# SCORING SIGNAL SELECTION FOR TEST-TIME SCALING

**Anonymous authors**
Paper under double-blind review

## ABSTRACT

Test-time scaling (TTS) improves reasoning by sampling multiple candidate solutions and selecting among them with a scoring function. Scoring signals include internal signals (e.g., model confidence), which add negligible cost beyond generation, and external signals (e.g., process reward models, PRMs), which provide additional evaluation at extra cost. Prior work largely compares or combines such signals under a fixed scoring rule across instances. We ask whether the scoring signal can be selected per instance after candidate generation but before invoking external scoring. To our knowledge, this post-generation selection problem has not been explicitly studied. In this work, we systematically study post-generation scoring signal selection. We empirically show that internal and external signals exhibit low rank correlation when evaluating the same candidate sets, while each uniquely succeeds on a non-trivial subset of instances. This complementarity creates a meaningful opportunity for per-instance signal selection. We then evaluate practical selection strategies, ranging from simple candidate disagreement statistics to learned routing models. We find that inexpensive disagreement statistics such as answer entropy capture useful coarse routing signals, while the learned selectors we evaluate do not consistently improve upon them across the full accuracy–call-rate frontier, even with labeled supervision and richer representations. These findings establish scoring signal selection as a practically relevant and largely unexplored axis for improving TTS efficiency, while showing that fine-grained per-instance selection remains challenging beyond coarse disagreement signals.

## 1 INTRODUCTION

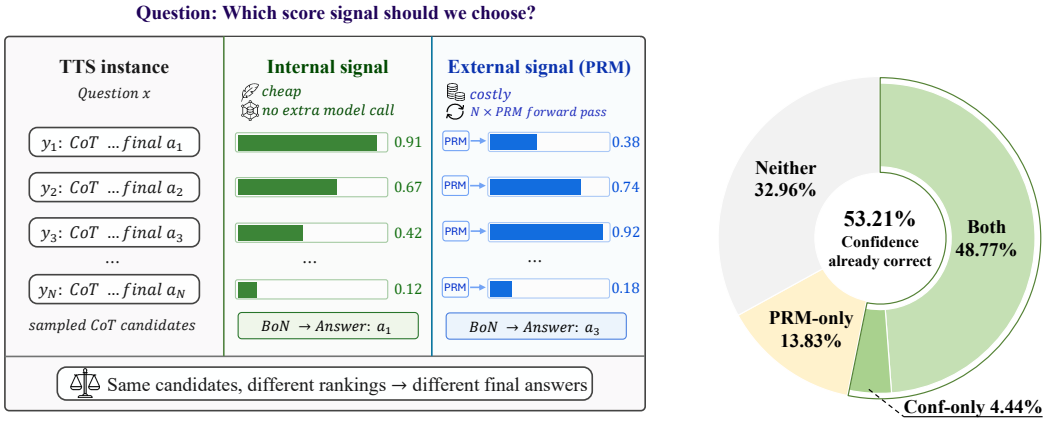
Recent advances in large language models (LLMs) show that reasoning performance can improve not only through training-time scaling, but also by investing more computation at inference time. Test-time scaling (TTS) exploits this by generating multiple candidate solutions and aggregating them rather than relying on a single response Wei et al. (2022); Wang et al. (2023b); Snell et al. (2025); Brown et al. (2020); Yao et al. (2023). Although majority voting alone can yield meaningful gains, aggregation methods that incorporate a scoring function tend to achieve substantially greater improvements Lightman et al. (2023); He et al. (2024); Snell et al. (2025). Scoring approaches broadly include internal signals (e.g., model confidence), derived directly from the base model at negligible cost Guo et al. (2017); Kadavath et al. (2022); Kang et al. (2025), and external signals (e.g., PRMs), which provide richer evaluation via auxiliary models at additional computational cost Lightman et al. (2023); Cobbe et al. (2021). Despite this diversity, existing work typically applies a single scoring signal uniformly across instances.

In this work, we study scoring signal selection. Given a question and a fixed set of sampled candidates, should we rank them using an internal confidence signal or invoke an external PRM (Figure 1(a))? This decision happens after candidate generation, so it only changes how the same candidates are ranked. Internal scoring adds negligible computation beyond generation, whereas a PRM requires an additional forward pass over every candidate. Selectively invoking the PRM could therefore reduce scoring cost while retaining its benefit on instances where it improves the final selection.

The challenge is determining when the external signal is actually useful. Internal confidence does not guarantee correctness, while the PRM can either correct an internal-scoring error or overturn a correct internal selection. The relevant outcome therefore depends on both scorers and the realized



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.




**(a)** Internal vs. external signal selection

**(b)** Signal outcome breakdown

Figure 1: Per-instance scoring signal selection in best-of-$N$ TTS. **(a)** The same $N$ candidates receive different rankings under the internal signal (confidence) and the external signal (PRM), producing different final answers. **(b)** On the held-out average of MMLU-Pro-Eval and MATH500, the internal signal already selects a correct answer for 53.2% of questions (48.8% both-correct and 4.4% internal-only), while the PRM uniquely rescues 13.8%. This motivates selectively invoking the PRM rather than applying it uniformly to all questions. Detailed results are in Appendix A.1.

candidate set. Scoring signal selection is thus not simply a question of which signal performs better on average, but of when external scoring is worth invoking for a particular instance.

Our empirical study yields three main findings. First, the two signals are complementary at the instance level. The internal signal already selects the correct answer on a slight majority of held-out questions, while the PRM uniquely succeeds on a smaller but important subset (Figure 1(b)). Always invoking the PRM, therefore, wastes computation on many instances where the internal signal already succeeds, while always relying on the internal signal misses recoverable PRM-only cases. An oracle that chooses the better signal per instance would therefore outperform either fixed scorer while calling the PRM on only a fraction of questions. The two signals also draw on different information sources: internal scores reflect the generator's own distributional confidence, whereas PRMs are produced by a separately trained evaluator.

Second, simple candidate disagreement statistics provide a useful but coarse routing signal. Normalized answer entropy roughly tracks when external scoring becomes more useful. Other inexpensive disagreement statistics, including majority margin and distinct-answer count, perform similarly, suggesting that the effect is not specific to entropy.

Third, the learned selectors we evaluate do not consistently improve over these disagreement statistics across the full accuracy–call-rate frontier. We consider both MLP classifiers over question and response representations and DeBERTa encoders adapted from prior LLM-routing work. This pattern persists across input features and when we replace the PRM-only target with a utility-aware objective (Appendix A.8.3). These results suggest that the evaluated learned models do not reliably recover finer-grained routing information beyond coarse candidate disagreement. We additionally test how the disagreement signal behaves across sampling budgets, internal signals, PRM aggregation rules, and external scorers.

Our contributions are as follows.

- We formulate scoring signal selection as a post-generation TTS problem, selecting the scoring signal per instance after generating a candidate set and before invoking the external scorer.
- We empirically show that internal and external signals induce different candidate rankings and succeed on different subsets of instances. Across the held-out benchmarks, the internal signal already selects a correct answer for 53.2% of questions, while PRM-only rescue accounts for 13.8%, motivating selective rather than uniform use of the external scorer.
- We evaluate training-free disagreement statistics and learned selectors across the full accuracy–call-rate frontier. Simple disagreement statistics such as answer entropy capture useful coarse routing structure, while richer representations and a utility-aware objective do not consistently

outperform these signals. We further test robustness across sampling budgets, internal signals, PRM aggregation rules, and external scorers.

## 2 Related Work

Test-time scaling (TTS) improves reasoning by leveraging additional computation during inference Wei et al. (2022); Wang et al. (2023b); Snell et al. (2025); Yao et al. (2023). A central component is the scoring function that evaluates candidate solutions. Previous work has focused on improving individual scoring models or aggregation strategies. Our work instead studies per-instance scoring signal selection after candidate generation.

**Internal Scoring Signals** assess candidate quality using information derived directly from the model's output distribution and require no additional inference beyond generation. Various approaches have been proposed, including token-level probabilities Guo et al. (2017); Kadavath et al. (2022), semantic entropy Kuhn et al. (2023), and verbalized confidence Tian et al. (2023); Xiong et al. (2024). Among these, Self-Certainty Kang et al. (2025) defines confidence as the average KL divergence between the predicted token distribution and a uniform distribution, making it mode-seeking and less susceptible to length bias. Despite their efficiency, internal signals reflect the model's own distributional preferences, and their calibration can degrade on difficult or out-of-distribution problems Guo et al. (2017); Xiong et al. (2024).

