp-main.507. URL https://aclanthology.org/2023.emnlp-main.507/.

Sirui Hong, Mingchen Zhuge, Jonathan Chen, Xiawu Zheng, Yuheng Cheng, Jinlin Wang, Ceyao Zhang, Zili Wang, Steven Ka Shing Yau, Zijuan Lin, Liyang Zhou, Chenyu Ran, Lingfeng Xiao, Chenglin Wu, and Jürgen Schmidhuber. MetaGPT: Meta programming for a multi-agent collaborative framework. In *International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=VtmBAGCN7o.

Audrey Huang, Adam Block, Qinghua Liu, Nan Jiang, Akshay Krishnamurthy, and Dylan J. Foster. Is best-of-N the best of them? coverage, scaling, and optimality in inference-time alignment. In *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pp. 25075–25126. PMLR, 2025. URL https://proceedings.mlr.press/v267/huang25c.html.

Geoffrey Irving, Paul Christiano, and Dario Amodei. AI safety via debate, 2018. URL https://arxiv.org/abs/1805.00899.

Dongfu Jiang, Xiang Ren, and Bill Yuchen Lin. LLM-blender: Ensembling large language models with pairwise ranking and generative fusion. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 14165–14178, Toronto, Canada, July 2023. Association for Computational Linguistics. doi: 10.18653/v1/2023.acl-long.792. URL https://aclanthology.org/2023.acl-long.792/.

Carlos E. Jimenez, John Yang, Alexander Wettig, Shunyu Yao, Kexin Pei, Ofir Press, and Karthik R. Narasimhan. SWE-bench: Can language models resolve real-world GitHub issues? In *International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=VTF8yNQM66.

Zachary Kenton, Noah Y. Siegel, János Kramár, Jonah Brown-Cohen, Samuel Albanie, Jannis Bulian, Rishabh Agarwal, David Lindner, Yunhao Tang, Noah D. Goodman, and Rohin Shah. On scalable oversight with weak LLMs judging strong LLMs. In *Advances in Neural Information Processing Systems*, volume 37, 2024. doi: 10.52202/079017-2395. URL https://proceedings.neurips.cc/paper_files/paper/2024/hash/899511e37a8e01e1bd6f6f1d377cc250-Abstract-Conference.html.

Jan Hendrik Kirchner, Yining Chen, Harri Edwards, Jan Leike, Nat McAleese, and Yuri Burda. Prover-verifier games improve legibility of LLM outputs. In *International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=j4s6V1dl8m.

Nathan Lambert, Valentina Pyatkin, Jacob Morrison, LJ Miranda, Bill Yuchen Lin, Khyathi Chandu, Nouha Dziri, Sachin Kumar, Tom Zick, Yejin Choi, Noah A. Smith, and Hannaneh Hajishirzi. RewardBench: Evaluating reward models for language modeling. In *Findings of the Association for Computational Linguistics: NAACL 2025*, pp. 1755–1797, Albuquerque, New Mexico, April

2025. Association for Computational Linguistics. doi: 10.18653/v1/2025.findings-naacl.96. URL https://aclanthology.org/2025.findings-naacl.96/.

Joonhyuk Lee, Virginia Ma, Sarah Zhao, Yash Nair, Asher Spector, Regev Cohen, and Emmanuel J. Candès. FUSE: Ensembling verifiers with zero labeled data. In *International Conference on Machine Learning*, 2026. URL https://arxiv.org/abs/2604.18547.

Guohao Li, Hasan Abed Al Kader Hammoud, Hani Itani, Dmitrii Khizbullin, and Bernard Ghanem. CAMEL: Communicative agents for "mind" exploration of large language model society. In *Advances in Neural Information Processing Systems*, volume 36, 2023. doi: 10.52202/075280-2264. URL https://papers.neurips.cc/paper_files/paper/2023/hash/a3621ee907def47c1b952ade25c67698-Abstract-Conference.html.

Junyou Li, Qin Zhang, Yangbin Yu, Qiang Fu, and Deheng Ye. More agents is all you need. *Transactions on Machine Learning Research*, 2024. URL https://openreview.net/forum?id=bgzUSZ8aeg.

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 Mankowitz, Esme Sutherland Robson, Pushmeet Kohli, Nando de Freitas, Koray Kavukcuoglu, and Oriol Vinyals. Competition-level code generation with AlphaCode. *Science*, 378(6624):1092–1097, 2022. doi: 10.1126/science.abq1158. URL https://doi.org/10.1126/science.abq1158.

Tian Liang, Zhiwei He, Wenxiang Jiao, Xing Wang, Yan Wang, Rui Wang, Yujiu Yang, Shuming Shi, and Zhaopeng Tu. Encouraging divergent thinking in large language models through multi-agent debate. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pp. 17889–17904, Miami, Florida, USA, November 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.emnlp-main.992. URL https://aclanthology.org/2024.emnlp-main.992/.

Shalev Lifshitz, Sheila A. McIlraith, and Yilun Du. Multi-agent verification: Scaling test-time compute with multiple verifiers. In *Second Conference on Language Modeling*, 2025. URL https://openreview.net/forum?id=LriQ3NY9uL.

Hunter Lightman, Vineet Kosaraju, Yuri Burda, Harrison Edwards, Bowen Baker, Teddy Lee, Jan Leike, John Schulman, Ilya Sutskever, and Karl Cobbe. Let's verify step by step. In *International Conference on Learning Representations*, 2024. URL https://proceedings.iclr.cc/paper_files/paper/2024/hash/aca97732e30bcf1303bc22ac3924fd16-Abstract-Conference.html.

Aman Madaan, Niket Tandon, Prakhar Gupta, Skyler Hallinan, Luyu Gao, Sarah Wiegreffe, Uri Alon, Nouha Dziri, Shrimai Prabhumoye, Yiming Yang, Shashank Gupta, Bodhisattwa Prasad Majumder, Katherine Hermann, Sean Welleck, Amir Yazdanbakhsh, and Peter Clark. Self-refine: Iterative refinement with self-feedback. In *Advances in Neural Information Processing Systems*, volume 36, 2023. URL https://openreview.net/forum?id=S37hOerQLB.

Hariharan Manikandan, Yiding Jiang, and J. Zico Kolter. Language models are weak learners. In *Advances in Neural Information Processing Systems*, volume 36, 2023. URL https://proceedings.neurips.cc/paper_files/paper/2023/hash/9f94298bac4668db4dc77ddb0a244301-Abstract-Conference.html.

Jon Saad-Falcon, Estefany Kelly Buchanan, Mayee Chen, Tzu-Heng Huang, Brendan McLaughlin, Tanvir Bhathal, Shang Zhu, Ben Athiwaratkun, Frederic Sala, Scott Linderman, Azalia Mirhoseini, and Christopher Ré. Weaver: Shrinking the generation-verification gap by scaling compute for verification. In *Advances in Neural Information Processing Systems*, volume 38, 2025a. doi: 10.52202/085713-2639. URL https://papers.neurips.cc/paper_files/paper/2025/hash/715f48023c26ad210321a4c0da67f27c-Abstract-Conference.html.

Jon Saad-Falcon, Adrian Gamarra Lafuente, Shlok Natarajan, Nahum Maru, Hristo Todorov, Etash Kumar Guha, E. Kelly Buchanan, Mayee F. Chen, Neel Guha, Christopher Ré, and Azalia Mirhoseini. An architecture search framework for inference-time techniques. In *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pp. 52475–52507. PMLR, 2025b. URL https://proceedings.mlr.press/v267/saad-falcon25a.html.

Robert E. Schapire. The strength of weak learnability. *Machine Learning*, 5(2):197–227, 1990. doi: 10.1007/BF00116037. URL https://doi.org/10.1007/BF00116037.

Timo Schick, Jane Dwivedi-Yu, Roberto Dessì, Roberta Raileanu, Maria Lomeli, Eric Hambro, Luke Zettlemoyer, Nicola Cancedda, and Thomas Scialom. Toolformer: Language models can teach themselves to use tools. In *Advances in Neural Information Processing Systems*, volume 36, 2023. URL https://proceedings.neurips.cc/paper_files/paper/2023/hash/d842425e4bf79ba039352da0f658a906-Abstract-Conference.html.

Yongliang Shen, Kaitao Song, Xu Tan, Dongsheng Li, Weiming Lu, and Yueting Zhuang. HuggingGPT: Solving AI tasks with ChatGPT and its friends in Hugging Face. In *Advances in Neural Information Processing Systems*, volume 36, 2023. URL https://proceedings.neurips.cc/paper_files/paper/2023/hash/77c33e6a367922d003ff102ffb92b658-Abstract-Conference.html.

