MEASURING AND AMPLIFYING SELF-VERIFICATION ON HARD PROBLEMS

Anonymous authors
Paper under double-blind review

ABSTRACT

An open question in scalable oversight is whether language models can assess solutions to problems that they cannot reliably solve. We test this by separating an assessor's informedness from its leniency. Informedness refers to a binary assessor's ability to distinguish correct from incorrect solutions. Leniency refers to its tendency to accept. Across six language models and seven benchmarks in mathematics, coding, and knowledge, we evaluate assessment on problems the model itself rarely solves (own solve rate ≤ 0.2). We find that informedness on difficult problems strongly depends on task family: it is significantly higher than chance on mathematics and coding, but near the chance baseline on knowledge tasks. Next, we investigate committees of assessors and find that they can increase informedness compared to a single assessor. However, the gain is not automatic: with majority voting, the committee is worse than a single assessor in 6 of 18 settings. We therefore apply a simple mathematical model to predict committee behavior and select consensus thresholds. Taken together, our results identify task family, assessor behavior, and aggregation rule as key determinants of whether oversight remains effective beyond a model's own solving capability.

1 INTRODUCTION

Language models are increasingly deployed in settings where direct human oversight may not scale with either model capability or deployment volume. Scalable oversight addresses this by asking whether weaker models can reliably supervise stronger ones (Christiano et al., 2018; Irving et al., 2018; Bowman et al., 2022; Wen et al., 2025; Engels et al., 2025). A natural intermediate case is self-assessment beyond a model's own solving capability. If reliable assessment already breaks down in this matched-model setting, it becomes harder to justify relying on weaker models to supervise more capable ones. This motivates our central question:

> When a language model cannot solve a problem itself, can it nevertheless tell a correct solution from an incorrect one?

Empirical tests of this question have produced mixed results. Language models often struggle to detect errors in their own answers (Luo et al., 2023; Jiang et al., 2025; Kamoi et al., 2024), yet step-level verification can be easier than generation on frontier mathematics (Pandit et al., 2025) and the relative difficulty of generation and evaluation varies across tasks (Bandyopadhyay, 2026). Two limitations prevent these studies from fully answering our central question. First, generation and verification are typically compared in aggregate over a benchmark (but see Luo et al., 2023; Pandit et al., 2025), which mixes problems the model can solve with those it cannot. Second, assessment is usually summarized by a single statistic that depends on the class prevalence, such as accuracy. Prevalence-independent summaries such as balanced accuracy remove this dependence (Powers, 2011; Collot et al., 2026), but do not capture the assessor's tendency to accept.

In this work, we address these limitations. We measure the assessor model's verdicts on solutions to problems that the model itself rarely solves. We characterize each assessor by two complementary coordinates: *informedness* $J = \mathrm{TPR} + \mathrm{TNR} - 1$, which measures discrimination between correct and incorrect proposals, and *leniency* $L = (\mathrm{TPR} - \mathrm{TNR} + 1)/2$, which captures the tendency to accept. Our contributions are threefold:

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.

1. We show that informedness and leniency play distinct roles in the utility an assessor adds to a system: increasing J always improves assessor utility, whereas the optimal L depends on proposer accuracy and the relative costs of incorrect answers and abstention. The same analysis also shows that the sign of the generation–verification gap (Song et al., 2025) is determined by J.
2. Across six language models and seven benchmarks, we find that self-verification on problems a model rarely solves is task-dependent: informedness remains above chance on mathematics and coding but is indistinguishable from chance on knowledge tasks. This gap holds for all six models and across difficulty cutoffs.
3. We show that (k/N) committees of repeated assessor samples can increase informedness, but that majority voting captures little of this gain: it is worse than a single assessor in 6 of 18 settings. We derive the committee's (L, J) operating point from moments of the single assessor's acceptance propensity, and show that a simple Beta model accurately predicts the resulting operating curve, allowing the consensus threshold k to be selected from limited single-assessor measurements.

2 PRELIMINARIES

We consider two roles for language model agents: a **proposer**, which produces a solution to a request, and an **assessor**, which judges the solution generated by the proposer. Their induced conditional distributions are denoted by $\mathcal{P}$ and $\mathcal{A}$. Each agent consists of a language model together with its *harness*, i.e. the prompts and procedures governing its inputs and outputs.[1] For a request r, the proposer samples a proposal y from the proposal space $\mathcal{Y}$, and the assessor returns a binary verdict v, with $v = 1$ denoting acceptance:

$$y \sim \mathcal{P}(\cdot \mid r), \quad y \in \mathcal{Y}, \qquad v \sim \mathcal{A}(\cdot \mid r, y), \quad v \in \{0, 1\}. \tag{1}$$

We write $q(r, y) \equiv \mathcal{A}(1 \mid r, y)$ for the proposal's **acceptance propensity**, and assume repeated verdicts for fixed (r, y) are i.i.d. Bernoulli draws with parameter $q(r, y)$.

Let $G_r(y) \in \{0, 1\}$ denote the ground-truth grade of proposal y for request r, with $G_r(y) = 1$ indicating correctness. We set $G_r(y) = 0$ for all unsuccessful proposals, whether incorrect, truncated, or unanswered (Section B). We define

$$\mathcal{Y}_r^+ \equiv \{y : G_r(y) = 1\}, \qquad \mathcal{Y}_r^- \equiv \{y : G_r(y) = 0\}. \tag{2}$$

The proposer's success probability (accuracy) is

$$p(r) \equiv \sum_{y \in \mathcal{Y}} \mathcal{P}(y \mid r) G_r(y) = \sum_{y \in \mathcal{Y}_r^+} \mathcal{P}(y \mid r). \tag{3}$$

The assessor's true- and false-positive rates are the mean acceptance propensities on correct and incorrect proposals:

$$\mathrm{TPR}(r) \equiv \frac{1}{p(r)} \sum_{y \in \mathcal{Y}_r^+} \mathcal{P}(y \mid r) q(r, y), \qquad \mathrm{FPR}(r) \equiv \frac{1}{1 - p(r)} \sum_{y \in \mathcal{Y}_r^-} \mathcal{P}(y \mid r) q(r, y). \tag{4}$$

The true-negative rate is $\mathrm{TNR} = 1 - \mathrm{FPR}$. Because these rates depend jointly on the proposer and assessor, we occasionally write $\mathrm{TPR}(r; \mathcal{P}, \mathcal{A})$ and similarly for the other rates. In the **self-assessment** setting, proposer and assessor use the same underlying language model $\mathcal{M}$, possibly with different role-specific harnesses, and we write, for example, $\mathrm{TPR}(r; \mathcal{M})$.

Two reference cases are useful. A **perfect assessor** has $q(r, y) = G_r(y)$, giving $\mathrm{TPR} = \mathrm{TNR} = 1$. A **chance assessor** cannot distinguish correct from incorrect proposals, so its acceptance propensity is constant: $q(r, y) = \lambda \in [0, 1]$ for all y. Hence $\mathrm{TPR} = \lambda$ and $\mathrm{TNR} = 1 - \lambda$.

Accuracy alone does not measure discrimination because it also depends on the prevalence of correct proposals:

$$\text{Assessor Accuracy} = p\ \mathrm{TPR} + (1 - p)\ \mathrm{TNR}\,. \tag{5}$$

For example, an always-accept chance assessor achieves accuracy p despite having no discrimination ability.

[1] See Appendix B for additional information on the harnesses.

3 METHODS

3.1 SEPARATING INFORMEDNESS AND LENIENCY

Consider two teachers grading the same set of essays. They may differ in their ability to assess the quality of the arguments presented; this is a difference in skill. They may also differ in how readily they deduct points; this is a difference in disposition. As a result, the more lenient teacher may assign systematically higher grades even if the two teachers are equally skilled. The same issue arises here: TPR and TNR reflect both the assessor's ability to distinguish correct from incorrect proposals and its tendency to accept. Our goal is to separate these two components.