**External Scoring Signals** use separately trained auxiliary models to evaluate candidates. Process reward models (PRMs) provide step-level feedback on intermediate reasoning Lightman et al. (2023); Uesato et al. (2022); Wang et al. (2023a); Setlur et al. (2025) and outperform outcome supervision on challenging mathematical benchmarks Lightman et al. (2023), though recent work has questioned whether this advantage holds outside math-adjacent settings Lee et al. (2025a). We use PRMs as our representative external signal, specifically VersaPRM Zeng et al. (2025) on MMLU-Pro-CoT-Eval and Qwen2.5-Math-PRM-7B Zhang et al. (2025) on MATH.

**Signal Combination and LLM Routing.** Signal combination is a natural alternative to signal selection. Recent work shows that the optimal weighting of LLM confidence and PRM scores varies substantially across LLM–PRM pairs, making fixed aggregation strategies generally suboptimal Kuang et al. (2025). However, combination does not reduce the cost of external scoring, and combining signals with different ranges and semantics can require additional calibration.

Our work is also related to adaptive routing in LLM inference. Model-routing methods choose among models of different cost before generation Ong et al. (2025); Ding et al. (2025; 2024); Lee et al. (2025b); Chen et al. (2023), while adaptive-sampling methods vary the number of generated candidates or decide when to stop sampling Aggarwal et al. (2023); Li et al. (2024); Wang et al. (2025). Scoring signal selection differs from both. The candidate set is fixed, and the decision only determines how the existing candidates are ranked. Our DeBERTa selector adapts the question-routing paradigm of BEST-Route Ding et al. (2025) to this setting by replacing model selection with a binary choice between internal and external scoring. The supervision target also differs from conventional query routing. PRM-only correction depends jointly on the realized candidate set and the rankings induced by the two scorers, because the external signal must succeed on an instance where the internal signal fails. In our experiments, the evaluated routing models do not consistently recover useful information beyond coarse candidate disagreement statistics, although this does not rule out gains from other model architectures, representations, or training data. To our knowledge, we are the first to formulate post-generation scoring signal selection over a fixed candidate set as an explicit decision problem in TTS.

## 3 Scoring Signal Selection

### 3.1 Problem Setup

We operate in the test-time scaling (TTS) setting, where a base language model generates a set of $N$ candidate solutions for a given input question $x \in \mathcal{X}$, with $\mathcal{X}$ denoting the space of input questions:

$$\mathcal{Y} = \{y_1, \ldots, y_N\}, \tag{1}$$

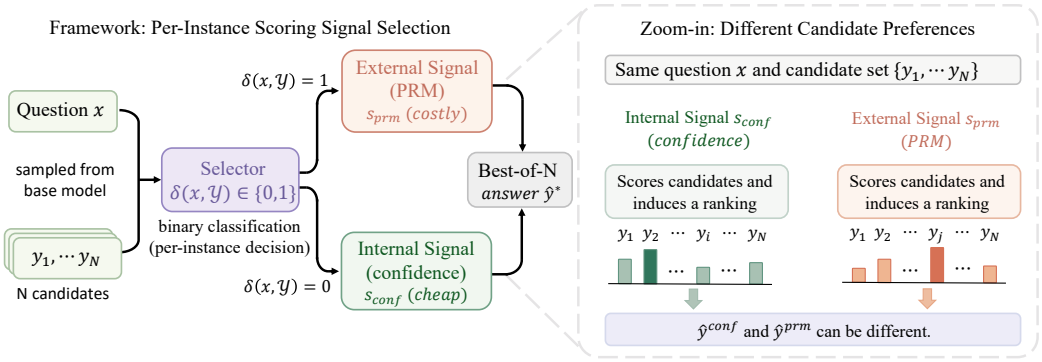



Figure 2: Overview of the per-instance scoring signal selection framework. Given $N$ candidate solutions sampled from the base model, a selector $\delta(x, \mathcal{Y})$ decides whether to use the internal signal $s_{\text{conf}}$ or the external signal $s_{\text{prm}}$.

each representing a chain-of-thought reasoning trajectory. From each candidate $y_i$, we extract a final answer $\text{ans}(y_i)$. When extraction fails, $\text{ans}(y_i)$ is left undefined. We denote by $\mathcal{A}$ the set of distinct answers successfully extracted from $\mathcal{Y}$.

A scoring function $s\colon \mathcal{Y} \to \mathbb{R}$ assigns a scalar to each candidate, and the final prediction is obtained via best-of-$N$ (BoN) selection:

$$\hat{y} = \arg\max_{y_i \in \mathcal{Y}} \; s(y_i). \tag{2}$$

We consider two types of scoring signals. The internal signal $s_{\text{conf}}$ is Self-Certainty Kang et al. (2025), derived directly from the base model's generation pass and measuring the divergence of the predicted token distribution from a uniform distribution.

The external signal $s_{\text{prm}}$ comes from a PRM Zeng et al. (2025); Zhang et al. (2025), which produces step-level scores that we aggregate using the minimum operator.

### 3.2 Per-Instance Signal Selection

Rather than using one signal globally, we study a per-instance selector that decides, for each input question $x \in \mathcal{X}$ and candidate set $\mathcal{Y}$, which signal to use. Formally, we define a binary selector $\delta(x, \mathcal{Y}) \in \{0, 1\}$ that determines which scoring signal is used for a given question and its candidates:

$$s^* = \begin{cases} s_{\text{prm}}, & \delta(x, \mathcal{Y}) = 1, \\ s_{\text{conf}}, & \delta(x, \mathcal{Y}) = 0, \end{cases} \tag{3}$$

the final answer is selected by BoN (Eq. 2) using $s^*$. Since routing occurs before external scoring, decision $\delta(x, \mathcal{Y}) = 0$ avoids the PRM cost entirely.

We evaluate selectors using **TTS accuracy**, the final BoN accuracy under the selected signal, and **Call rate**, the fraction of inputs for which the external signal is called. These jointly characterize the performance–efficiency tradeoff. Figure 2 illustrates an overview of the framework.

### 3.3 Design of Binary Selectors

We consider training-free selectors based on candidate-set statistics and learned selectors trained on labeled data.

**Training-free selectors.** The most direct training-free selector reuses the internal signal itself and routes to the PRM when the average self-certainty score across valid candidates is low. Formally, let $\bar{s}_{\text{conf}} = \frac{1}{N_{\text{ans}}} \sum_{y_i:\, \text{ans}(y_i)\ \text{defined}} s_{\text{conf}}(y_i)$. The conf heuristic is

$$\delta(x, \mathcal{Y}) = \mathbf{1}\big[\bar{s}_{\text{conf}} < \tau\big]. \tag{4}$$

Intuitively, a low average internal score suggests the generator lacks confidence, making external verification more likely to help. This requires no additional inference beyond generation.

We also consider answer entropy as a training-free measure of disagreement within the sampled candidate set. We evaluate whether this disagreement provides information about when external scoring is useful. Section 4.3 later characterizes empirically why this statistic is informative for routing. Defined as the entropy of the empirical distribution over extracted answers, it requires only a single pass over the candidate set and adds no inference cost beyond generation.

Let $N_{\text{ans}} = |\{y_i \in \mathcal{Y} : \text{ans}(y_i) \text{ is defined}\}|$ denote the number of candidates with valid answers. If $N_{\text{ans}} = 0$, we set $\hat{H}(\mathcal{Y}) = 0$. Otherwise, let

$$p(a) = \frac{|\{y_i \in \mathcal{Y} : \text{ans}(y_i) = a\}|}{N_{\text{ans}}} \tag{5}$$

be the empirical probability of answer $a \in \mathcal{A}$, where $\mathcal{A}$ is the set of distinct extracted answers. We define the normalized answer entropy as

$$\hat{H}(\mathcal{Y}) = \frac{-\sum_{a \in \mathcal{A}} p(a) \log p(a)}{\log N_{\text{ans}}}, \tag{6}$$

where dividing by $\log N_{\text{ans}}$ maps $\hat{H}(\mathcal{Y})$ into $[0, 1]$ regardless of the number of distinct answers, making it directly comparable across instances. $\hat{H}(\mathcal{Y})$ is a post-generation candidate-set statistic. It is computed from the sampled responses $\mathcal{Y}$ themselves, not from the question $x$ alone.[1] The simplest selector based on $\hat{H}(\mathcal{Y})$ is a threshold rule,

$$\delta(x, \mathcal{Y}) = \mathbf{1}\big[\hat{H}(\mathcal{Y}) > \tau\big], \tag{7}$$

where $\tau \in [0, 1]$ is a tunable threshold. The range follows from the normalization in Eq. 6. This selector requires no training or additional model inference. We sweep $\tau$ over its full range to construct accuracy–call-rate frontiers and use $\tau = 0.5$ for fixed-operating-point summaries.