Noah Shinn, Federico Cassano, Ashwin Gopinath, Karthik Narasimhan, and Shunyu Yao. Reflexion: Language agents with verbal reinforcement learning. In *Advances in Neural Information Processing Systems*, volume 36, 2023. URL https://papers.neurips.cc/paper_files/paper/2023/hash/1b44b878bb782e6954cd888628510e90-Abstract-Conference.html.

Junlin Wang, Jue Wang, Ben Athiwaratkun, Ce Zhang, and James Zou. Mixture-of-agents enhances large language model capabilities. In *International Conference on Learning Representations*, 2025. URL https://proceedings.iclr.cc/paper_files/paper/2025/hash/5434be94e82c54327bb9dcaf7fca52b6-Abstract-Conference.html.

Peiyi Wang, Lei Li, Zhihong Shao, Runxin Xu, Damai Dai, Yifei Li, Deli Chen, Yu Wu, and Zhifang Sui. Math-shepherd: Verify and reinforce LLMs step-by-step without human annotations. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 9426–9439, Bangkok, Thailand, August 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.acl-long.510. URL https://aclanthology.org/2024.acl-long.510/.

Xuezhi Wang, Jason Wei, Dale Schuurmans, Quoc V. Le, Ed H. Chi, and Denny Zhou. Rationale-augmented ensembles in language models, 2022. URL https://arxiv.org/abs/2207.00747.

Xuezhi Wang, Jason Wei, Dale Schuurmans, Quoc V. Le, Ed H. Chi, Sharan Narang, Aakanksha Chowdhery, and Denny Zhou. Self-consistency improves chain of thought reasoning in language models. In *International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=1PL1NIMMrw.

David H. Wolpert. Stacked generalization. *Neural Networks*, 5(2):241–259, 1992. doi: 10.1016/S0893-6080(05)80023-1. URL https://doi.org/10.1016/S0893-6080(05)80023-1.

Qingyun Wu, Gagan Bansal, Jieyu Zhang, Yiran Wu, Beibin Li, Erkang Zhu, Li Jiang, Xiaoyun Zhang, Shaokun Zhang, Jiale Liu, Ahmed Hassan Awadallah, Ryen W. White, Doug Burger, and Chi Wang. AutoGen: Enabling next-gen LLM applications via multi-agent conversations. In *First Conference on Language Modeling*, 2024. URL https://openreview.net/forum?id=BAakY1hNKS.

Chunqiu Steven Xia, Yinlin Deng, Soren Dunn, and Lingming Zhang. Demystifying LLM-based software engineering agents. *Proceedings of the ACM on Software Engineering*, 2(FSE):801–824, 2025. doi: 10.1145/3715754. URL https://doi.org/10.1145/3715754.

John Yang, Carlos E. Jimenez, Alexander Wettig, Kilian Lieret, Shunyu Yao, Karthik Narasimhan, and Ofir Press. SWE-agent: Agent-computer interfaces enable automated software engineering. In *Advances in Neural Information Processing Systems*, volume 37, 2024. doi: 10.52202/079017-1601. URL https://proceedings.neurips.cc/paper_files/paper/2024/hash/5a7c947568c1b1328ccc5230172e1e7c-Abstract-Conference.html.

Shunyu Yao, Dian Yu, Jeffrey Zhao, Izhak Shafran, Thomas L. Griffiths, Yuan Cao, and Karthik Narasimhan. Tree of thoughts: Deliberate problem solving with large language models. In *Advances in Neural Information Processing Systems*, volume 36, 2023a. URL https://openreview.net/forum?id=5Xc1ecxO1h.

Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik Narasimhan, and Yuan Cao. ReAct: Synergizing reasoning and acting in language models. In *International Conference on Learning Representations*, 2023b. URL https://openreview.net/forum?id=WE_vluYUL-X.

Qiyuan Zhang, Fuyuan Lyu, Zexu Sun, Lei Wang, Weixu Zhang, Wenyue Hua, Haolun Wu, Zhihan Guo, Yufei Wang, Niklas Muennighoff, Irwin King, Xue Liu, and Chen Ma. A survey on test-time scaling in large language models: What, how, where, and how well?, 2025. URL https://arxiv.org/abs/2503.24235.

Yuntong Zhang, Haifeng Ruan, Zhiyu Fan, and Abhik Roychoudhury. AutoCodeRover: Autonomous program improvement. In *Proceedings of the 33rd ACM SIGSOFT International Symposium on Software Testing and Analysis*, pp. 1592–1604. ACM, 2024. doi: 10.1145/3650212.3680384. URL https://doi.org/10.1145/3650212.3680384.

Lianmin Zheng, Wei-Lin Chiang, Ying Sheng, Siyuan Zhuang, Zhanghao Wu, Yonghao Zhuang, Zi Lin, Zhuohan Li, Dacheng Li, Eric P. Xing, Hao Zhang, Joseph E. Gonzalez, and Ion Stoica. Judging LLM-as-a-judge with MT-Bench and chatbot arena. In *Advances in Neural Information Processing Systems*, volume 36, 2023. URL https://openreview.net/forum?id=uccHPGDlao.

Andy Zhou, Kai Yan, Michal Shlapentokh-Rothman, Haohan Wang, and Yu-Xiong Wang. Language agent tree search unifies reasoning, acting, and planning in language models. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pp. 62138–62160. PMLR, 2024. URL https://proceedings.mlr.press/v235/zhou24r.html.

A Proofs and Theory Extensions

This section proves the results from Sections 4–6 and develops several extensions that clarify the scope of the theory. Throughout, probabilities are taken over the fresh randomness of the corresponding proposer, critic, or comparator calls, conditional on any state, candidate pool, or history specified in the statement.

A.1 Coverage and Identifiability

For Proposition 1, the identification procedure may know the task-family construction, the action set, and the proposer distribution. It may also observe the proposed actions and arbitrarily many additional samples from the same proposer. What it does not observe is the hidden world parameter θ. The critic and comparator requirements must hold uniformly over all values of θ.

A.1.1 Proof of Proposition 1

Proposition 1 (Coverage does not imply local identifiability)**.** *For every $M \geq 2$, there is a one-step task family with action set $\mathcal{A} = \{1, \ldots, M\}$ and hidden parameter $\theta \in \{1, \ldots, M\}$ such that the proposer is $\mathrm{Unif}(\mathcal{A})$ in every world and $\mathrm{Sound}_\theta = \mathcal{A} \setminus \{\theta\}$. Even with arbitrarily many samples from this proposer, a selector that receives no information about θ cannot have a uniform positive critic or comparator edge over all worlds.*

Proof. Fix $M \geq 2$. In world $\theta \in \{1, \ldots, M\}$, let $\mathrm{Sound}_\theta = \mathcal{A} \setminus \{\theta\}$, and let the proposer distribution be $\mu = \mathrm{Unif}(\mathcal{A})$, independently of θ. Thus a proposal is sound with probability $(M - 1)/M$ in every world, but the entire observable proposal transcript has exactly the same distribution for every value of θ.

We first rule out a one-sided critic with positive uniform rejection edge. Fix an action i. When $\theta = i$, the action i is unsound; when $\theta \neq i$, it is sound. Since the procedure does not observe θ, its observable input has the same distribution in all of these worlds, and hence its probability of rejecting i is the same in every world. The one-sided condition requires this rejection probability to be zero whenever $\theta \neq i$. It must therefore also be zero when $\theta = i$. Thus no one-sided critic can reject every unsound action with any uniform positive probability.

Now consider a comparator and fix two distinct actions i and j. In world $\theta = i$, j is sound and i is unsound, so a comparator with edge $\sigma > 0$ must choose j with probability at least $1/2 + \sigma$. In world $\theta = j$, the roles reverse, and the same procedure must choose i with probability at least $1/2 + \sigma$. But the observable transcript, including any additional proposer samples, has the same distribution in the two worlds. These are therefore requirements on one and the same output distribution. Since the probabilities of choosing i and j sum to at most one, they cannot both exceed $1/2$. Hence no uniform positive comparator edge exists. □

The construction deliberately separates *coverage* from *identifiability*: useful actions are abundant, but the proposer samples contain no information about which action is the exceptional bad one.