We therefore reparameterize the assessor's operating point by *informedness* J and *leniency* L, defined by

$$J \equiv \mathrm{TPR} + \mathrm{TNR} - 1, \qquad L \equiv \frac{\mathrm{TPR} - \mathrm{TNR} + 1}{2}. \tag{6}$$

Equivalently,

$$\mathrm{TPR} = \frac{J}{2} + L, \qquad \mathrm{TNR} = \frac{J}{2} + (1 - L). \tag{7}$$

The first coordinate is Youden's J, or informedness (Youden, 1950; Powers, 2011). Here, $J = 0$ corresponds to chance-level discrimination and $J = 1$ to perfect discrimination. The complementary leniency coordinate L captures the assessor's tendency to accept. For a chance assessor, $L = \lambda$, ranging from always reject at $L = 0$ to always accept at $L = 1$.

3.2 INTERPRETATION THROUGH UTILITY FUNCTION

We now provide further justification for why J may be treated as capability and L as leniency, by asking what utility an assessor adds to a system. Upon receiving a request r, a system built with language model $\mathcal{M}$ can respond correctly, incorrectly, or abstain. Suppose the reward for a correct response is normalized to 1, while incorrect responses and abstention have penalties γ and δ, respectively. Then the utility of the bare proposer u_{bare} is

$$u_{\text{bare}} = p - \gamma(1 - p), \tag{8}$$

and for a proposer–assessor system that generates a proposal, evaluates it with the assessor, and abstains if the proposal is rejected, the utility $u_{\text{proposer-assessor}}$ is

$$\begin{aligned} u_{\text{proposer-assessor}} &= p\,\mathrm{TPR} - \delta p\,\mathrm{FNR} - \gamma(1-p)\,\mathrm{FPR} - \delta(1-p)\,\mathrm{TNR} \\ &= L\big(\underbrace{p - \gamma(1-p)}_{u_{\text{bare}}} + \delta\big) + \frac{J}{2}\big(p + \gamma(1-p) + \delta(2p-1)\big) - \delta. \end{aligned} \tag{9}$$

The penalty δ captures the cost associated with abstaining rather than returning a proposal.

Now consider $\Delta u \equiv u_{\text{proposer-assessor}} - u_{\text{bare}}$, the additional utility that an assessor brings. We are interested in cases where the penalty of an incorrect response is higher than abstention ($\gamma > \delta$). In those cases, we find

$$\begin{aligned} \frac{\partial \Delta u}{\partial J} &= \frac{p(1+\delta) + (1-p)(\gamma - \delta)}{2} > 0 \\ \frac{\partial \Delta u}{\partial L} &= p - \gamma(1-p) + \delta, \quad \text{greater than 0 iff } p > \frac{\gamma - \delta}{1+\gamma}. \end{aligned} \tag{10}$$

In other words, increasing J always increases the utility provided by the assessor, whereas increasing leniency L is beneficial only when $p > \frac{\gamma-\delta}{1+\gamma}$. This gives an operational interpretation of J as assessor capability and L as leniency. As proposer accuracy increases, the utility-optimal operating point shifts toward greater leniency and less abstention. We therefore seek to maximize J while calibrating L to the task at hand.

This view also connects informedness to the generation–verification gap (GV-Gap) of Song et al. (2025), defined as the change in expected task utility obtained by reweighting a generator's output distribution using a verifier. We specialize their definition to our binary assessor by assigning weight

1 to accepted proposals and 0 to rejected proposals. Taking task utility to be accuracy corresponds to $\gamma = 0$ in our notation; the abstention penalty δ is not relevant. For a request with proposer accuracy p, the resulting per-request contribution to the GV-Gap is therefore

$$\text{GV-Gap}(r) = \frac{p\text{TPR}}{p\text{TPR} + (1-p)\text{FPR}} - p = \frac{p(1-p)J}{L + J\left(p - \frac{1}{2}\right)}. \tag{11}$$

The sign of the GV-Gap is determined by J, while its magnitude also depends on the proposer success probability p and the assessor's operating point L.

3.3 The Flipped Assessor as Intervention Baseline

Can an assessor's leniency be shifted after its verdicts have been produced? Consider post-processing the verdict by flipping accept to reject with probability a and reject to accept with probability b: an (a, b)-**flipped assessor**. This induces

$$\text{TPR} \to b + (1 - a - b)\,\text{TPR}, \quad \text{TNR} \to a + (1 - a - b)\,\text{TNR}, \tag{12}$$

or, in (L, J) coordinates,

$$L \to b + (1 - a - b)L, \qquad J \to (1 - a - b)J. \tag{13}$$

Flipping accepts to rejects raises TNR at the expense of TPR, while flipping rejects to accepts does the reverse. Since $|1 - a - b| \leq 1$, no such post-processing can increase $|J|$. As (a, b) ranges over $[0, 1]^2$, the reachable operating points form a parallelogram in (L, J) space with vertices (L, J), $(0, 0)$, $(1, 0)$, and $(1 - L, -J)$. This probabilistically flipped-assessor provides a zero-cost baseline for interventions that aim to control leniency.

3.4 The (k/N) committee

A common alternative to navigate the (TPR, TNR) space is to combine multiple assessors in a (k/N) committee: among N independent assessors, the committee accepts a proposal when at least $k \in \{1, \ldots, N\}$ assessors accept.

The question is which k, N pairs to pick. The cost of the committee is proportional to N, while committee leniency decreases with higher threshold k. This setting is an instance of determining optimal group decision-making policies for a fixed-size committee, which has been explored extensively in economics (Shapley & Grofman, 1984; Sah & Stiglitz, 1988; Ben-Yashar & Nitzan, 1997). We apply this framework to predict a committee's operating point and compare the predictions with observed performance.

We follow the notation in Sah & Stiglitz (1988). For a given proposal, suppose each assessor independently emits an accept verdict with probability q. The probability that a (k/N) committee accepts is then

$$\begin{aligned} q_{k/N} &= \sum_{j=k}^{N} \binom{N}{j} q^j (1-q)^{N-j} \\ &= \sum_{j=k}^{N} c_j q^j, \qquad c_j \equiv (-1)^{j-k} \binom{N}{j} \binom{j-1}{k-1}, \end{aligned} \tag{14}$$

where the second line follows by expanding and rearranging the binomial terms. To obtain the committee's TPR and FPR, we average $q_{k/N}$ over the correct and incorrect proposal classes $\mathcal{Y}^{\pm}$. Defining the class-conditional moments

$$m_{\pm}^{(j)} \equiv \langle q^j \rangle_{\pm}, \tag{15}$$

where the $+$ and $-$ subscripts denote expectations over the distributions of correct and incorrect proposals, respectively, we obtain

$$\text{TPR}_{k/N} = \sum_{j=k}^{N} c_j m_{+}^{(j)}, \qquad \text{FPR}_{k/N} = \sum_{j=k}^{N} c_j m_{-}^{(j)}. \tag{16}$$

Predicting the performance of a homogeneous committee therefore reduces to estimating higher moments of the assessor's proposal-level acceptance propensity.

3.5 Predicting Committee Performance and Selecting Consensus

Predicting an N-member committee requires moments up to order N. We approximate these higher moments using either the class means alone or a Beta distribution fitted to the first two moments.

Mean-only approximation. The mean-only approximation treats the propensity as constant within each class, giving

$$\widehat{m}^{(j)}_{\pm,\mathrm{mean}} = \left(\hat{m}^{(1)}_{\pm}\right)^{j}. \tag{17}$$

This uses only the single assessor's TPR and FPR, ignoring variation across proposals within each class.

Beta approximation. Beta distributions form a two-parameter family on $[0, 1]$ with closed-form expressions for all higher moments. Assuming the acceptance propensities can be approximated by a Beta distribution, we estimate α and β by matching the first two moments of q, then use the fitted distribution to approximate the higher moments required for committee prediction. Additional details are given in Appendix D.