**Learned selectors.** We also evaluate learned selectors that use richer question- and response-level representations. We evaluate MLP classifiers over response- and question-level hidden-state representations, optionally concatenated with answer entropy $\hat{H}(\mathcal{Y})$. We also evaluate encoder-based classifiers that adapt the question-routing paradigm Ding et al. (2025); Ong et al. (2025) to signal selection, testing whether the question-level representations used in model routing transfer to this setting. Both are trained with a binary cross-entropy loss, concrete feature definitions and training configuration are given in Section 4.1. Unlike the training-free heuristic, they require labeled training data and an additional inference pass, though the cost remains modest relative to LLM generation or PRM invocation.

## 4 Experiments

We present an empirical study of scoring signal selection under the framework in Section 3. Our goal is to understand when internal and external signals differ in their behavior, whether this complementarity can be exploited in practice, and how different selection strategies compare across settings. Appendix A.1 documents that the two signals are complementary enough to motivate per-instance selection. The main body then compares practical selection strategies (Section 4.2), analyzes the structure captured by disagreement-based selection (Section 4.3), and checks whether the resulting routing value is robust across external scorers (Section 4.4).

### 4.1 Experimental Setup

**Base model and candidate generation.** In the main experiments, we use Meta-Llama-3.1-8B-Instruct Dubey et al. (2024) as the base language model. For each input question, we draw $N = 32$ candidate solutions via chain-of-thought prompting with temperature sampling at $T = 0.8$. All main comparisons use the same pre-generated candidate pools to control for candidate-set variation.

**Benchmarks.** We evaluate on two reasoning benchmarks, pairing each with an external reward model. For MMLU-Pro Wang et al. (2024), a multiple-choice benchmark spanning 14 academic

[1] When $|\mathcal{A}| \leq 1$, indicating that all candidates agree or no valid answer is extracted, we set $\hat{H}(\mathcal{Y}) = 0$.

domains, we use VersaPRM Zeng et al. (2025) as the external scorer. Selector training uses the CoT-annotated dataset released by the VersaPRM authors[2], containing 5,750 questions split into training and validation sets stratified by positive label rate and entropy distribution. Evaluation uses the held-out VersaPRM evaluation set.[3] For the mathematical reasoning setting, we use MATH Hendrycks et al. (2021) for selector training and MATH500 Lightman et al. (2023) for evaluation, with Qwen2.5-Math-PRM-7B Zhang et al. (2025) as the external scorer. We sample 1,250 problems from the MATH training set and apply the same stratified train–validation split.

**Scoring signals and metrics.** The internal signal is Self-Certainty Kang et al. (2025), a training-free measure based on the KL divergence between the model's token-level distribution and uniform. The external signal is a PRM with step-level scores aggregated by the minimum operator. We report TTS accuracy, PRM call rate (CR), and **tradeoff** $=$ (TTS acc $-$ internal acc)$/$CR, measuring accuracy gain over the internal-only baseline per unit PRM call rate. For full-frontier comparisons, we report Pareto-AUC. We also use excess AUC to compare external scorers of different strength, defined as the selector's Pareto-AUC minus that of call-rate-matched random routing. Details are provided in Appendix A.4. Because generation cost is shared across methods, we use PRM call rate as the primary cost proxy, while selector inference is comparatively cheap relative to PRM scoring. [4]

**Selector implementation.** For learned selectors, we test whether representation-based routing features can improve over the entropy heuristic, following adaptive LLM routing work that uses input representations to predict question difficulty or select among models Chen et al. (2023); Ong et al. (2025); Ding et al. (2025); Lee et al. (2025b).

MLP classifiers use three feature families: Q-Rep, the projected last-token hidden state of the question. R-Rep, an aggregate representation of the sampled responses (each candidate's last-token hidden state projected to $\mathbb{R}^{64}$ and mean-pooled across the $N$ candidates). And Std, the per-dimension standard deviation of these projected response representations, captures dispersion among sampled reasoning trajectories. We optionally concatenate these with normalized answer entropy $\hat{H}(Y)$. Encoder-based classifiers use DeBERTa-v3-small He et al. (2021) with a special token <|source|> prepended to the question. We pass only this token's embedding to a linear classification head, optionally augmented with $\hat{H}(Y)$. We train all learned selectors with a binary cross-entropy loss targeting the PRM-only label (external correct, internal wrong) as the primary objective. Results under alternative PRM and conf label objectives are in Appendix A.8.

### 4.2 Comparing Selection Strategies

We now evaluate the selection strategies introduced in Section 3 on the same held-out test sets: the entropy threshold heuristic and the two families of learned selectors.

#### 4.2.1 Cross-Family Comparison

For context, Table 1 includes an oracle selector that always chooses the better signal for each instance. It reaches 66.86% on MMLU-Pro and 67.20% on MATH500, outpacing the external-only baseline by 6.07 and 2.80 percentage points. This difference also shows up in the tradeoff metric. The oracle achieves $\mathrm{Trd} = 1$ by construction (Section 4.1), since every external call it makes is a successful rescue, versus $\mathrm{Trd} \approx 0.10$–$0.14$ for the strongest practical operating points in Table 1 below. This gap is in the accuracy gain recovered per external call, not the fraction of the oracle's total accuracy improvement over the internal-only baseline.

A stronger external signal ($A_E > A_I$) only implies rescue mass exceeds harm mass in aggregate, not that a selector can identify which instances belong to each category. The oracle exceeds the external-only endpoint by calling the external signal only on the rescue mass, never on the harm mass, so every oracle call is a $+1$ outcome, so $\mathrm{Trd}_{\mathrm{oracle}} = 1$ exactly. We therefore use the PRM-only event as the primary training target because it directly identifies beneficial external calls. Harmful calls still affect final TTS accuracy.

[2] UW-Madison-Lee-Lab/MMLU-Pro-CoT-Train-Labeled on HuggingFace.

[3] UW-Madison-Lee-Lab/MMLU-Pro-CoT-Eval on HuggingFace.

[4] Given pre-collected CoT candidates and representations, DeBERTa-v3-small (44M parameters) trains in under 15 minutes per run on both benchmarks, and MLP in under 2 minutes.

Table 1: TTS accuracy (%), PRM call rate (CR), and trade-off (Trd.) at $\tau$=0.5, PRM-only label. Best observed configuration per learned family. Trd. = (TTS acc − internal acc)/CR (oracle: 1.000 by construction). Dashes: CR = 0. Full Pareto comparison in Figure 3.

| | MMLU-Pro | | | MATH500 | | |
|---|---|---|---|---|---|---|
| Method | Acc (%) | CR | Trd. | Acc (%) | CR | Trd. |
| Internal-only (Conf) | 53.21 | 0.00 | — | 53.20 | 0.00 | — |
| External-only (PRM) | 60.79 | 1.00 | 0.076 | 64.40 | 1.00 | 0.112 |
| Conf heuristic | **60.88** | 0.97 | 0.079 | **64.60** | 0.94 | 0.121 |
| Entropy heuristic | 60.50 | 0.71 | **0.103** | 63.20 | 0.73 | 0.137 |
| MLP (best) | 59.52 | 0.71 | 0.089 | 62.40 | 0.65 | **0.142** |
| DeBERTa (best) | 60.16 | 0.95 | 0.073 | 57.60 | 0.39 | 0.113 |
| Oracle | 66.86 | 0.14 | 1.000 | 67.20 | 0.14 | 1.000 |

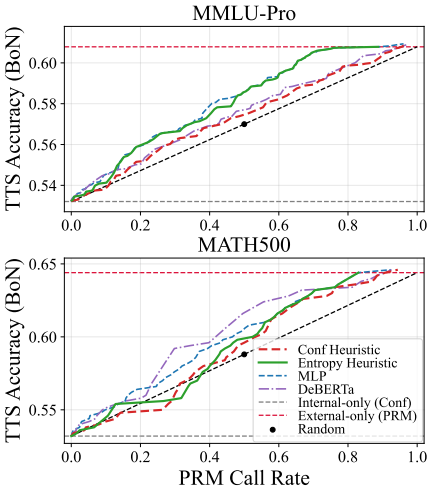



Figure 3: Pareto frontiers of TTS Acc vs. PRM call rate.