A.1.2 Proof of Theorem 1

Theorem 1 (Coverage amplification from a proposer portfolio)**.** *Fix a reachable valid nonterminal state s and suppose Assumption 1 holds. Query the proposers in P_N in round-robin order with fresh independent randomness. After k calls, the probability that the candidate pool contains a progressing-sound action is at least $1 - (1 - \alpha_0)^{\lfloor k/|P_N| \rfloor}$. In particular, this probability is at least $1 - \delta_{\mathrm{prop}}$ whenever $k \geq |P_N| \lceil \alpha_0^{-1} \log(1/\delta_{\mathrm{prop}}) \rceil$.*

Proof. Fix a reachable valid nonterminal state s, and let $p_s \in P_N$ be the portfolio member guaranteed by Assumption 1. Under round-robin assignment, every portfolio member is queried at least $q := \lfloor k/|P_N| \rfloor$ times. In particular, p_s is queried at least q times.

Each such call produces a progressing-sound action with probability at least α_0. Because these calls use fresh independent randomness, the probability that all q calls to p_s fail is at most $(1 - \alpha_0)^q$. The

event that the entire k-proposal pool contains no progressing-sound action is a subset of this event, so

$$\mathbb{P}(\text{no progressing-sound proposal}) \leq (1-\alpha_0)^{\lfloor k/|P_N| \rfloor}.$$

This proves the first claim.

For the second claim, suppose $k \geq |P_N| \lceil \alpha_0^{-1} \log(1/\delta_{\text{prop}}) \rceil$. Then $\lfloor k/|P_N| \rfloor \geq \alpha_0^{-1} \log(1/\delta_{\text{prop}})$. Using $1-u \leq e^{-u}$, the miss probability is at most $e^{-\alpha_0 q} \leq \delta_{\text{prop}}$, as required. □

A.1.3 Proof of Corollary 1

Corollary 1 (Local verification implies identifiability)**.** *Under Assumption 3, Assumption 2 holds with $\beta_0 = 1-\nu$ and $\sigma_0 = (1-\nu)/2$.*

Proof. Use $V_x(s,\cdot)$ directly as the critic. A progressing-sound action is accepted almost surely, while an unsound action is rejected with probability at least $1-\nu$. Thus the critic satisfies Assumption 2 with $\beta_0 = 1-\nu$.

For the comparator, apply the verifier to both candidates. If exactly one is rejected, choose the other; otherwise choose uniformly between them. Consider a pair consisting of a progressing-sound action a and an unsound action b. The verifier never rejects a. Let p be the probability that it rejects b; by assumption $p \geq 1-\nu$. The induced comparator therefore selects a with probability $p + (1-p)/2 = 1/2 + p/2$, which is at least $1/2 + (1-\nu)/2$. Hence $\sigma_0 = (1-\nu)/2$ is a valid comparator edge. □

Notice that independence between the two verifier calls is not needed for this argument: the sound candidate is accepted almost surely, so only the rejection probability of the unsound candidate enters the bound.

A.1.4 Final Verification Does Not Imply Local Identification

The existence of an efficient verifier for completed outputs is substantially weaker than Assumption 2. The following construction makes the distinction explicit.

Proposition 2 (A final verifier need not yield an efficient local critic)**.** *There is a polynomial-time verifiable search relation with a natural partial-state decomposition for which deciding whether a proposed action is progressing-sound is NP-complete, even at a reachable valid state. Consequently, if there were a* uniform *polynomial-time one-sided critic for this decomposition whose rejection edge satisfied $\beta_0 \geq 1/\text{poly}(N)$, then* $\text{coNP} \subseteq \text{RP}$*, equivalently* $\text{NP} \subseteq \text{coRP}$.

Proof. Consider circuit satisfiability. An instance is a Boolean circuit C, a complete output is an assignment y, and the final verifier accepts exactly when $C(y) = 1$. Complete outputs are therefore verifiable in polynomial time.

Use partial assignments as states. Declare a partial assignment valid exactly when it admits at least one satisfying completion. An action assigns a value to one previously unassigned variable, and let the rank be the number of unassigned variables. Since every action decreases the rank by one, such an action is progressing-sound exactly when the extended partial assignment remains valid, i.e., has a satisfying completion.

We now reduce circuit satisfiability to this local decision problem. Given an arbitrary circuit $\psi(u_1, \ldots, u_m)$, construct $C(x_0, u) = x_0 \vee \psi(u)$. The empty partial assignment is valid, since setting $x_0 = 1$ satisfies C. Consider at this reachable valid state the action $x_0 \leftarrow 0$. After this action, a satisfying completion exists if and only if ψ is satisfiable. Hence this action is progressing-sound if and only if ψ is satisfiable. Local progressing-soundness is therefore NP-hard. It is also in NP, because a satisfying completion is a polynomial-size certificate. Thus local progressing-soundness is NP-complete, and local unsoundness is coNP-complete.

Now suppose there is a uniform polynomial-time one-sided critic whose rejection probability on every unsound action is at least $\beta_0 \geq 1/\text{poly}(N)$. Given a state–action pair, run the critic independently $T := \lceil \log 3/\beta_0 \rceil$ times and accept the statement "unsound" if any run rejects. If the action is

sound, one-sidedness implies that this procedure never accepts. If the action is unsound, the probability of seeing no rejection is at most $(1-\beta_0)^T \le e^{-\beta_0 T} \le 1/3$. Thus local unsoundness has an RP algorithm. Since local unsoundness is coNP-complete, this implies $\mathrm{coNP} \subseteq \mathrm{RP}$. □

The uniformity qualifier is essential here: the reduction requires a single algorithm that can be run on an arbitrary encoded state–action pair, rather than an unrelated nonuniform procedure supplied separately for each state.

A.2 Local-to-Global Amplification

A.2.1 Proof of Theorem 2

Theorem 2 (Local error decomposition)**.** *Suppose the local-identifiability conditions hold at s with critic edge β and comparator edge σ, and let r be odd. Conditional on the current state and the proposed candidates, assume that repeated critic calls are independent, that repeated comparator votes for each pair are independent, and that the comparator randomness is fresh relative to the critic randomness. Then*

$$\varepsilon_{\mathrm{loc}}(s) \le \varepsilon_{\mathrm{prop}}(k;s) + k^2(1-\beta)^m e^{-2r\sigma^2} \le \varepsilon_{\mathrm{prop}}(k;s) + k^2 e^{-\beta m - 2r\sigma^2}.$$

If, in addition, the k proposal calls are conditionally independent and each is progressing-sound with probability at least α, then $\varepsilon_{\mathrm{prop}}(k;s) \le (1-\alpha)^k \le e^{-\alpha k}$.

Proof. Fix a reachable valid nonterminal state s. Let E_{prop} denote the event that none of the k proposed candidates lies in $\mathrm{Sound}_x(s)$. By definition, $\mathbb{P}(E_{\mathrm{prop}}) = \varepsilon_{\mathrm{prop}}(k;s)$.

Condition now on E^c_{prop}, so at least one progressing-sound candidate is present. Because the critic is one-sided, every progressing-sound candidate survives the critic stage. Thus, conditional on proposal coverage, the only way the local step can select an unsound action is through an unsound survivor that is not defeated by the sound survivors.

Fix one progressing-sound proposed candidate a and one unsound proposed candidate b. Each critic call rejects b with probability at least β. Conditional on the proposed pool, the m critic calls are independent, so

$$\mathbb{P}(b \text{ survives all critics}) \le (1-\beta)^m.$$

Suppose b survives. For comparator repetition j, let $Y_j = 1$ if the vote prefers a to b and $Y_j = 0$ otherwise. The comparator edge gives $\mathbb{E}[Y_j] \ge 1/2 + \sigma$. The votes are independent across repetitions, and their randomness is fresh relative to the critic stage. Since r is odd, a fails to win the pairwise majority only if $r^{-1}\sum_{j=1}^{r} Y_j \le 1/2$. Hoeffding's inequality therefore gives

$$\mathbb{P}(a \text{ fails to beat } b) \le e^{-2r\sigma^2}.$$

Freshness of the comparator randomness implies that this bound remains valid conditional on b's survival. Hence

$$\mathbb{P}(b \text{ survives and } a \text{ does not beat } b) \le (1-\beta)^m e^{-2r\sigma^2}.$$

There are at most k^2 ordered progressing-sound–unsound candidate pairs. A union bound therefore shows that, except with probability at most $k^2(1-\beta)^m e^{-2r\sigma^2}$, every surviving sound candidate defeats every surviving unsound candidate.

It remains to verify that this event forces a Copeland winner to be sound. Let $S \ge 1$ and $U \ge 0$ denote the numbers of surviving sound and unsound candidates. Every sound survivor defeats all U unsound survivors, so its Copeland score is at least U. Every unsound survivor loses to all S sound survivors and can defeat at most the other $U-1$ unsound survivors, so its score is at most $U-1$. Hence every Copeland maximizer is sound.