Selecting consensus threshold. For a fixed committee size N, both approximations give predicted informedness $\widehat{J}_{k/N} = \widehat{\mathrm{TPR}}_{k/N} - \widehat{\mathrm{FPR}}_{k/N}$. We select the consensus threshold that maximizes this prediction:

$$\hat{k} \in \underset{k\in\{1,\ldots,N\}}{\arg\max}\ \widehat{J}_{k/N}. \tag{18}$$

Applying this rule separately to the mean-only and Beta approximations yields two consensus-threshold selection methods. We evaluate both the informedness achieved by the selected thresholds and the accuracy of the predicted committee operating points using held-out verdicts.

4 Experiments

We evaluate six language models (Llama-4-Maverick-17B, Qwen3-32B, Qwen3-235B-A22B, GPT-OSS-20B, GPT-OSS-120B, and GPT-5.6-Luna) on seven benchmarks spanning mathematics (AIME2026, BRUMO2025, SMT2025, and ApexShortList; Dekoninck et al., 2026), knowledge (GPQA-Diamond; Rein et al., 2024, and MMLU-Pro; Wang et al., 2024), and coding (LiveCodeBench; Jain et al., 2025). Knowledge questions are multiple-choice, whereas mathematics and coding are open-ended. Each model serves as both proposer and assessor under the same inference settings. Full benchmark, model, prompting, and grading details are provided in Appendix B.

Our data collection consists of four steps. **1. Initial screening.** Each model produces 20 independent proposals per question, and we retain model–question pairs containing at least one correct and one unsuccessful non-error proposal, where unsuccessful includes incorrect, truncated, and no-answer generations. **2. Proposal sampling.** For each retained pair, we sample 80 additional proposals, yielding 100 proposals from which we estimate proposer accuracy $\hat{p}$. **3. Verdict sampling.** Each proposal receives two independent assessor verdicts from the same model. **4. Committee sampling.** A 10% subset of proposals receives 16 additional verdicts for fitting and evaluating assessor committees. Full sampling, grading, and filtering procedures are given in Appendix B.

We exclude invalid assessor verdicts and retain only model–question pairs containing both correct and unsuccessful proposals with valid assessor verdicts, so that both TPR and TNR are estimable. For the symmetric chance reference ($\mathrm{TPR} = \mathrm{TNR} = 0.5$) analyzed in Appendix A, the confidence-interval width scales as $1/\sqrt{\hat{p}(1-\hat{p})}$, illustrating the rapid growth of uncertainty of J as problems become easy or hard. At $\hat{p} \in \{0, 1\}$, one class is absent and either TPR or TNR is undefined, so we analyze only model–question pairs with $0 < \hat{p} < 1$. We choose to define hard problems as $\hat{p} \leq 0.2$, additionally we perform a sensitivity analysis.

Filtering yields overlapping but model-specific question sets spanning a broad range of proposer accuracies; retained sample counts are reported in Appendix B. The additional verdicts are used to measure (k, N) committee operating points and compare them with the predictions of Section 3.5.

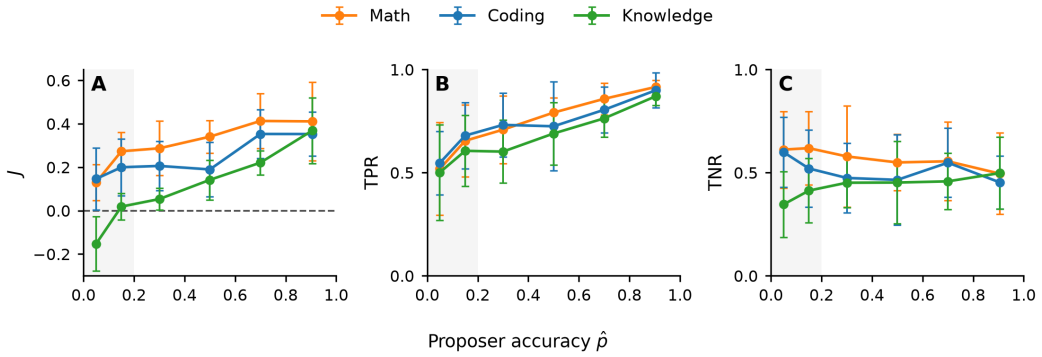

Figure 1: **Self-verification performance as a function of proposer accuracy. A.** Model-balanced informedness J across proposer-accuracy bins. **B–C.** Corresponding true-positive rate (TPR) and true-negative rate (TNR). For each bin, quantities are first averaged over questions within each model and then averaged across models, giving equal weight to each model. Points are plotted at bin centers and error bars denote 95% confidence intervals over model-level means. The shaded region marks the hard-problem regime $\hat{p} \leq 0.2$ used in the primary analysis. The dashed horizontal line in panel A marks $J = 0$.

Response-format control. Because the knowledge benchmarks are multiple-choice whereas mathematics and coding are open-ended, we test whether knowledge self-assessment is sensitive to the assessor-facing response format. Using the same sampled knowledge proposals, we remove the explicit answer choices from the request and replace the final option label with the text of the selected answer, while otherwise preserving the proposal content. We resample assessor verdicts and compare TPR, TNR, J, and L between conditions on matched proposals; no proposals are regenerated or regraded. This control changes the assessor-facing presentation but does not eliminate multiple-choice cues retained in the proposal or effects of multiple-choice elicitation itself; full details are given in Appendix B.

5 Results

5.1 Verification informedness on hard problems

Self-verification informedness becomes increasingly task-dependent as proposer accuracy decreases (Fig. 1A). At high proposer accuracy, the three task families have similar J, but as $\hat{p}$ decreases, knowledge falls toward and below $J = 0$ while mathematics and coding remain positively informed. The task-family differences emerge primarily in the low-accuracy regime.

We examine this separation further using $\hat{p} \leq 0.2$ as the primary hard-problem regime (Table 1). The model-balanced informedness, obtained by first averaging over questions within each model and then averaging equally across models, is $J = 0.214$ $[0.148, 0.281]$ on mathematics, 0.162 $[0.027, 0.298]$ on coding, and -0.054 $[-0.132, 0.023]$ on knowledge. The negative point estimate for hard knowledge questions appears in both constituent benchmarks under the same $\hat{p} \leq 0.2$ cutoff: informedness is $J = -0.028$ $[-0.139, 0.083]$ on GPQA-Diamond and $J = -0.081$ $[-0.163, 0.001]$ on MMLU-Pro.[2] Comparing mathematics and knowledge within each model gives the paired difference [3]

$$\Delta J_{\text{math}-\text{knowledge}} = 0.269\ [0.174,\ 0.364], \tag{19}$$

with a positive difference for all six models. The math–knowledge gap is driven primarily by TNR: it accounts for approximately 84% of ΔJ, indicating reduced rejection of unsuccessful knowledge proposals (Fig. 1B–C; Table 1). The mathematics–knowledge contrast is robust to the hard-problem

[2]We report the two knowledge benchmarks separately because their multiple-choice formats have different random-choice accuracies (0.25 for GPQA-Diamond with 4 choices, and 0.10 for MMLU-Pro with 10 choices).

[3]The paired difference is computed by taking the mathematics–knowledge difference within each model and then averaging these six model-level differences, giving each model equal weight.