Figure 3 compares the Pareto frontiers of all selector families against a random-routing baseline. The black dot marks a random selector at $\mathrm{CR} = 0.5$, and the dashed iso-tradeoff line through it separates selectors that gain more accuracy per PRM call than random routing from those that do not.

The conf heuristic, despite reusing the same internal signal as the BoN scorer, is dominated by the entropy heuristic across the full frontier on both benchmarks (CR $\approx$ 0.97/0.94 vs. 0.71/0.73 to reach comparable peak accuracy), indicating that mean Self-Certainty ($\bar{s}_{\mathrm{conf}}$) is a weaker routing proxy than answer consistency ($\hat{H}$) in these settings.

The main finding is that the entropy heuristic and the best learned selectors produce closely matched frontiers across most of the call-rate range on both benchmarks. On MMLU-Pro the heuristic and MLP curves are nearly indistinguishable above CR $\approx$ 0.3, and on MATH500 the two track closely over most of the range. The under-1% accuracy margins at $\tau$=0.5 in Table 1 do not translate into a consistent full-frontier advantage. A paired bootstrap on excess AUC (Appendix A.7) confirms this pattern. Conditional on the reported Table 1 configurations, MLP obtains a small, statistically significant excess-AUC advantage over entropy on MATH500, but this advantage does not reproduce on MMLU-Pro or for DeBERTa, so the learned selectors' advantage over entropy is inconsistent rather than entirely absent.

One limitation appears at very low call rates on MATH500, where the heuristic's frontier dips below the random iso-tradeoff line (absent on MMLU-Pro): the PRM-only correct rate is non-monotone in entropy (Figure 4, Table 3), so a very high threshold routes the most uncertain questions to the PRM, even though neither signal typically succeeds on them. Learned selectors, which optimize directly for PRM-only correct probability rather than entropy, do not exhibit this failure mode.

Taken together, without labeled routing data, candidate disagreement heuristics such as the entropy threshold provide a useful training-free baseline that tracks the learned frontiers closely over most of the call-rate range. Learned selectors can improve on this in particular operating regions (e.g., low call rates on MATH500), but this advantage does not generalize consistently across datasets or the full frontier. This pattern replicates across two additional settings(Appendix A.10, Table 12).

Majority margin and distinct-answer count yield Pareto-AUC values similar to entropy on both benchmarks, while confidence variance performs slightly worse (Appendix A.6). We therefore use entropy as a representative candidate disagreement statistic rather than as a uniquely privileged signal. At its default operating point, the entropy heuristic reduces total generation-plus-scoring FLOPs by 13.6% on MATH500 and 17.1% on MMLU-Pro relative to always invoking the PRM (Appendix A.4).

### 4.2.2 Input Feature Analysis

Input feature analysis shows answer entropy is the most informative single feature, and augmenting it with richer representations or more expressive function classes does not yield reliable gains.

Table 2: MLP input feature comparison at $\tau$=0.5, PRM-only label. Bold: best accuracy per benchmark. Trd. = (TTS acc − internal acc)/CR.

| | MMLU-Pro | | | MATH500 | | |
|---|---|---|---|---|---|---|
| Configuration | Acc (%) | CR | Trd. | Acc (%) | CR | Trd. |
| Entropy-only | 59.23 | 0.59 | **0.102** | **62.40** | 0.65 | **0.142** |
| R-Rep | 53.40 | 0.06 | 0.032 | 61.20 | 0.80 | 0.100 |
| R-Rep + Ent + Std | 56.41 | 0.34 | 0.094 | 56.00 | 0.23 | 0.122 |
| Q-Rep | 56.17 | 0.42 | 0.071 | 57.00 | 0.29 | 0.131 |
| Q-Rep + Ent | **59.52** | 0.71 | 0.089 | 57.00 | 0.28 | 0.136 |

Table 3: Label distribution by entropy regime. Entropy-regime boundaries are discussed in Appendix A.10.

| Regime | $N$ | PRM-only | Conf-only | Both correct | Neither |
|---|---|---|---|---|---|
| *MMLU-Pro* | | | | | |
| (0, 0.35] | 429 | 0.012 | 0.000 | **0.874** | 0.114 |
| (0.35, 0.88] | 1243 | 0.156 | 0.079 | 0.415 | 0.350 |
| (0.88, 1.0] | 386 | 0.212 | 0.070 | 0.205 | **0.513** |
| *MATH500* | | | | | |
| (0, 0.35] | 100 | 0.040 | 0.000 | **0.930** | 0.030 |
| (0.35, 0.88] | 241 | 0.191 | 0.029 | 0.581 | 0.199 |
| (0.88, 1.0] | 159 | 0.126 | 0.044 | 0.119 | **0.711** |

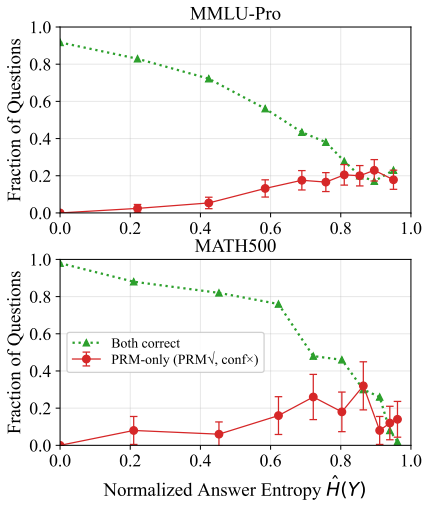



Figure 4: Outcome rates by entropy decile. As entropy rises, the "both correct" fraction falls, and PRM benefit increases before peaking and declining at the highest entropy levels, where "both wrong" becomes dominant.

Table 2 shows the MLP results for a representative subset of input configurations. Entropy-only achieves the highest TTS accuracy on MATH500 (62.40% at CR = 0.65). On MMLU-Pro, Q-Rep+Ent marginally outperforms entropy-only, but this advantage does not generalize. Entropy-only leads on MATH500 by a wide margin (62.40% vs. 57.00%). Adding response-level hidden-state representations (R-Rep) without entropy performs poorly on MMLU-Pro. Combining all features (R-Rep+Ent+Std) recovers somewhat but still trails entropy-only. The pattern holds across configurations. Entropy is the dominant signal, while additional features provide no consistent benefit.

For DeBERTa, appending answer entropy to the question embedding raises accuracy on both benchmarks (MMLU-Pro: 1.75%, MATH500: 2.2%), but at the cost of a substantially higher call rate (MMLU-Pro: +0.27, MATH500: +0.19). This behavior is consistent with entropy acting as a broad routing signal, increasing PRM usage without a proportional gain in tradeoff efficiency. Full results for all families and configurations are in Appendix A.5.

### 4.3 Structure Captured by Disagreement-Based Selection

Section 4.2.1 shows that candidate disagreement heuristics, including answer entropy, compete with learned selectors across most of the call-rate range. We now analyze the coarse structure these heuristics exploit, using answer entropy as a representative statistic. The key observation is that answer entropy captures a prominent coarse-grained structure of signal utility. Low-entropy candidate sets are usually already handled by the internal signal, whereas higher-entropy candidate sets are more likely to contain PRM-only rescue cases. This makes entropy a useful coarse routing statistic, and Table 5 shows that other disagreement measures behave similarly.

**Entropy tracks changes in internal-signal reliability.** Figure 4 and Table 3 show the same pattern from two views. At low entropy, sampled candidates tend to agree, and the internal signal is already reliable, so PRM-only rescue is rare. As entropy increases, the "both correct" fraction falls while the PRM-only rate rises. At the highest entropy levels, both-wrong cases become dominant, so PRM-only rescue can decline again. Entropy therefore does not perfectly predict rescue cases, but it captures the coarse transition from internal sufficiency to increased external-signal utility. Routing low-entropy questions to the internal signal accordingly avoids 29.3% of PRM calls on MMLU-Pro and 27.2% on MATH500, at an accuracy cost of only 0.29% and 1.20% relative to always calling the PRM. The relationship between answer entropy and the internal confidence score is further analyzed in Appendix A.3. The contrast with the conf heuristic, which measures token-level distributional

confidence rather than answer consistency, illustrates why entropy is the more effective routing feature.