Combining this selection-failure event with E_{prop} yields

$$\varepsilon_{\mathrm{loc}}(s) \le \varepsilon_{\mathrm{prop}}(k;s) + k^2(1-\beta)^m e^{-2r\sigma^2}.$$

Using $1-\beta \le e^{-\beta}$ gives the second displayed bound in the theorem.

Finally, if the k proposer calls are conditionally independent and each lands in $\mathrm{Sound}_x(s)$ with probability at least α, then all k miss with probability at most $(1-\alpha)^k \le e^{-\alpha k}$. □

Two aspects of the proof are worth noting. Independence is needed across repeated critic calls for a given candidate and across repeated comparator votes for a given pair, but not across different candidates or different pairs. Likewise, the Copeland argument does not require comparisons among sound candidates, or among unsound candidates, to be accurate.

A.2.2 Proof of Lemma 1

Lemma 1 (Adaptive cumulative-error bound)**.** *Let $\mathcal{F}_t$ denote the history available before the decision at step t. Suppose that, whenever the protocol is at a reachable valid nonterminal state, the conditional probability of local failure at step t is at most ε_t. Then* $\mathrm{err}_x(k, m, r) \le \mathbb{E}[\sum_{t=0}^{L_x-1} \varepsilon_t]$. *In particular, if $\varepsilon_t \le \varepsilon$ uniformly along the trajectory, then* $\mathrm{err}_x(k, m, r) \le L_x\varepsilon$.

Proof. For $t = 0, \dots, L_x - 1$, let B_t be the event that step t is reached at a valid nonterminal state and the local step fails, either by returning failure or by selecting an action outside $\mathrm{Sound}_x(S_t)$. If step t is not reached at such a state, define B_t to be false.

Suppose none of $B_0, \dots, B_{L_x-1}$ occurs. Every action that is actually applied is then progressing-sound. Consequently, validity is preserved and the nonnegative integer rank decreases strictly at every applied step. Starting from rank at most L_x, the process must therefore enter a rank-zero state within at most L_x steps. By Definition 1, every valid rank-zero state is terminal, and every terminal output is accepted by R_x. Hence global failure implies $\bigcup_{t=0}^{L_x-1} B_t$.

By the union bound, $\mathrm{err}_x(k, m, r) \le \sum_{t<L_x} \mathbb{P}(B_t)$. Moreover, by the assumed conditional bound and the tower property, $\mathbb{P}(B_t) = \mathbb{E}[\mathbb{P}(B_t \mid \mathcal{F}_t)] \le \mathbb{E}[\varepsilon_t]$. Therefore

$$\mathrm{err}_x(k, m, r) \le \mathbb{E}\left[\sum_{t=0}^{L_x-1} \varepsilon_t\right].$$

If $\varepsilon_t \le \varepsilon$ uniformly, the right-hand side is at most $L_x\varepsilon$. □

Crucially, no independence across reasoning steps is assumed. The state at step t, the candidate distribution there, and the local error probability may all depend arbitrarily on the preceding trajectory; the lemma uses only a conditional error bound at the state actually reached.

A.2.3 Proof of Corollary 2

Proof. At every reachable valid nonterminal state s, Theorem 2 and the uniform assumptions give

$$\varepsilon_{\mathrm{loc}}(s) \le \varepsilon_{\mathrm{prop}}(k) + k^2 e^{-\beta m - 2r\sigma^2}.$$

This is a uniform conditional bound on the failure probability of every local decision encountered along the adaptive trajectory. Applying Lemma 1 therefore gives

$$\mathrm{err}_x(k, m, r) \le L_x\left(\varepsilon_{\mathrm{prop}}(k) + k^2 e^{-\beta m - 2r\sigma^2}\right).$$

If the quantity in parentheses is at most δ/L_x, the right-hand side is at most δ, proving the claimed success probability. □

A.2.4 Allowing Critic False Rejections

The one-sided critic assumption isolates the main mechanism cleanly, but the argument extends to critics that occasionally reject good candidates.

Proposition 3 (Two-sided critic extension)**.** *Suppose that, conditional on the state and candidate, each critic call rejects a progressing-sound candidate with probability at most γ and rejects an unsound candidate with probability at least β. Under the comparator and independence conditions of Theorem 2,*

$$\varepsilon_{\mathrm{loc}}(s) \le \varepsilon_{\mathrm{prop}}(k; s) + m\gamma + k^2(1-\beta)^m e^{-2r\sigma^2}.$$

Proof. Condition on the event that the proposal pool contains at least one progressing-sound candidate. Given the realized proposal pool, choose one such candidate $a_\star$ by a deterministic rule, for example the lowest-index sound candidate. We do not need every sound candidate to survive; it is enough to ensure that this single witness survives.

Each of the m critic calls rejects $a_\star$ with probability at most γ. No independence is needed here: by the union bound, the probability that at least one call rejects $a_\star$ is at most $m\gamma$. Outside this event, at least one sound candidate survives.

Now fix any sound–unsound proposed pair (a, b). For this pair to threaten the Copeland selection, both candidates must survive and a must fail to beat b. We may upper-bound the sound candidate's survival probability by one. The unsound candidate survives all m independent critic calls with probability at most $(1-\beta)^m$, and conditional on the critic stage the fresh comparator votes fail with probability at most $e^{-2r\sigma^2}$. Thus

$$\mathbb{P}(a, b \text{ survive and } a \text{ fails to beat } b) \le (1-\beta)^m e^{-2r\sigma^2}.$$

A union bound over fewer than k^2 sound–unsound pairs shows that, except with probability given by the last term in the proposition, every surviving sound candidate defeats every surviving unsound candidate. Provided $a_\star$ was not rejected, at least one sound candidate survives, and the same Copeland-score argument as in Theorem 2 implies that every winner is sound. Adding proposal failure and the possible rejection of $a_\star$ proves the result. □

The $m\gamma$ term reveals the basic critic-repetition tradeoff: increasing m suppresses unsound survivors exponentially, but, when false rejections are possible, it also increases the chance of discarding a useful candidate.

A.3 Blind Spots, Diversity, and Finite-Width Rates

A.3.1 Proof of Lemma 2

Lemma 2 (Oracle miss and blind-spot floor)**.** *Fix a reachable valid nonterminal state s. Let Z capture latent heterogeneity shared across proposal calls. Conditional on Z, suppose the k calls are independent and each proposes a progressing-sound action with probability $q_s(Z)$. Then*

$$\varepsilon_{\text{prop}}(k; s) = \mathbb{E}[(1 - q_s(Z))^k] = B_s + R_k(s),$$

where $B_s := \mathbb{P}(q_s(Z) = 0)$ and $R_k(s) := \mathbb{E}[(1 - q_s(Z))^k \mathbf{1}\{q_s(Z) > 0\}]$. Moreover, $R_k(s) \to 0$, so $\varepsilon_{\text{prop}}(k; s) \to B_s$.

Proof. Conditional on Z, each proposal is progressing-sound with probability $q_s(Z)$, and the k proposal calls are independent. Hence all k proposals miss with probability $(1-q_s(Z))^k$. Averaging over Z gives $\varepsilon_{\text{prop}}(k; s) = \mathbb{E}[(1 - q_s(Z))^k]$.

Split this expectation according to whether $q_s(Z) = 0$ or $q_s(Z) > 0$. On the first event, the integrand equals one, contributing $B_s = \mathbb{P}(q_s(Z) = 0)$; the second contribution is exactly $R_k(s)$. Thus $\varepsilon_{\text{prop}}(k; s) = B_s + R_k(s)$.

On the event $q_s(Z) > 0$, $(1 - q_s(Z))^k \to 0$ pointwise as $k \to \infty$, while the integrand is bounded by one. Dominated convergence therefore gives $R_k(s) \to 0$, and hence $\varepsilon_{\text{prop}}(k; s) \to B_s$. □

A.3.2 Heterogeneous Proposer Portfolios

Proposition 4 (Heterogeneous portfolio oracle miss)**.** *Fix a finite set of proposer families $g \in \{1, \dots, G\}$, and suppose family g is called n_g times. Conditional on Z, assume all calls are independent and a call from family g is progressing-sound with probability $q_g(Z)$. Then*

$$\varepsilon_{\text{prop}}(\{n_g\}; s) = \mathbb{E}\left[\prod_{g=1}^{G} (1 - q_g(Z))^{n_g}\right] \le \mathbb{E}[e^{-Q(Z)}],$$

where $Q(Z) := \sum_g n_g q_g(Z)$. *If* $Q(Z) \geq \lambda$ *outside an event of probability at most* η, *then* $\varepsilon_{\text{prop}}(\{n_g\}; s) \leq \eta + e^{-\lambda}$. *If every* $n_g \to \infty$, *the miss probability converges to* $\mathbb{P}(q_g(Z) = 0$ *for all* $g)$.