Primary hard regime: $\hat{p} \leq 0.2$				
Task	J	TPR	TNR	Mean $\hat{p}$
Math	0.214 [0.148, 0.281]	0.602 [0.408, 0.795]	0.613 [0.430, 0.795]	0.096 [0.081, 0.110]
Coding	0.162 [0.027, 0.298]	0.594 [0.445, 0.742]	0.569 [0.397, 0.740]	0.086 [0.078, 0.093]
Knowledge	-0.054 [-0.132, 0.023]	0.559 [0.369, 0.748]	0.387 [0.232, 0.542]	0.104 [0.092, 0.115]
Math − Knowledge	+0.269 [+0.174, +0.364]	+0.043 [-0.068, +0.153]	+0.226 [+0.062, +0.390]	-0.008 [-0.031, +0.016]
Sensitivity to the hard-problem cutoff				
Cutoff		$\Delta J_{\mathrm{math-knowledge}}$ [95% CI]		Models with $\Delta J > 0$
$\hat{p} \leq 0.1$		+0.275 [+0.158, +0.392]		6/6
$\hat{p} \leq 0.2$		+0.269 [+0.174, +0.364]		6/6
$\hat{p} \leq 0.3$		+0.252 [+0.163, +0.341]		6/6
$\hat{p} \leq 0.4$		+0.247 [+0.160, +0.333]		6/6
$\hat{p} \leq 0.5$		+0.233 [+0.160, +0.305]		6/6
$\hat{p} \leq 0.6$		+0.224 [+0.153, +0.296]		6/6

Table 1: **Self-verification on hard questions and sensitivity to the difficulty cutoff.** Upper: model-balanced metrics for $\hat{p} \leq 0.2$; Math–Knowledge reports paired model-level differences. Lower: paired mathematics–knowledge ΔJ as the cutoff varies. Brackets denote 95% confidence intervals over the six model-level means.

cutoff: across $\hat{p} \leq 0.1$ to $\hat{p} \leq 0.6$, $\Delta J_{\mathrm{math-knowledge}}$ remains positive for all six models, ranging from 0.275 to 0.224 (Table 1).

Mean proposer accuracy within the hard subset is similar for mathematics (0.096) and knowledge (0.104), so the lower knowledge informedness is not explained by lower mean proposer accuracy in the knowledge subset.

Because both knowledge benchmarks are multiple-choice whereas mathematics and coding are open-ended, we test sensitivity to the assessor-facing response format. On hard knowledge questions ($\hat{p} \leq 0.2$), removing the explicit answer choices and replacing the final option label with the selected option text changes TPR by $\Delta\mathrm{TPR} = -0.072\,[-0.126, -0.018]$ and TNR by $\Delta\mathrm{TNR} = +0.120\,[0.085, 0.155]$. The resulting change in informedness is smaller and statistically indistinguishable from zero, $\Delta J = 0.048\,[-0.035, 0.131]$. Thus, the explicit multiple-choice presentation shifts leniency but does not produce a clear systematic change in informedness. The control remains partial because the proposals were originally elicited in multiple-choice format and their explanations may retain multiple-choice cues.

5.2 Committee

Having characterized the assessor's operating point, we next ask whether additional assessor samples can increase informedness. For $N = 9$, majority voting does not reliably improve over a single assessor, with a model-balanced gain of $\Delta J_{\mathrm{majority}} = 0.011\,[-0.008, 0.027]$. By contrast, the best observed threshold for each model–task pair yields $\Delta J_{\mathrm{best}} = 0.066\,[0.056, 0.078]$. Majority performs worse than a single assessor in 6 settings (Table 2). Thus, most of the available committee gain lies at non-majority consensus thresholds.

Better consensus threshold k can be selected without the cost of evaluating the entire held-out committee results. We use 5 verdict slots per proposal for fitting and evaluate the resulting predictions on held-out verdicts for an $N = 9$ committee. Using only the mean acceptance rates on correct and unsuccessful proposals, the mean-only approximation selects thresholds with a model-balanced gain of $\Delta J_{\mathrm{mean}} = 0.049\,[0.038, 0.060]$. The mean-only selected rule is within 0.05 of the best observed J in 15 of 18 settings, better than the 9 of 18 for majority voting, and the resulted committee is always better than a single assessor. The Beta-selected threshold performs similarly, with a gain of $0.046\,[0.034, 0.059]$ and the same 15 of 18 settings within 0.05 of the best observed threshold (Table 2).

The gain from adaptive consensus is not merely a change in leniency: relative to the single-assessor flip baseline from Section 3.3, the mean-only selected committee reaches higher informedness at matched leniency in all 18 settings, with model-balanced excess $\Delta J_{\mathrm{flip}} = 0.066\,[0.053, 0.080]$.

Selecting a useful threshold and predicting the full committee operating curve are different tasks. Although the mean-only approximation selects competitive thresholds, it predicts the full (L, J)

Rule	ΔJ vs. single	Regret R to best observed k						
		Mean	Median	$\leq .01$	$\leq .02$	$\leq .05$	Worse	ΔJ_{flip}
Observed best k	$+0.066$ [$+0.056$, $+0.078$]	—	—	—	—	—	0/18	$+0.098$
Mean-only selected k	$+0.049$ [$+0.038$, $+0.060$]	0.018	0.003	10/18	12/18	15/18	0/18	$+0.066$
Beta selected k	$+0.046$ [$+0.034$, $+0.059$]	0.021	0.008	11/18	12/18	15/18	1/18	$+0.083$
Majority	$+0.011$ [-0.008, $+0.027$]	0.056	0.049	4/18	7/18	9/18	6/18	$+0.020$

Table 2: Consensus-threshold selection for an $N = 9$ committee, evaluated on held-out committee verdicts across the 18 model–task settings. ΔJ vs. single: model-balanced informedness gain over a single assessor, with a 95% model-cluster bootstrap interval, using models as the resampling units. Regret R: per-setting gap between the informedness of the selected threshold and the best observed threshold; we report its mean and median and the number of settings with R at or below each tolerance. Worse: settings in which the committee has lower J than a single assessor. ΔJ_{flip}: model-balanced excess informedness over the flipped single assessor at matched leniency. Mean-only selection uses only the mean acceptance rates on correct and unsuccessful proposals; Beta selection uses the method-of-moments fit. The best observed k is selected on the held-out committee curve and serves only as an optimistic reference.

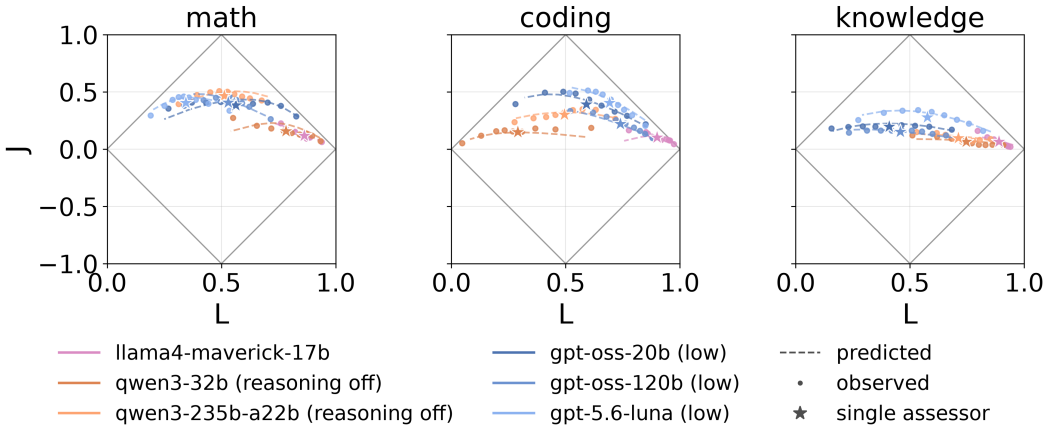

Figure 2: **Predicted and observed committee operating points.** Each panel shows the (L, J) operating points for mathematics, coding, and knowledge, respectively. Stars denote the single-assessor operating points, solid points the observed $(k, 9)$ committee operating points on held-out verdicts, and dashed curves the corresponding Beta-model predictions. Increasing the consensus threshold moves the committee toward lower leniency. The Beta approximation closely tracks the observed committee trajectories across models and tasks.

trajectory poorly: across committee operating points, its RMSE is 0.143 in J and 0.182 in L, compared with 0.0287 and 0.0125 for the Beta model. For each model–benchmark pair with a valid fit, the Beta model uses five assessment slots per proposal to fit two parameters separately for the correct and unsuccessful proposal classes, and closely predicts the held-out $N = 9$ committee trajectories across tasks (Fig. 2). Thus, committee behavior can be predicted from a compact characterization of the single assessor, providing a principled way to select consensus thresholds and predict the resulting operating points without exhaustively evaluating every committee rule.