**Limits of fine-grained rescue prediction.** The limited gains of learned selectors are not simply because entropy crowds out other features, since representation-only inputs also perform poorly. Entropy directly exposes the regime transition in internal-signal reliability, while the residual variation appears weakly predictable from the evaluated representations, though we cannot rule out that other architectures or objectives may capture more of it. However, an objective explicitly aligned with net routing utility also does not close the gap (Appendix A.8.3). Learned selectors consequently show limited discrimination on the PRM-only label, and high ROC-AUC on related objectives does not translate into better routing efficiency. Full distributions and label-objective comparisons are in Appendix A.7 and Appendix A.8.

### 4.4 Robustness Across External Scorers

To test whether the entropy heuristic's routing value depends on the external scorer, we rescore the same MATH500 candidate pool with Qwen2.5-Math-PRM-72B, Qwen2.5-Math-RM-72B, and AceMath-7B-RM. We use excess AUC to account for differences in external-scorer strength, with 95% CIs from 2000 paired bootstrap resamples (Appendix A.4). Excess AUC is positive for all three alternative scorers on MATH500 ($+0.0095$, $+0.0127$, and $+0.0164$), with confidence intervals excluding zero. The primary 7B PRM also has a positive point estimate ($+0.0070$), although its interval overlaps zero. VersaPRM on MMLU-Pro similarly yields positive excess AUC ($+0.0112$) with an interval excluding zero.

Routing value is thus positive in point estimate for every scorer evaluated. Only the primary MATH500 setting's CI overlaps zero, so we do not claim statistical significance for that setting in isolation. This evaluation used a fixed 32-candidate pool across scorers, so it does not speak to robustness under a different generator or sampling budget (assessed separately in Appendix A.10 and Appendix A.12). Full results, including the 72B-scale robustness check restricted to MATH500, are in Appendix A.9.

## 5 Conclusion

We have studied post-generation scoring signal selection for test-time scaling, where the decision is whether to invoke an external PRM to re-rank a fixed candidate set or rely on the internal signal. The complementarity we document between internal and external signals (Section 4.2.1) is only partially captured by cheap, training-free disagreement heuristics, and is not consistently improved upon by the learned selectors we evaluate despite labeled supervision, richer representations, and alternative supervision objectives (Sections 4.2.1–4.3 and Appendix A.8.3). These results show that simple candidate disagreement signals can capture broad patterns in when external scoring is useful, while the learned selectors we evaluate do not consistently identify additional instance-specific routing information beyond these patterns.

**Limitations and future work.** The selectors evaluated in our work do not reliably recover fine-grained routing information beyond coarse candidate disagreement. A key open question is therefore how to model the scorer-relative value of a candidate set more directly. Promising directions include architectures that reason over candidate-level ranking changes, objectives aligned with the marginal utility of external scoring, and joint adaptation of candidate generation and scorer selection.

### AI use statement

Generative AI tools were used to assist with writing and language polishing of the manuscript, including improvements to grammar, clarity, and phrasing. The authors determined and verified all scientific content, experimental results, and conclusions and take full responsibility for the final manuscript.

### Ethics statement

This work studies scoring signal selection for test-time scaling using publicly available benchmarks and pretrained models. It does not involve human subjects, user studies, personally identifiable

information, or the collection of sensitive data. We use all datasets and models under their respective terms of release.

The work focuses on improving test-time scaling through better use of scoring signals and does not introduce new model capabilities or release new datasets or foundation models. As with the underlying language models and reward models, deployment in real-world applications may require additional task-specific validation and safety evaluation.

### Reproducibility statement

We provide the experimental settings needed to reproduce the main results throughout the paper and appendix. Section 4.1 specifies the datasets, base models, candidate generation procedure, scoring signals, selector architectures, feature definitions, and training setup. Appendix A.4 describes the Pareto-AUC and excess-AUC metrics, paired-bootstrap procedure, and compute-cost calculation. Additional experimental settings for alternative objectives, generators, external scorers, sampling budgets, PRM aggregation rules, and internal signals are provided in the corresponding appendix sections.

Upon acceptance, we will publicly release the code and experimental configurations needed to reproduce the results, including scripts for candidate generation, selector training, and evaluation. All benchmarks and pretrained models used in the experiments are publicly available and cited in the paper.

## References

Pranjal Aggarwal, Aman Madaan, Yiming Yang, et al. Let's sample step by step: Adaptive-consistency for efficient reasoning and coding with llms. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pp. 12375–12396, 2023.

Tom Brown et al. Language Models are Few-Shot Learners. In *NeurIPS*, 2020.

Lingjiao Chen, Matei Zaharia, and James Zou. FrugalGPT: How to Use Large Language Models While Reducing Cost and Improving Performance. *arXiv preprint arXiv:2305.05176*, 2023.

Karl Cobbe et al. Training Verifiers to Solve Math Word Problems. *arXiv preprint arXiv:2110.14168*, 2021.

Dujian Ding, Ankur Mallick, Chi Wang, Robert Sim, Subhabrata Mukherjee, Victor Ruhle, Laks V.S. Lakshmanan, and Ahmed H. Awadallah. Hybrid LLM: Cost-Efficient and Quality-Aware Query Routing. In *International Conference on Learning Representations*, 2024.

Dujian Ding, Ankur Mallick, Shaokun Zhang, Chi Wang, Daniel Madrigal, Mirian Del Carmen Hipolito Garcia, Menglin Xia, Laks VS Lakshmanan, Qingyun Wu, and Victor Rühle. Best-route: Adaptive llm routing with test-time optimal compute. *arXiv preprint arXiv:2506.22716*, 2025.

Abhimanyu Dubey et al. The Llama 3 Herd of Models. *arXiv preprint arXiv:2407.21783*, 2024.

Chuan Guo et al. On Calibration of Modern Neural Networks. In *ICML*, 2017.

Chaoqun He, Renjie Luo, Yuzhuo Bai, Shengding Hu, Zhen Leng Thai, Junhao Shen, Jinyi Hu, Xu Han, Yujie Huang, Yuxiang Zhang, Jie Liu, Lei Qi, Zhiyuan Liu, and Maosong Sun. OlympiadBench: A Challenging Benchmark for Promoting AGI with Olympiad-Level Bilingual Multimodal Scientific Problems. In *Annual Meeting of the Association for Computational Linguistics*, 2024. URL https://api.semanticscholar.org/CorpusID:267770504.

Pengcheng He, Jianfeng Gao, and Weizhu Chen. Debertav3: Improving DeBERTa using electra-style pre-training with gradient-disentangled embedding sharing. *arXiv preprint arXiv:2111.09543*, 2021.

Dan Hendrycks et al. Measuring Mathematical Problem Solving With the MATH Dataset. *NeurIPS*, 2021.

Saurav Kadavath, Tom Conerly, Amanda Askell, Thomas Henighan, Dawn Drain, Ethan Perez, Nicholas Schiefer, Zachary Dodds, Nova Dassarma, Eli Tran-Johnson, Scott Johnston, Sheer El-Showk, Andy Jones, Nelson Elhage, Tristan Hume, Anna Chen, Yuntao Bai, Sam Bowman, Stanislav Fort, Deep Ganguli, Danny Hernandez, Josh Jacobson, John Kernion, Shauna Kravec, Liane Lovitt, Kamal Ndousse, Catherine Olsson, Sam Ringer, Dario Amodei, Tom B. Brown, Jack Clark, Nicholas Joseph, Benjamin Mann, Sam McCandlish, Chris Olah, and Jared Kaplan. Language Models (Mostly) Know What They Know. *ArXiv*, abs/2207.05221, 2022. URL https://api.semanticscholar.org/CorpusID:250451161.

Zhewei Kang, Xuandong Zhao, and Dawn Song. Scalable best-of-n selection for large language models via self-certainty. *arXiv preprint arXiv:2502.18581*, 2025.

Peng Kuang et al. Optimal Aggregation of LLM and PRM Signals for Efficient Test-Time Scaling. *arXiv preprint arXiv:2510.13918*, 2025.

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

Dong Bok Lee, Seanie Lee, Sangwoo Park, Minki Kang, Jinheon Baek, Dongki Kim, Dominik Wagner, Jiongdao Jin, Heejun Lee, Tobias Bocklet, Jinyu Wang, Jingjing Fu, Sung Ju Hwang, Jiang Bian, and Lei Song. Rethinking Reward Models for Multi-Domain Test-Time Scaling. *arXiv preprint arXiv:2510.00492*, 2025a.