Proof. Conditional on Z, the probability that every call from family g misses is $(1 - q_g(Z))^{n_g}$. Conditional independence across calls therefore gives the product formula. Applying $1 - u \leq e^{-u}$ to every factor yields the pointwise bound

$$\prod_g (1 - q_g(Z))^{n_g} \leq \exp\left(-\sum_g n_g q_g(Z)\right) = e^{-Q(Z)}.$$

Let $A = \{Q(Z) \geq \lambda\}$. On A, the product is at most $e^{-\lambda}$; on A^c, it is at most one. Since $\mathbb{P}(A^c) \leq \eta$, its expectation is at most $e^{-\lambda} + \eta$.

Finally, suppose every $n_g \to \infty$. For a fixed Z, if $q_g(Z) = 0$ for every family, the product equals one for all sample sizes. If at least one family has $q_g(Z) > 0$, the corresponding factor tends to zero, so the entire product tends to zero. The product is bounded by one, and dominated convergence gives the claimed limit. □

Thus proposer diversity affects the asymptotic ceiling only when the families have complementary support. Additional samples from one family reduce finite-width error on cases it can already solve; they cannot repair cases on which that family has zero useful-action mass.

A.3.3 Finite-Width Rates

Proposition 5 (Lower-tail bound)**.** *Suppose that for some* $a, C > 0$, $\mathbb{P}(0 < q_s(Z) \leq t) \leq Ct^a$ *for all sufficiently small* t. *Then, for all sufficiently large* k,

$$R_k(s) \leq C\left(\frac{a \log k}{k}\right)^a + k^{-a}.$$

If instead $q_s(Z) \geq \tau > 0$ *whenever* $q_s(Z) > 0$, *then* $R_k(s) \leq e^{-\tau k}$.

Proof. For the second statement, on the event $q_s(Z) > 0$ we have $q_s(Z) \geq \tau$, and therefore $(1 - q_s(Z))^k \leq e^{-kq_s(Z)} \leq e^{-\tau k}$. Averaging gives the claimed bound.

For the first statement, set $t_k := a \log(k)/k$. For sufficiently large k, t_k lies in the range on which the assumed lower-tail estimate holds. Split $R_k(s)$ according to the events $0 < q_s(Z) \leq t_k$ and $q_s(Z) > t_k$. On the first event, the integrand is at most one, so its contribution is at most Ct_k^a. On the second, $(1 - q_s(Z))^k \leq e^{-kq_s(Z)} \leq e^{-kt_k} = k^{-a}$. The second contribution is therefore at most k^{-a}. Adding the two pieces yields the result. □

The result separates the limiting blind-spot floor from the rate at which the finite-width residual approaches it. Even when $B_s = 0$, substantial mass near $q_s(Z) = 0$ can make convergence with k slow.

A.3.4 Terminal Oracle Curves

For completeness, the task-level oracle formula in Section 6 follows by the same conditioning argument. Given $X = x$, k independent samples from $P(\cdot \mid x)$ all fail with probability $(1 - q_P(x))^k$. Averaging over X gives $p_{\text{oracle}}(k; P) = \mathbb{E}[1 - (1 - q_P(X))^k]$. For every x, the integrand converges to $\mathbf{1}\{q_P(x) > 0\}$, and it is bounded by one. Dominated convergence therefore yields $\lim_{k\to\infty} p_{\text{oracle}}(k; P) = 1 - \mathbb{P}(q_P(X) = 0)$.

This also makes clear why finite-k oracle failure should not be identified with the asymptotic blind-spot mass: at finite width it contains the residual sampling term as well.

B Polynomial-Resource Consequence

Corollary 2 gives an efficient amplification regime whenever coverage, local identifiability, and trajectory length remain polynomially controlled.

Corollary 3 (Polynomial-resource amplification)**.** *Suppose that, for size-N inputs, L_x, $|P_N|$, $1/\alpha_0$, $1/\beta_0$, and $1/\sigma_0$ are polynomially bounded, and that each proposer, critic, comparator, and verifier call runs in polynomial time. Then for every $\delta \in (0,1)$, there exist $k, m, r = \mathrm{poly}(N, \log(1/\delta))$ such that $\mathbb{P}(R_x(\Pi_{k,m,r}(x)) = 0) \le \delta$. The all-pairs implementation uses $O(L_x(k + mk + rk^2))$ role calls.*

Proof. If $L_x = 0$, the initial state is already terminal and the claim is trivial, so assume $L_x \ge 1$.

By Theorem 1, it suffices to choose

$$k \ge |P_N| \left\lceil \alpha_0^{-1} \log \frac{2L_x}{\delta} \right\rceil .$$

Then the proposal-failure probability at every reachable valid nonterminal state is at most $\delta/(2L_x)$.

It remains to make the selection term at most the same amount. It suffices that $\beta_0 m + 2\sigma_0^2 r \ge \log(2k^2 L_x/\delta)$. One convenient allocation is

$$m \ge \left\lceil \frac{1}{2\beta_0} \log \frac{2k^2 L_x}{\delta} \right\rceil , \qquad r \ge \left\lceil \frac{1}{4\sigma_0^2} \log \frac{2k^2 L_x}{\delta} \right\rceil .$$

With these choices, $k^2 e^{-\beta_0 m - 2\sigma_0^2 r} \le \delta/(2L_x)$. Corollary 2 therefore gives total failure probability at most δ.

At each reasoning step, the implementation makes k proposer calls and at most mk critic calls. At most k candidates survive, so there are at most $\binom{k}{2}$ compared pairs and hence at most $r\binom{k}{2}$ comparator calls. There are at most L_x steps, giving $O(L_x(k + mk + rk^2))$ role calls overall.

Finally, the assumed polynomial bounds on L_x, $|P_N|$, $1/\alpha_0$, $1/\beta_0$, and $1/\sigma_0$ imply that the displayed choices of k, m, r are polynomial in N and $\log(1/\delta)$. Polynomial per-call running time then gives polynomial overall running time. □

The assumptions are substantive. Efficient amplification can fail because the useful proposal mass is exponentially small, local identification edges vanish too rapidly, the required trajectory is superpolynomially long, or the local checking procedures are themselves computationally expensive. Efficient verification of completed outputs alone rules out none of these obstructions.

C SWE-bench Implementation Details

The SWE-bench experiments are built around a fixed-pool design. Proposal generation is completed first; the same cached patches are then reused across selector variants. This keeps proposal coverage fixed when we compare critics, comparators, gates, and aggregation rules.

C.1 Benchmark and proposal pools

SWE-bench Verified (Jimenez et al., 2024) contains 500 repository-level software-engineering tasks. Each task provides a repository state and an issue description, and a patch is correct only if it passes the benchmark's held-out tests. Hidden-test outcomes are used for final scoring and oracle diagnostics only; they are never exposed to the selector.

For each task we generate $k = 8$ candidate patches with independent `GPT-5.4 nano` Mini-SWE-agent trajectories. Each trajectory receives the issue, interacts with the repository, and returns a patch. The eight patches are cached before selector ablations begin.

Oracle best-of-k is computed after hidden-test evaluation. A task is oracle solved if at least one of the k cached patches passes the hidden tests. It therefore measures proposal coverage, not a deployable selection rule.

C.2 SELECTION PIPELINE

Binary critic. The binary patch judge receives the issue and one candidate patch and predicts whether the patch resolves the issue without breaking previously working behavior. Each response contains a binary decision, a short rationale, and a confidence score. Multiple independent judgments are cached for each patch. The basic critic score is the number of positive votes; the confidence-weighted variant sums confidence over positive votes. Ties are broken by proposal index.

Pairwise comparator. The comparator receives the issue and two candidate patches. It states a failure hypothesis, summarizes the relevant changes in each patch, checks whether those changes address the hypothesized failure, and records possible collateral changes. It then outputs A, B, or TIE, together with a confidence score.

For each unordered pair (p_i, p_j), we query the comparator in both presentation orders. A pair is counted as a win for p_i only if both orders, after mapping positions back to patch identities, select p_i; likewise for p_j. Disagreement, a tie, or a missing response is treated as a tie. This conservative rule removes direct lead-position wins from the tournament.

Pairwise outcomes are aggregated by Copeland score: a pairwise win contributes one point, and a tie contributes no point to either patch. The patch with the largest score is selected; remaining ties are broken by proposal index.