Together, these results show that committee amplification depends strongly on the consensus threshold: majority voting leaves much of the available gain unused, while simple predictive models can select effective thresholds, and the Beta model accurately predicts the resulting operating points.

6 Related Work

Scalable oversight relies in part on the possibility that evaluating proposed solutions remains tractable when producing them does not (Amodei et al., 2016; Irving et al., 2018; Christiano et al., 2018;

Bowman et al., 2022; Wen et al., 2025; Engels et al., 2025). Closest to our work are studies that compare generation and self-evaluation directly. Bandyopadhyay (2026) find that evaluation outperforms generation on only one of four in-context QA benchmarks, comparing benchmark-level accuracies. Luo et al. (2023) find that critique accuracy is lowest on queries where models are least certain, and Pandit et al. (2025) compare per-problem solve and verification rates on frontier mathematics, finding verification generally easier. We instead condition directly on the model's own solve rate and measure discrimination separately from leniency. Collot et al. (2026) motivate balanced accuracy, a linear transformation of Youden's J, for prevalence-robust judge evaluation. We use J together with leniency L to characterize the assessor's operating point. The generation–verification gap of Song et al. (2025) addresses the complementary question of how much verification improves generation; our Section 3.2 relates it directly to J.

A second line of work concerns aggregating multiple assessor judgments, which we study in the form of k/N committees. Aggregating multiple language model judgments can improve evaluation and selection (Lifshitz et al., 2025; Zhang et al., 2025; Saad-Falcon et al., 2025), but the aggregation rule itself matters. Prior work considers richer aggregation settings, including combining heterogeneous weak verifiers with learned weights (Saad-Falcon et al., 2025). We study the more restricted case of a homogeneous committee, which allows us to predict the committee's (L, J) operating curve and select consensus thresholds from single-assessor measurements. Classical committee theory provides the corresponding foundation for threshold-based aggregation (Sah & Stiglitz, 1988; Ben-Yashar & Nitzan, 1997).

7 Conclusion

We asked whether a language model that rarely solves a problem can nevertheless assess proposed solutions to it, separating assessor informedness J from leniency L. On hard mathematics and coding problems, self-verification remains positively informed. In contrast, on hard knowledge tasks informedness was indistinguishable from chance at the model-balanced level. Removing the multiple-choice options from the assessor's view shifted leniency but not informedness. Committees of repeated assessor samples increased informedness, but majority voting often left these gains unrealized, and the committee's operating curve could be predicted from the first two moments of the single assessor's acceptance propensity.

These findings indicate that the assumption that verification is easier than generation cannot be taken for granted: whether it holds depends on the task, on the assessor's operating behavior, and on how judgments are aggregated. Separating informedness from leniency was essential to finding this. As we have shown, an assessor that is simply more lenient or more strict can appear more or less accurate without discriminating any better, and methods that move leniency, such as verdict flipping, cannot increase informedness.

Several directions follow naturally from this work. Extending the analysis to heterogeneous assessors, such as committees drawing on different models, would require modeling joint rather than treating each model separately. Eliciting knowledge proposals in open-ended form would fully separate task family from response format. Finally, relaxing the assumption that repeated verdicts are independent would allow committee predictions to account for correlated errors. More broadly, we hope that separating informedness from leniency can serve as a basis for future work on when, and how far, oversight can extend to problems beyond an assessor's own capability to solve.

8 AI Usage Acknowledgement

Generative AI tools were used to provide feedback on research methodology and experimental design, assist with code development and review, check mathematical derivations and theoretical exposition, support interpretation and presentation of results, and assist with manuscript drafting, editing, organization, and literature and reference checking. All AI-assisted code, text, analyses, mathematical claims, citations, and experimental results were reviewed and verified by the authors. The authors take responsibility for the final content of the work and all associated artifacts.

9 Reproducibility Statement

We include detailed documentation of the experimental procedure and analysis throughout the paper and supplementary material. Appendix B specifies the benchmarks, model and inference settings, proposer and assessor prompts, sampling and filtering procedures, grading, response-format control, and committee data collection; Appendices C and D describe the committee evaluation procedure and Beta-based prediction method, respectively. The supplementary material contains the experimental pipeline and executed analysis notebooks used to produce the reported figures, tables, and summary statistics.

References

Dario Amodei, Chris Olah, Jacob Steinhardt, Paul Christiano, John Schulman, and Dan Mané. Concrete problems in ai safety. *arXiv preprint arXiv:1606.06565*, 2016.

Sambaran Bandyopadhyay. Can llms judge better than they generate? evaluating task asymmetry, mechanistic interpretability and transferability for in-context qa. *arXiv preprint arXiv:2606.28050*, 2026.

Ruth C Ben-Yashar and Shmuel I Nitzan. The optimal decision rule for fixed-size committees in dichotomous choice situations: the general result. *International Economic Review*, pp. 175–186, 1997.

Samuel R Bowman, Jeeyoon Hyun, Ethan Perez, Edwin Chen, Craig Pettit, Scott Heiner, Kamilė Lukošiūtė, Amanda Askell, Andy Jones, Anna Chen, et al. Measuring progress on scalable oversight for large language models. *arXiv preprint arXiv:2211.03540*, 2022.

Paul Christiano, Buck Shlegeris, and Dario Amodei. Supervising strong learners by amplifying weak experts. *arXiv preprint arXiv:1810.08575*, 2018.

Stephane Collot, Colin Fraser, Justin Zhao, William F Shen, Timon Willi, and Ilias Leontiadis. Balanced accuracy: The right metric for evaluating llm judges-explained through youden's j statistic. In *Proceedings of the 19th Conference of the European Chapter of the Association for Computational Linguistics (Volume 5: Industry Track)*, pp. 927–936, 2026.

Jasper Dekoninck, Nikola Jovanović, Tim Gehrunger, Kári Rögnvaldsson, Ivo Petrov, Chenhao Sun, and Martin Vechev. Beyond benchmarks: Matharena as an evaluation platform for mathematics with llms. *arXiv preprint arXiv:2605.00674*, 2026.

Joshua Engels, David Baek, Subhash Kantamneni, and Max Tegmark. Scaling laws for scalable oversight. In D. Belgrave, C. Zhang, H. Lin, R. Pascanu, P. Koniusz, M. Ghassemi, and N. Chen (eds.), *Advances in Neural Information Processing Systems*, volume 38, Main Conference, pp. 97327–97366. Curran Associates, Inc., 2025. doi: 10.52202/085713-3254. URL https://proceedings.neurips.cc/paper_files/paper/2025/file/8c3caae2f725c8e2a55ecd600563d172-Paper-Conference.pdf.

Geoffrey Irving, Paul Christiano, and Dario Amodei. Ai safety via debate. *arXiv preprint arXiv:1805.00899*, 2018.

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

Dongwei Jiang, Jingyu Zhang, Orion Weller, Nathaniel Weir, Benjamin Van Durme, and Daniel Khashabi. Self-[in] correct: Llms struggle with discriminating self-generated responses. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 39, pp. 24266–24275, 2025.

Ryo Kamoi, Yusen Zhang, Nan Zhang, Jiawei Han, and Rui Zhang. When can llms actually correct their own mistakes? a critical survey of self-correction of llms. *Transactions of the Association for Computational Linguistics*, 12:1417–1440, 2024.

Shalev Lifshitz, Sheila A McIlraith, and Yilun Du. Multi-agent verification: Scaling test-time compute with multiple verifiers. *arXiv preprint arXiv:2502.20379*, 2025.

Liangchen Luo, Zi Lin, Yinxiao Liu, Lei Shu, Yun Zhu, Jingbo Shang, and Lei Meng. Critique ability of large language models. *arXiv preprint arXiv:2310.04815*, 2023.

Shrey Pandit, Austin Xu, Xuan-Phi Nguyen, Yifei Ming, Caiming Xiong, and Shafiq Joty. Hard2verify: A step-level verification benchmark for open-ended frontier math. *arXiv preprint arXiv:2510.13744*, 2025.