Seanie Lee, Dong Bok Lee, Dominik Wagner, Minki Kang, Haebin Seong, Tobias Bocklet, Juho Lee, and Sung Ju Hwang. Saferoute: Adaptive model selection for efficient and accurate safety guardrails in large language models. In *Findings of the Association for Computational Linguistics: ACL 2025*, pp. 2053–2069, 2025b.

Yiwei Li, Peiwen Yuan, Shaoxiong Feng, Boyuan Pan, Xinglin Wang, Bin Sun, Heda Wang, and Kan Li. Escape sky-high cost: Early-stopping self-consistency for multi-step reasoning. In *International Conference on Learning Representations*, volume 2024, pp. 14751–14768, 2024.

Hunter Lightman et al. Let's Verify Step by Step. *arXiv preprint arXiv:2305.20050*, 2023.

Isaac Ong, Amjad Almahairi, Vincent Wu, Wei-Lin Chiang, Tianhao Wu, Joseph E Gonzalez, Mohammed Kadous, and Ion Stoica. Routellm: Learning to route llms from preference data. In *International Conference on Learning Representations*, volume 2025, pp. 34433–34448, 2025.

Amrith Setlur, Chirag Nagpal, Adam Fisch, Xinyang Geng, Jacob Eisenstein, Rishabh Agarwal, Alekh Agarwal, Jonathan Berant, and Aviral Kumar. Rewarding Progress: Scaling Automated Process Verifiers for LLM Reasoning. In *International Conference on Learning Representations (ICLR)*, 2025.

Charlie Snell et al. Scaling LLM Test-Time Compute Optimally can be More Effective than Scaling Model Parameters. In *ICLR*, 2025.

Katherine Tian, Eric Mitchell, Allan Zhou, Archit Sharma, Rafael Rafailov, Huaxiu Yao, Chelsea Finn, and Christopher D. Manning. Just Ask for Calibration: Strategies for Eliciting Calibrated Confidence Scores from Language Models Fine-Tuned with Human Feedback. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pp. 5433–5442, 2023.

Jonathan Uesato, Nate Kushman, Ramana Kumar, Francis Song, Noah Siegel, Lisa Wang, Antonia Creswell, Geoffrey Irving, and Irina Higgins. Solving Math Word Problems with Process- and Outcome-Based Feedback. *arXiv preprint arXiv:2211.14275*, 2022.

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. *arXiv preprint arXiv:2312.08935*, 2023a.

Xinglin Wang, Shaoxiong Feng, Yiwei Li, Peiwen Yuan, Yueqi Zhang, Chuyi Tan, Boyuan Pan, Yao Hu, and Kan Li. Make every penny count: Difficulty-adaptive self-consistency for cost-efficient reasoning. In *Findings of the Association for Computational Linguistics: NAACL 2025*, pp. 6919–6932, 2025.

Xuezhi Wang et al. Self-Consistency Improves Chain of Thought Reasoning in Language Models. In *ICLR*, 2023b.

Yubo Wang, Xueguang Ma, Ge Zhang, Yuansheng Ni, Abhranil Chandra, Shiguang Guo, Weiming Ren, Aaran Arulraj, Xuan He, Ziyan Jiang, Tianle Li, Max Ku, Kai Wang, Alex Zhuang, Rongqi Fan, Xiang Yue, and Wenhu Chen. MMLU-Pro: A More Robust and Challenging Multi-Task Language Understanding Benchmark, 2024. URL https://arxiv.org/abs/2406.01574.

Jason Wei et al. Chain-of-Thought Prompting Elicits Reasoning in Large Language Models. In *NeurIPS*, 2022.

Miao Xiong, Zhiyuan Hu, Xinyang Lu, Yifei Li, Jie Fu, Junxian He, and Bryan Hooi. Can LLMs Express Their Uncertainty? An Empirical Evaluation of Confidence Elicitation in LLMs. In *The Twelfth International Conference on Learning Representations*, 2024.

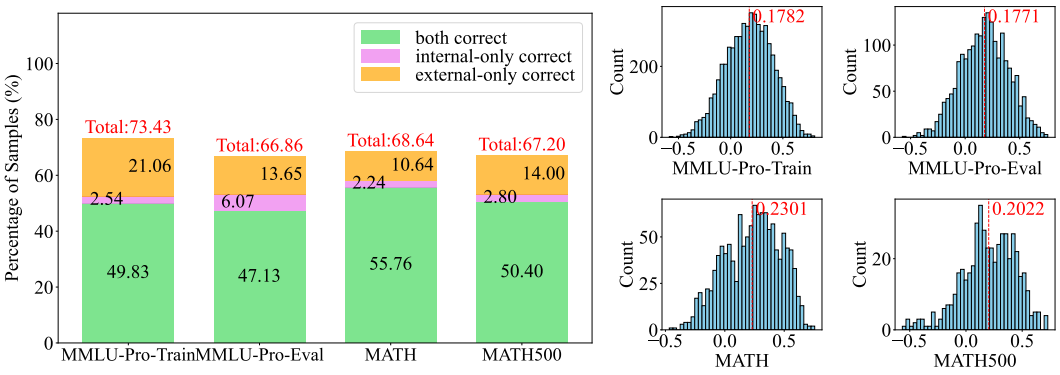
Shunyu Yao, Dian Yu, Jeffrey Zhao, Izhak Shafran, Tom Griffiths, Yuan Cao, and Karthik Narasimhan. Tree of Thoughts: Deliberate Problem Solving with Large Language Models. In *NeurIPS*, 2023.

Thomas Zeng, Shuibai Zhang, Shutong Wu, Christian Classen, Daewon Chae, Ethan Ewer, Minjae Lee, Heeju Kim, Wonjun Kang, Jackson Kunde, et al. VersaPRM: Multi-domain process reward model via synthetic reasoning data. *arXiv preprint arXiv:2502.06737*, 2025.

Zhenru Zhang, Chujie Zheng, Yangzhen Wu, Beichen Zhang, Runji Lin, Bowen Yu, Dayiheng Liu, Jingren Zhou, and Junyang Lin. The lessons of developing process reward models in mathematical reasoning. In *Findings of the Association for Computational Linguistics: ACL 2025*, pp. 10495–10516, 2025.

# A Appendix

## A.1 Difference Between Scoring Signals




(a) Question-level overlap under BoN

(b) Distribution of Kendall's $\tau$

Figure 5: Internal and external signals exhibit complementary strengths and low rank correlation. **(a)** BoN overlap at the question level: each signal exclusively solves a non-trivial fraction of questions the other fails on. **(b)** Per-question Kendall's $\tau$ between the rankings of $N$=32 candidates induced by the two signals, showing weak but consistently positive correlation (mean 0.18–0.23).

Both analyses in Figure 5 ask whether the signals differ enough for per-instance selection to be worthwhile, but approach the question from complementary angles.

The BoN comparison in Figure 5a takes a top-1 perspective: for each test question, we run BoN independently under the internal and external signals using the same pool of 32 candidates, and classify questions into four mutually exclusive outcomes: internal-only correct, external-only correct, both correct, or neither correct. This is the most directly performance-relevant view, since BoN TTS accuracy depends entirely on whether the top-ranked candidate is correct. The two signals agree on most questions, with roughly half solved correctly by both. At the same time, each signal exclusively solves a non-trivial fraction of questions the other misses. External-only wins are more frequent across both benchmarks, reflecting the PRM's advantage on uncertain questions, the internal-only region is non-negligible throughout, ruling out the strategy of always invoking the external signal.

The Kendall's $\tau$ analysis in Figure 5b examines the same disagreement at the full-ranking level. For each question, we compute $\tau$ between the full orderings of the 32-candidate pool under the internal and external signals, then plot the distribution of these per-question correlations. While the BoN comparison captures only whether the top-1 pick changes, this correlation reflects how differently the two signals order the entire candidate set, including disagreements on lower-ranked candidates that BoN never exposes. Mean $\tau$ ranges from 0.177 to 0.230 across benchmarks, confirming that the two signals are positively correlated but far from interchangeable in their rankings. This persistent low correlation indicates that the two signals induce substantially different ranking preferences over the candidate set, so the top-1 agreement visible in the BoN overlap does not imply broad agreement over the full ranking.