Critic-gated comparator. The combined selector first filters patches by critic support and then runs the pairwise tournament on the survivors. If no patch passes the gate, it falls back to the highest-scoring patch under the critic. Unless otherwise stated, the gated-comparator ablations use yes-rate threshold $\tau = 0.5$. Threshold sweeps are offline diagnostics.

All selector prompts require strict JSON. A binary-judge parse failure is an abstention and contributes no positive vote. A missing or unparseable pairwise response is treated as a tie.

C.3 EVALUATION AND ABLATIONS

We vary proposal and judging budgets separately. Proposal-budget curves restrict the cached eight-patch pool to subsets of size k and recompute both oracle and selector performance. When several subsets of a given size are available, we average over them rather than privilege a proposal ordering. Vote-budget curves reuse the same proposal pool while varying the number of critic votes per patch or comparator votes per pair.

For the repository-level failure decomposition, each task falls into one of three disjoint groups:

- **solved:** the selected patch passes the hidden tests;
- **oracle reachable, missed:** some cached patch passes, but the selector does not choose a passing patch; or
- **oracle unreachable:** none of the cached patches passes.

The second group is selection error on the fixed pool; the third is proposal failure at that budget.

The main metric is hidden-test solve rate. We also report oracle best-of-k and the oracle-gap recovery ratio from Section 6. For the fixed-pool analyses, p_1 is the mean success rate of a uniformly chosen proposal from the pool; this is distinct from the separate 67.0% headline single-run baseline.

C.4 SELECTOR PROMPTS

The issue text is truncated to 2000 characters and each patch to 8000 characters before judging.

Binary patch-judge prompt.

Binary patch-judge prompt

```
You are evaluating a single patch against the GitHub issue it claims to fix.
Output yes if and only if running the patched code passes the failing test
described in the issue and does not break any test that previously passed.

Trace the failure path from the issue to the originating code. Check whether
```

```
the patch modifies that code path. Then state whether the patch produces the
expected behavior on the smallest input that exhibits the failure.

ISSUE:
{problem_statement}

PROPOSED PATCH:
{patch}

Respond with EXACTLY this JSON, no markdown fences, no extra text:
{
  "resolves": <true|false>,
  "reasoning": "<one sentence: smallest input under original code produces X; under patch
     produces Y>",
  "confidence": <integer 1-5>
}
```

Malformed responses, refusals, or parse failures are treated as abstentions. The default binary score is the number of positive votes, $s_{\text{yes}}(p) = \#\{v : v.\text{resolves} = \text{true}\}$. The confidence-weighted ablation instead uses the sum of reported confidence over positive votes.

Pairwise comparator prompt.

Pairwise comparator prompt

```
You are deciding which of two patches is more likely to make the failing
tests pass without breaking the passing tests. That is the only question
you are answering. Aesthetics, ambition, and visible effort are not relevant.

Do NOT select a patch because it makes more changes, addresses more cases,
or appears more thorough. The patch that minimally resolves the stated
issue is preferred. A 4kb patch that fixes the bug beats an 11kb patch
that doesn't.

ISSUE:
{problem_statement}

PATCH A:
{patch_a}

PATCH B:
{patch_b}

REQUIRED STRUCTURAL COMPARISON (fill these in BEFORE deciding):

  1. Failing test hypothesis: state the smallest hypothesis about what is
     wrong, derived from the issue text alone. One sentence. This is the
     ground truth you compare both patches against.

  2. A_changes: list the specific lines/functions Patch A modifies.
  3. B_changes: list the specific lines/functions Patch B modifies.

  4. A_consistent_with_hypothesis: do A's changes plausibly cause the
     failing test in the issue to start passing? (true/false + one-line
     justification)
  5. B_consistent_with_hypothesis: same question for B.

  6. A_collateral: does A change behavior on inputs unrelated to the
     failure mode? (true/false + one-line justification)
  7. B_collateral: same question for B.

The decision falls out of this comparison; do not pull a winner from prior.
If exactly one patch is consistent with the hypothesis and the other is
not, that one wins. If both are consistent, prefer the one with less
collateral. If both fail the hypothesis, output TIE. If they are
functionally equivalent (same lines changed differently, or different
lines that produce the same behavior), output TIE.

Respond in this EXACT JSON format -- JSON only, no markdown fences, no extra text:
{
  "hypothesis": "<one sentence>",
  "a_changes": "<files/functions modified by A>",
  "b_changes": "<files/functions modified by B>",
  "a_consistent": <true|false>,
  "b_consistent": <true|false>,
```

Table 1: **Selector ablations on SWE-bench Verified.** All rows use the same $k = 8$ `GPT-5.4 nano` proposal pool. Tournament rows vary the `GPT-5.4 nano` comparator aggregation rule with 5 comparator calls per pairwise matchup. Threshold rows vary the five-critic gate with 5 critics and 5 comparators. Oracle-gap recovery is computed using the metric defined in Section 6.

Ablation	Setting	Resolve rate	Oracle-gap recovery
Tournament rule	Copeland round-robin	75.8%	73.3%
Tournament rule	Sequential king	75.6%	71.7%
Tournament rule	Strict dominance	75.8%	73.3%
Tournament rule	Single-elim bracket	74.6%	63.3%
Critic threshold	$\tau \geq 0$ no gate	75.8%	73.3%
Critic threshold	$\tau \geq 1$, drop 0-yes patches	**76.4%**	**78.3%**
Critic threshold	$\tau \geq 2$, tied with $\tau \geq 1$	**76.4%**	**78.3%**
Critic threshold	$\tau \geq 3$, starts losing 1 instance	76.2%	76.7%
Critic threshold	$\tau \geq 4$	76.2%	76.7%
Critic threshold	$\tau \geq 5$, unanimous gate	76.0%	75.0%

```
  "a_collateral": <true|false>,
  "b_collateral": <true|false>,
  "winner": "A" | "B" | "TIE",
  "confidence": <integer 1-5>,
  "reasoning": "<one sentence: why the winner falls out of the structural comparison>"
}
```

If a pairwise response remains malformed after the allowed retries, it is treated as missing and therefore as a tie under the conservative aggregation rule above.

C.5 Selector ablations

Table 1 uses the same cached $k = 8$ `GPT-5.4 nano` pool throughout. The tournament block varies the aggregation rule with five comparator calls per matchup. Copeland round-robin and strict dominance reach 75.8%, while single elimination reaches 74.6%. The threshold block shows a different effect: a permissive critic gate improves on pure Copeland, while stricter gates begin to remove useful patches. This is consistent with the two-sided-critic extension in Proposition 3: repeated rejection is useful only while false rejection remains controlled.

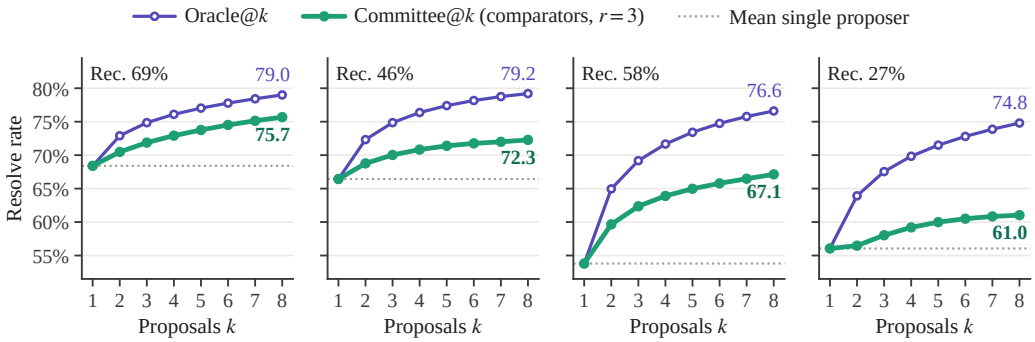
D Additional Empirical Results

This section provides the detailed empirical evidence behind the main-text scaling, identifiability, and resource results. We first standardize the fixed-pool protocol, then report cross-model scaling, controlled proposal-width experiments, simple selection controls, judge diagnostics, resource accounting, and additional boundary cases.

D.1 Protocol Notes

All additional experiments use fixed proposal pools: proposals are generated first, hidden outcomes are withheld from the selector, and hidden outcomes are used only afterward for scoring and oracle diagnostics. The cross-model SWE-bench comparison uses the same three-vote comparator-only replay for all four pools; the headline nano result remains the separate full critic–comparator run described in the main text.