David M W Powers. Evaluation: from precision, recall and f-measure to roc, informedness, markedness and correlation. *Journal of Machine Learning Technologies*, 2(1):37–63, 2011. arXiv:2010.16061.

David Rein, Betty Li Hou, Asa Cooper Stickland, Jackson Petty, Richard Yuanzhe Pang, Julien Dirani, Julian Michael, and Samuel R. Bowman. GPQA: A graduate-level google-proof q&a benchmark. In *First Conference on Language Modeling*, 2024. URL https://openreview.net/forum?id=Ti67584b98.

Jon Saad-Falcon, E Kelly Buchanan, Mayee F Chen, Tzu-Heng Huang, Brendan McLaughlin, Tanvir Bhathal, Shang Zhu, Ben Athiwaratkun, Frederic Sala, Scott Linderman, et al. Shrinking the generation-verification gap with weak verifiers. *arXiv preprint arXiv:2506.18203*, 2025.

Raaj Kumar Sah and Joseph E. Stiglitz. Committees, hierarchies and polyarchies. *The Economic Journal*, 98(391):451–470, 1988. doi: 10.2307/2233372.

Lloyd Shapley and Bernard Grofman. Optimizing group judgmental accuracy in the presence of interdependencies. *Public Choice*, 43(3):329–343, 1984.

Yuda Song, Hanlin Zhang, Carson Eisenach, Sham Kakade, Dean Foster, and Udaya Ghai. Mind the gap: Examining the self-improvement capabilities of large language models. In *International Conference on Learning Representations*, volume 2025, pp. 39894–39931, 2025.

Yubo Wang, Xueguang Ma, Ge Zhang, Yuansheng Ni, Abhranil Chandra, Shiguang Guo, Weiming Ren, Aaran Arulraj, Xuan He, Ziyan Jiang, et al. Mmlu-pro: A more robust and challenging multi-task language understanding benchmark. *Advances in Neural Information Processing Systems*, 37: 95266–95290, 2024.

Xueru Wen, Jie Lou, Xinyu Lu, Junjie Yang, Yanjiang Liu, Yaojie Lu, Debing Zhang, and Xing Yu. Scalable oversight for superhuman ai via recursive self-critiquing. *arXiv preprint arXiv:2502.04675*, 2025.

W. J. Youden. Index for rating diagnostic tests. *Cancer*, 3(1):32–35, 1950. doi: https://doi.org/10.1002/1097-0142(1950)3:1⟨32::AID-CNCR2820030106⟩3.0.CO;2-3. URL https://acsjournals.onlinelibrary.wiley.com/doi/abs/10.1002/1097-0142%281950%293%3A1%3C32%3A%3AAID-CNCR2820030106%3E3.0.CO%3B2-3.

Lunjun Zhang, Arian Hosseini, Hritik Bansal, Seyed Mehran Kazemi, Aviral Kumar, and Rishabh Agarwal. Generative verifiers: Reward modeling as next-token prediction. In *International Conference on Learning Representations*, volume 2025, pp. 12476–12505, 2025.

A The Uncertainty of Informedness J

In this section we review and derive the relationship between the uncertainty of Youden's J statistics and the number of proposal samples as well as the number of verdict samples.

A.1 Sampling Design

For a fixed model $\mathcal{M}$ and request r, we draw n proposals from the proposer, and on each proposal V verdicts from the assessor. There are therefore n proposer LM runs and $n \times V$ assessor LM runs.

A single proposer attempt either solves the problem or not, so we model the outcome with a Bernoulli trial with success probability p. Over n independent attempts we denote the number of correct and incorrect proposals as n_+, n_-, giving an estimated accuracy $\hat{p} = n_+/n$.

For a fixed proposal, we model the verdicts as i.i.d. Bernoulli draws at the accept propensity q. If a of its V verdicts accept, its estimated accept propensity is

$$\hat{q}_i = \frac{a}{V}. \tag{20}$$

The TPR (p_{++}) and the FPR (p_{-+}) are estimated by averaging $\hat{q}_i$ across the proposals of each grade:

$$\hat{p}_{++} = \frac{1}{n_+} \sum_{i:G_r(y_i)=1} \hat{q}_i, \qquad \hat{p}_{-+} = \frac{1}{n_-} \sum_{i:G_r(y_i)=0} \hat{q}_i. \tag{21}$$

A.2 Confidence Intervals for J and L

Over a random proposal of a grade, the uncertainty in $\hat{q}_i$ has two sources: which proposal is drawn, and the verdicts drawn conditioned on the proposal. By the law of total variance, for the true-positive rate,

$$\mathrm{Var}(\hat{p}_{++}) = \underbrace{\frac{\mathrm{Var}_+(q)}{n_+}}_{\text{across proposals}} + \underbrace{\frac{\mathbb{E}_+[q(1-q)]}{V n_+}}_{\text{across verdicts}} \tag{22}$$

and likewise for the incorrect grade. The subscript plus denotes the distribution of q across correct proposals, and minus across incorrect ones. To have a better parameterization of the variance, we will work with the normalized versions

$$\rho_+ \equiv \frac{\mathrm{Var}_+(q)}{p_{++}(1-p_{++})}, \qquad \rho_- \equiv \frac{\mathrm{Var}_-(q)}{p_{-+}(1-p_{-+})}. \tag{23}$$

Substituting the definition of $\rho_\pm$ into the variance decomposition gives

$$\mathrm{Var}(\hat{p}_{++}) = \frac{p_{++}(1-p_{++})}{n_+ V} \left[1 + (V-1)\rho_+\right], \tag{24}$$

and likewise for $\hat{p}_{-+}$. The width W_J of the confidence interval for J takes the form $W_J = 2z\sqrt{\mathrm{Var}(J)}$, whereas $\mathrm{Var}(J) = \mathrm{Var}(\hat{p}_{++}) + \mathrm{Var}(\hat{p}_{-+})$. We would like to understand how W_J scales with the number of proposals that we sample n, the number of verdicts we sample each proposal V, as well as the difficulty of the request given by the accuracy p. To analyze the dependence of W_J on these factors let us suppose our assessor is a "symmetric-chance" one: $p_{++} = p_{-+} = \frac{1}{2}$, and that the normalized variance of the accept propensities q is the same across the correct and incorrect proposals and given by some $\rho = \rho_+ = \rho_-$. Then we find

$$W_J(\hat{p}; n, V) = z\sqrt{\frac{1 + (V-1)\rho}{nV\hat{p}(1-\hat{p})}}, \quad p_{++} = p_{-+} = \tfrac{1}{2}. \tag{25}$$

With n proposal samples and nV verdict samples, the number of proposals required to obtain a target width W_J is determined by

$$n \gtrsim \underbrace{\frac{4z^2}{W_J^2}}_{\text{Width}} \underbrace{\frac{1}{4\hat{p}(1-\hat{p})}}_{\text{Difficulty}} \underbrace{\frac{1 + (V-1)\rho}{V}}_{\text{Var. across Proposals}}. \tag{26}$$

We have factored Eq. (26) into three terms. The first term arises from one's requirement on the width, the second arises from the difficulty of the request, and the third arises from the variance of the accept propensities across the proposals. We note that the total number of LM calls required is $n(V+1)$, and this scales very unfavorably at extremely small p, making it experimentally difficult to determine J for very easy or very difficult questions.

B EXPERIMENTAL PROCEDURES

B.1 EXPERIMENTAL SETUP

We evaluate mathematical reasoning (AIME2026, BRUMO2025, SMT2025, and ApexShortList), knowledge (GPQA-Diamond and MMLU-Pro), and coding (LiveCodeBench). The math benchmarks contain 30, 30, 53, and 47 questions, respectively. We use all 198 GPQA-Diamond questions and a fixed subset of 300 MMLU-Pro questions spanning all 14 subject categories. LiveCodeBench questions are loaded from `test6.jsonl` in `livecodebench/code_generation_lite`. Knowledge questions are multiple-choice; math and coding questions are open-ended.