### A.2 Effect of Aggregation Rules

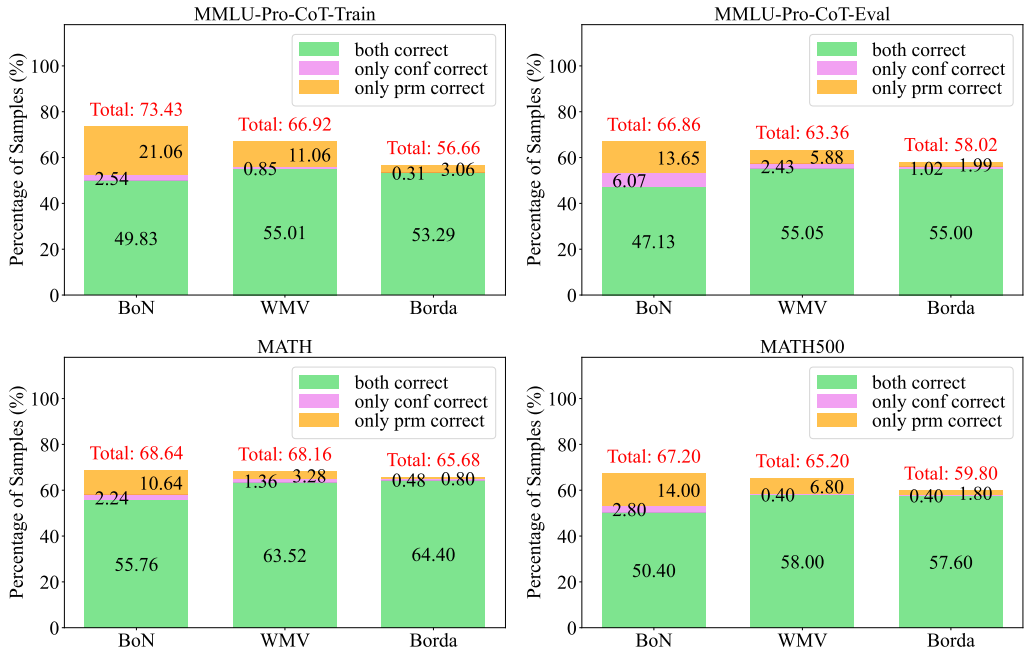



Figure 6: Question-level overlap between internal confidence and external PRM-based scoring signals under different aggregation rules. For each method (BoN, WMV, and Borda), questions are partitioned into three cases: both correct, only internal correct, and only external correct. Union accuracy denotes the proportion of questions solved by at least one signal under the same aggregation rule.

Figure 6 provides the full per-dataset overlap statistics under alternative aggregation rules, complementing the summary discussion in Section A.1. Across all four datasets, weighted majority voting (WMV) and Borda Voting substantially reduce the discrepancy between internal confidence and external PRM-based scores relative to BoN. Under these majority-based rules, the shared both-correct region dominates, while the scorer-specific only-internal-correct and only-external-correct regions

shrink substantially. This behavior is consistent with the mechanism of WMV and Borda: both methods rely more heavily on agreement structure among sampled candidates, which tends to smooth out ranking differences induced by the underlying scoring signals. In contrast, BoN consistently preserves much larger exclusive regions, particularly for the external signal, indicating that direct score-based selection better exploits fine-grained differences between the two scorers. These results suggest that majority-based aggregation provides a strong, robust baseline by consolidating consensus answers, but it is less effective at recovering minority-but-correct candidates favored by only one signal. BoN, by letting candidate-level score differences directly determine selection, remains the most sensitive setting for revealing and using the complementary information in internal confidence and external PRM-based scores.

### A.3 Relationship Between Answer Entropy and Confidence Score

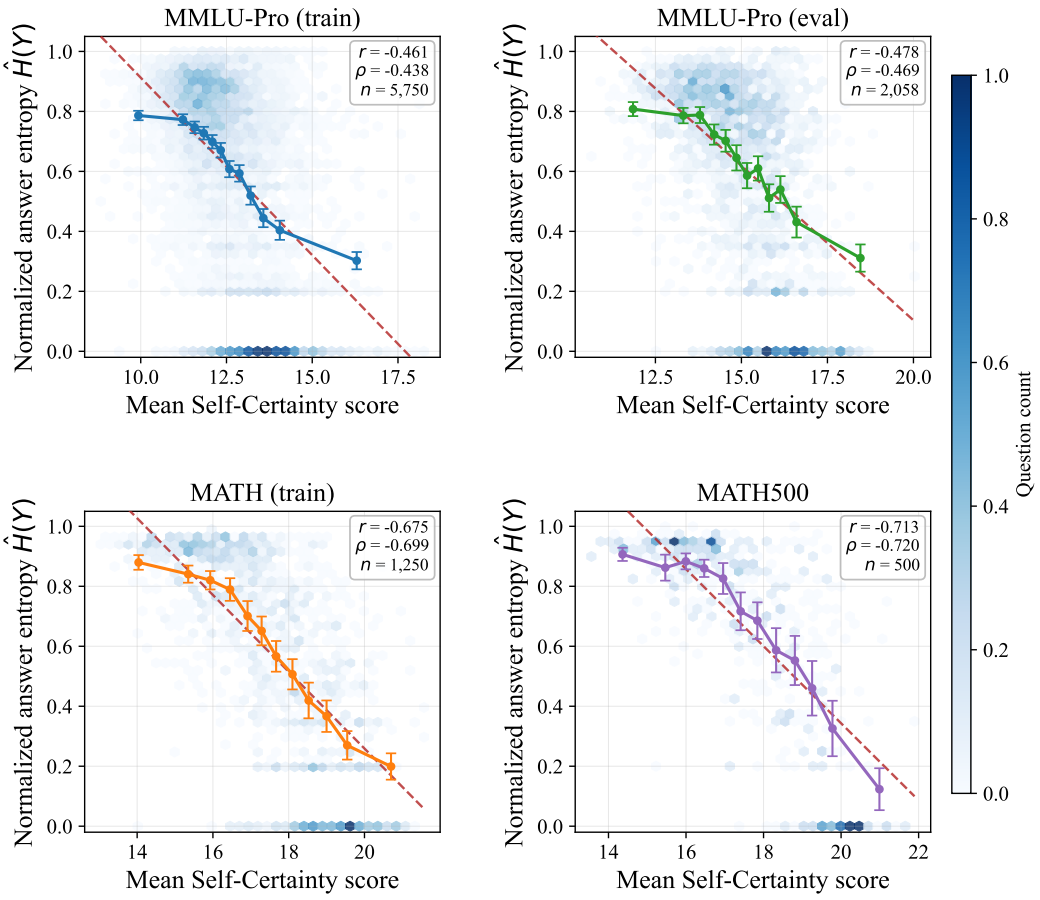



Figure 7: Normalized answer entropy $\hat{H}(Y)$ versus mean Self-Certainty score (averaged over $N$=32 candidates) across all four datasets. Each hexagonal bin encodes question count (blue shading). Colored markers show the bin-wise mean entropy $\pm$ 95% CI. The red dashed line is the linear regression fit. Pearson $r$ and Spearman $\rho$ are reported per dataset.

Figure 7 shows that normalized answer entropy and mean Self-Certainty score are consistently negatively correlated across all four datasets: Pearson $r = -0.46$ (MMLU-Pro train), $-0.48$ (MMLU-Pro eval), $-0.68$ (MATH train), and $-0.71$ (MATH500). The correlation is substantially stronger on MATH than MMLU. This is consistent with the structural difference between the two tasks: mathematical answers are precise symbolic expressions, so a confident model outputs the same expression across nearly all candidates. In contrast, MMLU is a multiple-choice task in which surface similarities between options allow some answer variation even under high confidence.

Two features of this relationship are worth noting. First, high-confidence instances are concentrated at low entropy, whereas low-confidence instances span a much wider entropy range. The second is that the correlation, while consistent, is far from tight ($|r| \leq 0.71$), confirming that entropy and confidence capture overlapping but non-identical information. Entropy and confidence therefore capture related but distinct information. This helps explain why entropy can provide routing information beyond mean Self-Certainty: the two signals tend to agree on which questions are hard in the aggregate, but diverge on individual instances in ways that matter for the selection decision.