For MATH L4–5 and GSM-Plus, Qwen2.5 proposers are judged by Qwen3-1.7B. LiveCodeBench uses Qwen3.6-35B-A3B proposals and pairwise judging, with public tests as a free control. miniF2F uses Goedel-Prover-V2-8B whole-proof proposals and Qwen3.6-35B-A3B selection without Lean outcomes. The additional Qwen3.6 tasks reuse saved five-proposal pools across first-proposal, plurality, critic-only, comparator-only, and full-committee policies.



(a) `GPT-5.4 nano` (b) Qwen3.6-27B (c) Qwen3.6-35B-A3B (d) Gemma-4-31B

Figure 5: **Cross-model scaling on SWE-bench Verified.** Oracle and common comparator-only selection as the proposal budget grows for four proposer models. Dotted lines show mean single-proposal performance.

Table 2: **Fixed-pool transfer results across tasks and proposer families.** SWE-bench committee values use the common $r = 3$ comparator-only replay; the headline nano result is the separate 76.4% critic–comparator run.

Benchmark	Proposer	n	k	Pass@1	Free ctrl.	Committee	Oracle@k	Rec.
SWE-bench Verified								
	GPT-5.4 nano	500	8	68.4	70.0[a]	75.7	79.0	69%
	Qwen3.6-27B	500	8	66.4	67.4[a]	72.3	79.2	46%
	Qwen3.6-35B-A3B	500	8	53.8	57.4[a]	67.1	76.6	58%
	Gemma-4-31B	500	8	56.0	62.6[a]	61.0	74.8	27%
MATH L4–5	Qwen2.5-3B	500	8	48.8	58.6[b]	71.2	75.0	85%
GSM-Plus	Qwen2.5-1.5B	1,319	8	49.1	66.9[b]	75.5	83.6	77%
LiveCodeBench	Qwen3.6-35B-A3B	258	5	70.2	75.2[c]	75.2	81.0	46%
miniF2F-hard	Goedel-Prover-V2-8B	164	5	19.5	–	25.0	32.3	43%

[a] String-similarity consensus; [b] majority vote; [c] public-test filter.

D.2 Cross-Model Scaling

The cross-model study asks whether the SWE-bench pattern is specific to the headline nano pool. Figure 5 applies the same three-vote comparator-only selector to four fixed proposal pools. Oracle coverage rises with proposal width in every pool, but the fraction recovered by selection differs substantially: at $k = 8$, recovery is 69%, 46%, 58%, and 27% for nano, Qwen3.6-27B, Qwen3.6-35B-A3B, and Gemma-4-31B, respectively. Table 2 reports the corresponding endpoints together with the cross-domain results used in Figure 3a.

D.3 Controlled GSM-Plus Scaling

To isolate proposal width from changes in the proposer or judge, we subsample a single saved eight-proposal GSM-Plus pool and average over subsets of each size. The oracle rises from 49.1% at $k = 1$ to 83.6% at $k = 8$. Pairwise selection tracks this increase while recovering 76.7%–79.5% of the oracle-exposed gain for $k = 2, 4, 8$. In contrast, the pointwise critic increasingly loses resolution: its top-score tie rate rises from 55.6% to 92.6%. Table 3 gives the exact values behind Figure 2c.

D.4 Selection Controls

Simple heuristics provide a useful check on whether learned judging is needed. Their behavior is not stable across proposal pools: for example, shortest-patch selection reaches 62.8% on Gemma but only 29.4% on the Qwen3.6-35B-A3B pool. Similarity consensus is the only zero-cost rule here that exceeds random selection on all four pools, but the learned comparator is much stronger on

Table 3: **Controlled proposal-width sweep on GSM-Plus.** All rows are computed from the same saved eight-proposal pool by subset averaging. Pairwise selection recovers a nearly constant fraction of the oracle-exposed gain as k grows, while the pointwise critic increasingly ties candidates at its maximum score. At $k = 8$, the bootstrap 95% interval for pairwise oracle-gap recovery is $[0.727, 0.807]$. Critic statistics use 1,318 tasks because one critic response is missing; other columns use 1,319 tasks.

k	Oracle	Majority	Critic	Pairwise	Pairwise Rec.	Critic top-tie
1	49.1	49.1	–	49.1	–	–
2	63.6	49.6	57.0	60.2	76.7%	55.6%
4	75.0	59.7	61.2	69.7	79.5%	80.4%
8	83.6	66.9	62.8	75.5	76.7%	92.6%

Table 4: **Simple controls on the fixed $k = 8$ SWE-bench pools.** The committee and oracle are included as references; the remaining rows require no learned judge.

Selection rule	Nano	Qwen-27B	Qwen-A3B	Gemma-31B
Random pick	68.4	66.4	53.8	56.0
First proposal	68.4	68.6	54.8	61.8
Shortest patch	69.4	65.4	29.4	**62.8**
Fewest files	69.4	65.0	26.4	62.2
Similarity consensus	70.0	67.4	57.4	62.6
Committee ($r = 3$)	**75.7**	**72.3**	**67.1**	61.0
Oracle@8	79.0	79.2	76.6	74.8

three of them. The Gemma exception is therefore a genuine boundary case rather than evidence that one surface heuristic is generally sufficient.

Aggregation choice is secondary on the corrected Qwen pools: Copeland, forced-margin, confidence-weighted, and anti-tie variants differ by at most one percentage point. The larger variation comes from the quality of the available identification signal, not the tournament rule itself.

D.5 Detailed Identifiability Diagnostics

Table 5 connects observed judge behavior to the identifiability quantities used in the theory and main text. MATH and GSM-Plus have large empirical comparator edges, $\hat{\sigma} = 0.40$ and 0.38, and correspondingly high oracle-gap recovery. LiveCodeBench has a smaller edge and lower recovery. Cross-task comparisons are descriptive, since the proposers, judges, and tasks differ. The clean intervention is the final pair of SWE rows: on the same Qwen3.6-27B proposal pool, repository access raises $\hat{\sigma}$ from 0.04 to 0.09 and recovery from 24% to 40%.

D.6 Cost and Token Accounting

The resource accounting separates two questions: how much extra selection budget is useful on a fixed pool, and how large the judging stage is relative to proposal generation. On the nano SWE-bench pool, one comparator vote per pair reaches 75.48% at $\$0.969$ per task and three votes reach 76.16% at $\$1.104$; substantially larger comparison budgets do not improve accuracy. The token split is task-dependent, however: judging is only 2.8% of logical model tokens in the reported SWE configuration but 98% on MATH and GSM-Plus. Thus the inexpensive-selector observation is specific to the SWE deployment rather than a universal property of orchestration.

D.7 Additional Tasks and Verifier Boundary Cases

These experiments probe cases in which learned selection is either unnecessary or weakly aligned with the task. Blocksworld is the clearest verifier-backed boundary: a public plan checker reaches the same 99.0% success as the learned committee on the saved five-proposal pool while using sub-

Table 5: **Empirical identifiability diagnostics.** TPR is critic acceptance on correct candidates; $\hat{\beta}$ is critic rejection rate on incorrect candidates; τ is comparator tie rate; a is accuracy conditional on a decisive comparison; and swap agr. is agreement across presentation orders. The empirical comparator edge is $\hat{\sigma} = (1 - \tau)(a - \frac{1}{2})$, as defined in Section 7.3.

Setting	Judge	TPR	$\hat{\beta}$	τ	a	swap agr.	$\hat{\sigma}$	Rec.
MATH L4–5	Qwen3-1.7B	0.94	0.69	0.15	0.98	0.81	0.40	85%
GSM-Plus	Qwen3-1.7B	0.96	0.65	0.18	0.97	0.85	0.38	77%
LiveCodeBench	Qwen3.6-35B-A3B	0.97	0.44	0.35	0.78	0.70	0.18	46%
SWE critic	Qwen3.6-27B	0.94	0.04	–	–	–	–	–
SWE comparator, patch only	Qwen3.6-27B	–	–	0.32	0.56	0.74	0.04	24%
SWE comparator, repo-aware	Qwen3.6-27B	–	–	0.33	0.63	0.74	0.09	40%

Table 6: **Selection budget on the fixed $k = 8$ nano pool.** Costs are modeled from measured role-level token profiles.