Models and inference settings. Each model uses the same inference settings for proposing and assessing, as listed in Table 3. Temperature is not explicitly set for GPT-5.6-Luna. The Qwen presets request low reasoning effort, which the serving configuration treats as reasoning disabled.

Model	Provider	Temperature	Reasoning	Token limit
Llama-4-Maverick-17B	Bedrock	0.6	–	8,000
Qwen3-32B	Bedrock	0.6	Disabled	16,000
Qwen3-235B-A22B	Bedrock	0.6	Disabled	16,000
GPT-OSS-20B	Bedrock	0.6	Low	16,000
GPT-OSS-120B	Bedrock	0.6	Low	16,000
GPT-5.6-Luna	OpenAI	Not set	Low	16,000

Table 3: Inference settings for both proposer and assessor calls. Token limits apply to generated output.

The corresponding API model identifiers are `us.meta.llama4-maverick-17b-instruct-v1:0`, `qwen.qwen3-32b-v1:0`, `qwen.qwen3-235b-a22b-2507-v1:0`, `openai.gpt-oss-20b-1:0`, `openai.gpt-oss-120b-1:0`, and `gpt-5.6-luna`.

Proposer prompts. Math requests use the problem statement as the user message and the following system message:

> Solve the following math problem step by step. Put your final answer within `\boxed{}`.

Knowledge requests use the question followed by options labeled `(A)`, `(B)`, etc., with the following system message:

> Reason step by step to answer a multiple-choice question. Read the question and the options carefully. The last line of your response should be of the following format: `\boxed{LETTER}` where LETTER is one of `{LETTERS}`.

The placeholder `{LETTERS}` is replaced with `ABCD` for GPQA-Diamond and `ABCDEFGHIJ` for MMLU-Pro.

Coding requests use the following system message:

> You are an expert Python programmer. You will be given a question (problem specification) and will generate a correct Python program that matches the specification and passes all tests.

The user message contains the problem statement under `### Question:`, formatting instructions and a Python code block under `### Format:`, and ends with `### Answer: (use the provided format with backticks)`. When starter code is available, the code block contains it and the formatting instruction is:

> You will use the following starter code to write the solution to the problem and enclose your code within delimiters.

Otherwise, the code block contains `# YOUR CODE HERE` and the formatting instruction is:

> Read the inputs from stdin solve the problem and write the answer to stdout (do not directly test on the sample inputs). Enclose your code within delimiters as follows. Ensure that when the python program runs, it reads the inputs, runs the algorithm and writes output to STDOUT.

Assessor prompts. In the primary condition, each assessment uses a fresh request containing the original proposer instructions, question, and full response content. Any explanation within the response content is included, but separately returned reasoning traces or summaries are excluded. The response-format control modifies the embedded request and final-answer representation as described in Appendix B.6, while preserving the remaining response content. The assessor receives no reference answer or grading result. Its system message is:

> Assess the assistant's response to the user's request against these 3 criteria:
>
> 1. Consistency: is it an appropriate response to the request it was given?
> 2. Logical Soundness: is it free of logical contradictions?
> 3. Factual Accuracy: is it free of factual errors?
>
> Do not assume the claims in the response are true; verify them yourself.
>
> Reply with a brief justification, then the verdict line, which must be exactly one of these tokens, on its own line, verbatim:
>
> ```
> [Verdict: PASS]
> [Verdict: FAIL]
> ```
>
> PASS only if all three criteria pass; FAIL if any one fails, or if you are unsure.

The user message follows this template:

```
The user's request:
instructions: {instructions}
request body: {request_body}

The assistant's response:
{proposal}
```

B.2 Initial proposal sampling and screening

For each model–question pair, we sample 20 independent proposals using the settings in Appendix B.1. Each proposal receives one of five outcomes:

- `error`: the API call fails after retries.
- `truncated`: generation reaches the output-token limit.
- `no_answer`: the response content is empty, or answer extraction or grading raises a handled exception.
- `correct`: the extracted answer passes the benchmark grader.
- `incorrect`: the extracted answer fails the benchmark grader.

For math, we extract the last `\boxed{}` expression and check symbolic equivalence to the reference answer using `math_verify`. For knowledge, we extract the last boxed answer, remove surrounding whitespace and an optional `\text{}` wrapper, and compare the option letter to the reference without regard to case. For coding, we extract the last fenced code block and require it to pass all public and private tests.

We exclude infrastructure errors from the screening accuracy denominator. Incorrect, truncated, and no-answer proposals count as unsuccessful. A model–question pair proceeds to additional sampling only if it has at least one correct and one unsuccessful non-error proposal.

B.3 Additional proposal sampling and aggregation

For each retained model–question pair, we sample and grade 80 additional proposals and combine them with the initial 20. We retain pairs with 100 non-error proposals, including at least one correct and one unsuccessful proposal. Proposer accuracy $\hat{p}$ is the fraction of these 100 proposals graded correct, computed before verdict filtering.

B.4 Assessor sampling and verification-rate estimation

For each aggregated proposal, we sample two independent assessments using the same model and the assessor prompt in Appendix B.1. The last occurrence of `[Verdict: PASS]` or `[Verdict: FAIL]` in the response determines acceptance or rejection. Responses without either token are parse errors; missing response bodies are errors. Both are excluded from rate estimation. Truncated assessments remain usable if they contain a verdict token.

We retain proposals with at least one valid verdict and model–question pairs with both correct and unsuccessful proposals remaining. For proposal i, we estimate its acceptance probability from its acceptance count a_i and valid-verdict count V_i:

$$\hat{q}_i = \frac{a_i}{V_i}.$$

Let S_+ and S_- denote the retained correct and unsuccessful proposals for a model–question pair. We estimate

$$\widehat{\mathrm{TPR}} = \frac{1}{|S_+|} \sum_{i \in S_+} \hat{q}_i, \qquad \widehat{\mathrm{FPR}} = \frac{1}{|S_-|} \sum_{i \in S_-} \hat{q}_i. \tag{27}$$

Each proposal receives equal weight within its class, regardless of its valid-verdict count. We then compute

$$\widehat{\mathrm{TNR}} = 1 - \widehat{\mathrm{FPR}}, \qquad \hat{J} = \widehat{\mathrm{TPR}} - \widehat{\mathrm{FPR}}, \qquad \hat{L} = \frac{\widehat{\mathrm{TPR}} + \widehat{\mathrm{FPR}}}{2}. \tag{28}$$

After filtering, the retained question sets differ across models. Depending on the model, we retain 53–99 mathematics questions, 46–85 coding questions, and 137–294 knowledge questions.

B.5 Committee sampling and evaluation

For each retained model–question pair, we select one in every ten proposals for committee analysis by retaining proposals whose proposal ID is divisible by 10. This gives a deterministic 10% subset that is fixed across reruns. Before collecting additional verdicts, we retain only model–question pairs for which this subset contains both correct and unsuccessful proposals. Each retained proposal then receives 16 additional assessor verdicts, giving 18 verdict slots in total when combined with the two verdicts collected during the primary assessment stage.

For the primary $N = 9$ analysis, the first five assessment slots for each proposal are allocated to fitting and the remaining thirteen to held-out committee evaluation. After invalid assessments are removed, we require at least two valid fitting verdicts per proposal to estimate the second moment and at least nine valid held-out verdicts to evaluate an $N = 9$ committee. We evaluate predicted and observed committee performance on the intersection of proposals satisfying both requirements, and retain questions with both correct and unsuccessful proposals on this common support. For task-level committee analyses, we aggregate benchmark-level curves within each model using retained-question counts as weights, so each retained question receives equal weight within a model–task pair.

B.6 Response-format control

For each previously sampled knowledge proposal with an unambiguous final boxed option label c, the primary condition presents the assessor with the original proposer instructions, the question and answer choices, and the full response content. The control condition removes the explicit answer choices from the request and replaces only the final boxed option label with the text of the selected option. All other response content is preserved verbatim, including any explanation or references to other options. The replacement uses the proposer's selected option, not the reference answer.

We also modify the proposer instructions embedded in the assessor's request to remove references to multiple-choice options and request a boxed answer rather than a boxed option label. The assessor's own system instructions and verdict format remain unchanged. Separately returned reasoning traces or summaries are excluded in both conditions.

No proposals are regenerated or regraded. We retain the original proposal identifiers and correctness labels, and compute proposer accuracy from the original proposal samples before control-specific filtering. We reuse the original assessments for the primary condition and sample two new verdicts per proposal for the control condition, using the same model, inference settings, and verdict parser.

The comparison includes only proposals whose final boxed label maps unambiguously to an available option and that have at least one valid verdict in each condition. We retain model–question pairs with both correct and unsuccessful proposals on this common support. For each condition, we estimate TPR, TNR, J, and L using the per-proposal averaging procedure in Appendix B.

This intervention changes the explicit choice list and final-answer representation while retaining the explanation available to the assessor. It does not produce fully open-ended proposals: the preserved content may still discuss distractors or refer to option labels, and the original proposals were elicited using multiple-choice prompts.

C Measuring Committee Performance

This appendix outlines the method used in this paper to calculate the observed (k, N) assessor committee accept propensity, given $V \geq N$ independently sampled verdicts.

Recall that a (k, N) committee accepts when at least k of its N verdicts signal accept. Empirically, if V verdicts are independently sampled from the same verifier, and u of them turn out to be "accept", our estimate for the probability that within an N-member committee, exactly a of the members will accept is

$$\hat{\Pr}[a] = \binom{u}{a}\binom{V-u}{N-a}\Big/\binom{V}{N}. \tag{29}$$

The numerator counts the subsets of N verdicts containing exactly a accepts and $N - a$ rejects. The denominator counts all subsets of N verdicts drawn from the V available verdicts.

Thus our estimate for the accept propensity $\hat{q}_{(k,N)}$ of the committee is then

$$\hat{q}_{(k,N)} = \sum_{a=k}^{N} \hat{\Pr}[a]. \tag{30}$$

From here one can go on to compute the relevant rates such as $\mathrm{TPR}_{(k,N)}, \mathrm{TNR}_{(k,N)}$.

D Review of Beta Distribution

For completeness, this appendix includes information on the Beta distribution which we used to approximate the distribution of the accept propensities q across the correct and incorrect proposals and thus to obtain the higher moments $m_{\pm}^{(j)} = \langle q^j \rangle_{\pm}$.

The probability density function of the Beta distribution is given by

$$f(x; \alpha, \beta) = \frac{x^{\alpha-1}(1-x)^{\beta-1}}{\mathrm{B}(\alpha, \beta)}, \tag{31}$$

under which the moments of every order are in closed form,

$$m^{(j)} \equiv \langle x^j \rangle = \prod_{u=0}^{j-1} \frac{\alpha + u}{\alpha + \beta + u}. \tag{32}$$

To predict committee behavior, we need the distribution of the assessor's proposal-level acceptance propensity q. We approximate this distribution separately for correct and unsuccessful proposals with

a Beta distribution. We fit the two Beta parameters in each class by matching the first two moments of q:

$$m_{\pm}^{(1)} = \mathbb{E}_{\pm}[q], \qquad m_{\pm}^{(2)} = \mathbb{E}_{\pm}[q^2], \tag{33}$$

where $+$ and $-$ denote the correct and unsuccessful proposal classes, respectively.

The acceptance propensity of an individual proposal is not observed directly. For proposal i, let q_i denote its probability of receiving an accept verdict. Let V_i be the number of valid assessor verdicts sampled for that proposal and let a_i be the number of those verdicts that accept. Under our assumption that repeated verdicts are conditionally independent,

$$a_i \mid q_i \sim \mathrm{Binomial}(V_i, q_i). \tag{34}$$

The observed acceptance fraction provides an unbiased estimate of q_i:

$$\hat{q}_i = \frac{a_i}{V_i}, \qquad \mathbb{E}\left[\hat{q}_i \mid q_i\right] = q_i. \tag{35}$$

Averaging this quantity across proposals therefore estimates the first moment $m_{\pm}^{(1)}$.

Estimating the second moment requires more care. Simply squaring the observed acceptance fraction does not give an unbiased estimate of q_i^2, because $\hat{q}_i$ also contains noise from the finite number of verdict samples:

$$\mathbb{E}\left[\left(\frac{a_i}{V_i}\right)^2 \middle| q_i\right] = q_i^2 + \frac{q_i(1-q_i)}{V_i}. \tag{36}$$

Thus, using $\hat{q}_i^2$ would attribute some within-proposal verdict-sampling noise to variation in acceptance propensity across proposals.

Instead, we estimate q_i^2 using

$$\widehat{q_i^2} = \frac{a_i(a_i-1)}{V_i(V_i-1)}. \tag{37}$$

We require at least two valid fitting verdicts for each proposal entering the second-moment estimate. This quantity has a simple interpretation. The numerator counts ordered pairs of accept verdicts among the V_i sampled verdicts, while the denominator counts all ordered pairs of distinct verdicts. Conditional on q_i, any two distinct verdicts both accept with probability q_i^2. Consequently,

$$\mathbb{E}\left[\frac{a_i(a_i-1)}{V_i(V_i-1)} \middle| q_i\right] = q_i^2. \tag{38}$$

Let $\mathcal{R}$ denote the set of retained questions for a model–benchmark pair, and let $S_{\pm,r}$ denote the correct and unsuccessful proposals, respectively, for question $r \in \mathcal{R}$. We give each question equal weight and, within each question and correctness class, give each proposal equal weight. We therefore estimate the first two moments in each class as

$$\hat{m}_{\pm}^{(1)} = \frac{1}{|\mathcal{R}|}\sum_{r\in\mathcal{R}}\frac{1}{|S_{\pm,r}|}\sum_{i\in S_{\pm,r}}\frac{a_i}{V_i}, \qquad \hat{m}_{\pm}^{(2)} = \frac{1}{|\mathcal{R}|}\sum_{r\in\mathcal{R}}\frac{1}{|S_{\pm,r}|}\sum_{i\in S_{\pm,r}}\frac{a_i(a_i-1)}{V_i(V_i-1)}. \tag{39}$$

We then fit a Beta distribution separately to each class by matching these two estimated moments. For either class, the method-of-moments estimates are

$$\hat{\alpha} = \hat{m}^{(1)}\left(\frac{\hat{m}^{(1)}\left(1-\hat{m}^{(1)}\right)}{\hat{m}^{(2)} - \left(\hat{m}^{(1)}\right)^2} - 1\right), \qquad \hat{\beta} = \left(1-\hat{m}^{(1)}\right)\left(\frac{\hat{m}^{(1)}\left(1-\hat{m}^{(1)}\right)}{\hat{m}^{(2)} - \left(\hat{m}^{(1)}\right)^2} - 1\right). \tag{40}$$

Once $\hat{\alpha}$ and $\hat{\beta}$ have been obtained, Eq. (32) gives estimates of all higher moments required by Eq. (16), and hence the predicted operating point of any (k, N) committee.

For the primary $N = 9$ analysis, the first five assessment slots for each proposal are allocated to fitting and the remaining thirteen to held-out committee evaluation. After invalid assessments are removed, we require at least two valid fitting verdicts per proposal to estimate the second moment above and at least nine valid held-out verdicts to evaluate an $N = 9$ committee. Prediction and observation are evaluated on the intersection of proposals satisfying both requirements.

Across the 42 model–benchmark pairs in the primary analysis, 40 admit valid method-of-moments Beta fits. The two exceptions occur for the correct-proposal class on AIME2026: the Llama-4-Maverick fit lies on the boundary with $\hat{m}^{(1)} = 1$ and zero estimated variance, while the Qwen3-32B fit yields a slightly negative estimate of the variance in acceptance propensity across proposals, which is a finite-sample artifact. These two model–benchmark pairs are omitted symmetrically from both the predicted and observed committee analyses, so all committee methods are compared on matched support.