			Calls		Judge tok. (k)	$ / task			
m	r	Aggregation	critic	comp.	in / out	judge	total	Resolved	Rec.
–	–	random	0	0	0 / 0	0.000	0.899	68.40	0.0
0	3	top-3 pruned	0	9	16 / 18	0.023	0.922	72.47	38.4
0	3	top-4 pruned	0	17	30 / 35	0.046	0.945	73.92	52.1
0	3	bracket	0	20	35 / 41	0.052	0.951	75.20	64.2
0	1	all pairs	0	26	46 / 53	0.070	0.969	75.48	66.8
0	3	all pairs, early stop	0	78	138 / 160	0.203	1.102	76.16	73.2
0	3	all pairs	0	78	138 / 160	0.205	1.104	76.16	73.2
5	5	bracket	38	30	107 / 193	0.246	1.145	75.80	69.8
0	5	all pairs	0	131	232 / 268	0.341	1.240	76.19	73.5
5	3	all pairs	38	69	176 / 273	0.348	1.247	76.36	75.1
5	5	all pairs	38	115	257 / 367	0.467	1.366	76.13	73.0
0	15	all pairs	0	368	652 / 753	0.955	1.854	75.89	70.7

stantially fewer model tokens. The remaining tasks show smaller gains, and CommonGen gives a negative control in which comparator-based selection reduces success.

On the 598 completed Blocksworld tasks, the first proposal solves 59.5% and both the committee and public verifier reach 99.0%. On all 600 intended tasks, counting two task errors as failures, the corresponding rates are 59.3% and 98.7%. The full committee uses 113.70M logical model tokens, while checking the five proposals with the public verifier uses 18.10M proposal tokens; stopping after the first verified plan would use 6.08M proposal tokens on the saved draws. When a cheap sound verifier exists, it should be preferred to learned judging.

CommonGen is a useful negative control: comparator-based selection is misaligned with exact required-concept coverage and reduces success from 80.5% to 73.0%. Orchestration is therefore not generically monotone; its value depends on the alignment of the local identification signal with the terminal objective.

D.8 Preliminary IFBench Result

On the 220 prompts with complete selections, loose prompt-level success changes from 29.1% to 31.4% with oracle@5 45.5%; strict prompt-level success is unchanged at 22.3% with oracle@5 37.3%. Because the intended 300-prompt run is incomplete, we report these values only as preliminary diagnostics and exclude them from the main empirical claims.

E Extended Literature Review

The literature on inference-time reasoning spans several traditions that are close to our setting. We organize them around two questions: whether additional computation increases the chance of producing a useful candidate, and whether the system can identify that candidate once it appears.

Boosting and weak-to-strong amplification. Classical boosting combines weak predictive signals into a stronger predictor under supervised feedback (Schapire, 1990; Freund & Schapire, 1995;

Table 7: **Per-task model-token usage** (thousands) for proposal generation and judging. Judge share is the fraction of total logical model tokens attributable to the judging stage.

Experiment	n	k	Prop. in	Prop. out	Judge in	Judge out	Judge share
SWE-bench (nano, $m = 5, r = 3$)	500	8	15,102.2	400.6	175.8	272.5	2.8%
MATH L4–5	500	8	1.1	5.7	136.1	194.8	98%
GSM-Plus	1,319	8	1.0	2.6	80.7	136.6	98%
LiveCodeBench v5–v6	258	5	3.3	135.1	59.1	348.9	75%
miniF2F-hard	164	5	–	41.6	–	97.7	70%
Blocksworld	598	5	2.4	27.8	6.6	153.3	84%

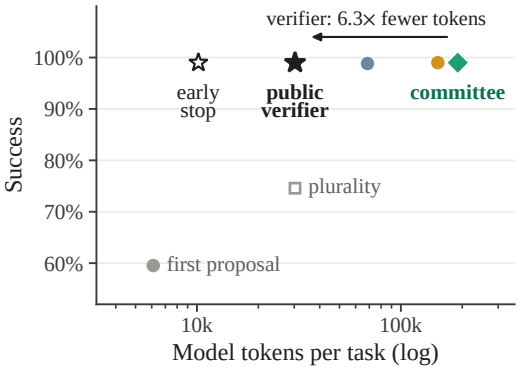


Figure 6: **Direct verification on Blocksworld.** A public plan checker matches the learned selector on the same five-proposal pool while requiring substantially fewer model tokens.

1997), while bagging, stacking, and related ensemble methods combine multiple predictors for variance reduction or robustness (Breiman, 1996; Wolpert, 1992; Dietterich, 2000). Several recent works bring related ideas to language models. (Manikandan et al., 2023) use language models as weak learners in a supervised boosting pipeline; weak-to-strong generalization studies whether weak supervision can elicit stronger behavior from a more capable model (Burns et al., 2024); and iterated amplification uses decomposition to let weak experts supervise harder problems (Christiano et al., 2018). LLMBoost (Chen et al., 2025) combines models through training and representation sharing. Our setting keeps model weights fixed and studies amplification entirely at inference time.

Sampling and test-time scaling. Self-consistency, rationale ensembles, and Universal Self-Consistency improve reasoning by drawing several outputs and aggregating them (Wang et al., 2023; 2022; Chen et al., 2023). More Agents Is All You Need studies repeated agent calls (Li et al., 2024); Large Language Monkeys studies how repeated sampling changes task coverage (Brown et al., 2024); and work on compound-system scaling and best-of-N characterizes how performance changes with inference-time compute (Chen et al., 2024; Huang et al., 2025). These papers motivate the coverage side of our analysis. Additional samples matter only when they expose candidates that were absent from a single draw.

Verification and selection. A separate line of work asks how to choose among sampled candidates. Learned verifiers select mathematical solutions (Cobbe et al., 2021); process supervision scores intermediate reasoning steps (Lightman et al., 2024; Wang et al., 2024); and pairwise rankers or verifier ensembles combine imperfect selection signals (Jiang et al., 2023; Lifshitz et al., 2025; Saad-Falcon et al., 2025a; Lee et al., 2026). LLM-as-a-judge and reward-model benchmarks show that selectors have their own biases and failure modes (Zheng et al., 2023; Lambert et al., 2025; Gu et al., 2026). This is the distinction captured by identifiability in our theory: a useful candidate can be present without being recoverable by the available selector.

Search, tools, and agentic systems. Tree of Thoughts, RAP, LATS, ReAct, Reflexion, and Self-Refine operate over intermediate states using search, feedback, planning, or revision (Yao et al., 2023a; Hao et al., 2023; Zhou et al., 2024; Yao et al., 2023b; Shinn et al., 2023; Madaan et al., 2023). CRITIC, Chain-of-Verification, Toolformer, and HuggingGPT add external evidence or computation (Gou et al., 2024; Dhuliawala et al., 2024; Schick et al., 2023; Shen et al., 2023). Multi-agent frameworks such as CAMEL, AutoGen, MetaGPT, Mixture-of-Agents, Archon, and Smoothie explore broader orchestration spaces (Li et al., 2023; Wu et al., 2024; Hong et al., 2024; Wang et al., 2025;

Table 8: **Selection policies on additional Qwen3.6-35B-A3B tasks.** Success is reported on completed selections (%). Δ is the full committee minus the first-proposal baseline, computed from unrounded scores. Tokens reports full-committee model-token use relative to the first-proposal baseline.

Task	n	First	Plurality	Critics	Comparators	Committee	Δ	Verifier	Tokens
Blocksworld	598	59.5	74.6	98.8	**99.0**	**99.0**	+39.5	99.0	31×
IFEval	532	88.9	88.9	**91.4**	91.0	91.0	+2.1	–	26×
SocialIQA	1,954	82.5	83.1	82.9	83.4	**83.5**	+1.0	–	12×
AlphaNLI	1,532	86.2	**87.0**	86.3	86.4	86.3	+0.1	–	11×
CommonGen	992	80.5	80.5	**80.7**	72.7	73.0	-7.6	–	28×

Saad-Falcon et al., 2025b; Guha et al., 2024). Our analysis abstracts away from these architectural choices and isolates three requirements for reliable amplification: useful moves must be proposed, distinguishable, and composable along a bounded-progress trajectory.

Verifier-backed code and formal reasoning. Code and formal reasoning are natural settings for this question because candidate artifacts can often be checked by execution, tests, types, or proof assistants. AlphaCode uses large-scale sampling and behavioral filtering (Li et al., 2022); SWE-bench and systems such as SWE-agent, AutoCodeRover, and Agentless study repository-level repair with structured generation and validation (Jimenez et al., 2024; Yang et al., 2024; Zhang et al., 2024; Xia et al., 2025). These settings make it possible to measure separately whether a correct candidate was generated and whether it was selected.

Benchmarking implication. For verifier-backed tasks, reporting pass@1 alone conflates proposal quality with selection. We therefore report pass@1, oracle best-of-k, implemented system performance, oracle-gap recovery, and resource use. Together these quantities separate one-shot success, proposal-pool coverage, selection quality, and the cost of approaching the proposal-side ceiling.