### A.4 Evaluation and Cost Details

**Pareto-AUC and excess AUC.** For each selector, we sweep the decision threshold over its full range and retain the non-dominated points in PRM call rate–TTS accuracy space. We include the shared internal-only ($\mathrm{CR} = 0$) and external-only ($\mathrm{CR} = 1$) endpoints and compute Pareto-AUC by trapezoidal integration over $\mathrm{CR} \in [0, 1]$.

At call rate $c$, call-rate-matched random routing invokes the external scorer independently with probability $c$, giving expected accuracy

$$A_{\mathrm{rand}}(c) = (1 - c)A_I + cA_E, \tag{8}$$

where $A_I$ and $A_E$ are the internal-only and external-only accuracies. Its AUC is therefore

$$\mathrm{AUC}_{\mathrm{rand}} = \frac{A_I + A_E}{2}. \tag{9}$$

We define excess AUC as

$$\mathrm{ExcessAUC} = \mathrm{AUC}_{\mathrm{selector}} - \mathrm{AUC}_{\mathrm{rand}}. \tag{10}$$

This removes the accuracy gain obtained mechanically from calling a stronger external scorer more often and measures routing value relative to random routing at matched call rates.

**Statistical uncertainty.** We compute 95% confidence intervals using 2,000 paired bootstrap resamples over test questions. For each resample, we recompute the threshold sweep, Pareto frontier, Pareto-AUC, and excess AUC. Pairwise method comparisons use the same resampled question indices. These intervals capture test-question sampling uncertainty conditional on the trained selector and realized candidate set. We analyze training-seed and candidate-subset variability separately.

**Compute-cost estimate.** Generation cost is shared across selection strategies because routing occurs after candidate generation. Let $C_{\mathrm{gen}}$ denote generation FLOPs and $C_{\mathrm{ext}}$ the cost of scoring all candidates with the external model. At PRM call rate $c$, the expected generation-plus-scoring cost is

$$C(c) = C_{\mathrm{gen}} + cC_{\mathrm{ext}}. \tag{11}$$

Relative to always invoking the external scorer, the fractional FLOPs reduction is

$$\frac{(1 - c)C_{\mathrm{ext}}}{C_{\mathrm{gen}} + C_{\mathrm{ext}}}. \tag{12}$$

Using the entropy heuristic's default operating point, this corresponds to total generation-plus-scoring FLOPs reductions of 13.6% on MATH500 and 17.1% on MMLU-Pro.

### A.5 Input Feature Analysis

Figure 8 and Table 4 break down performance within each family, varying the input features while holding the learning objective and architecture constant. The following ablations use the PRM-only label throughout, the effect of alternative training objectives is analyzed in Appendix A.8.

For the MLP family, response-level entropy alone proves to be the strongest input feature, with additional representations providing little consistent benefit. Using only normalized answer entropy with no hidden-state representations achieves 59.23% at CR=0.59 on MMLU-Pro and 62.40% at CR=0.65 on MATH500. Representation-based configurations trail by a substantial margin on MMLU-Pro: Rep-only reaches only 53.40% (CR=0.06), and even the full combination (Rep + Ent + Std)

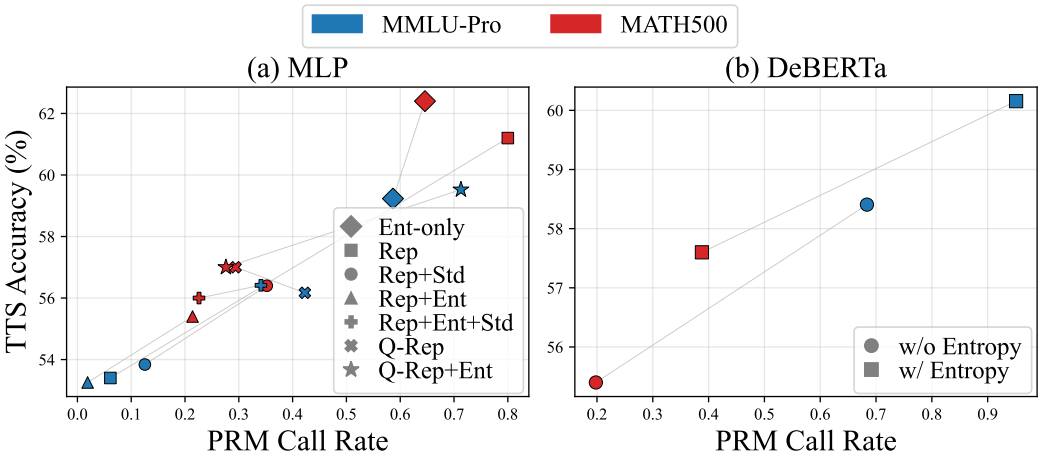



Figure 8: Within-family ablation at $\tau$=0.5, PRM-only label. Each point is one configuration, gray lines connect the same configuration across datasets (blue: MMLU-Pro, red: MATH500).

Table 4: Within-family ablation at $\tau$=0.5, PRM-only label. Bold: best accuracy within each family per benchmark. Trd. = (TTS acc − internal acc)/CR.

| | MMLU-Pro | | | MATH500 | | |
|---|---|---|---|---|---|---|
| Configuration | Acc (%) | CR | Trd. | Acc (%) | CR | Trd. |
| *MLP (response-level)* | | | | | | |
| Entropy-only | 59.23 | 0.59 | **0.102** | **62.40** | 0.65 | **0.142** |
| Rep | 53.40 | 0.06 | 0.032 | 61.20 | 0.80 | 0.100 |
| Rep + Std | 53.84 | 0.13 | 0.048 | 56.40 | 0.35 | 0.091 |
| Rep + Ent + Std | 56.41 | 0.34 | 0.094 | 56.00 | 0.23 | 0.122 |
| *MLP (question-level)* | | | | | | |
| Q-Rep | 56.17 | 0.42 | 0.071 | 57.00 | 0.29 | 0.131 |
| Q-Rep + Ent | **59.52** | 0.71 | 0.089 | 57.00 | 0.28 | 0.136 |
| *DeBERTa* | | | | | | |
| w/o entropy | 58.41 | 0.68 | 0.076 | 55.40 | 0.20 | 0.110 |
| w/ entropy | **60.16** | 0.95 | 0.073 | **57.60** | 0.39 | 0.113 |

achieves 56.41%, still about 3 percentage points below entropy-only. On MATH500, Rep-only reaches 61.20% but requires a much higher call rate (CR = 0.80), making its $\tau$=0.5 operating point Pareto-dominated by entropy-only (62.40% at CR = 0.65). Adding the per-dimension standard deviation of representations provides some gain over mean-only Rep on MMLU-Pro (53.40%→53.84%), but the effect is small and does not close the gap with entropy-only. Question-level representations (Q-Rep) perform similarly to the weakest response-level variants. Even Q-Rep + Ent, the best question-level configuration, reaches 59.52% on MMLU-Pro but falls to 57.00% on MATH500, 5.4 percentage points below entropy-only. The cross-dataset inconsistency of question-level features indicates that they do not yield reliable gains under the representations evaluated here.

For DeBERTa, appending answer entropy to the question embedding consistently raises accuracy: by +1.75 pp on MMLU-Pro (58.41%→60.16%) and by +2.20 pp on MATH500 (55.40%→57.60%). Both gains come at a substantially higher call rate (MMLU-Pro: 0.68→0.95, MATH500: 0.20→0.39). This behavior is consistent with entropy acting primarily as a broad alarm signal, deferring to the external scorer for a large fraction of inputs rather than filtering selectively, so tradeoff efficiency does not improve proportionally.

The ablations across both families converge on the same conclusion: among the evaluated features, answer entropy is the most informative and most robust single feature. Augmenting it with richer

representations or more expressive function classes yields inconsistent results across benchmarks, with no configuration reliably outperforming entropy-only across both datasets.

### A.6 Entropy Among Other Cheap Disagreement Statistics

Section 4.2.1 compares entropy with three other training-free measures computed from the same candidate set. Majority margin measures the gap between the top two answer vote shares, distinct-answer count measures the number of unique extracted answers, and confidence variance measures the spread of per-candidate Self-Certainty scores. Table 5 reports Pareto-AUC (Appendix A.4) and fixed-budget accuracy for all four measures. Entropy, majority margin, and distinct-answer count achieve very similar Pareto-AUC values on both benchmarks, while confidence variance is slightly lower.

Table 5: Pareto-AUC and fixed-budget accuracy for training-free candidate disagreement statistics.

| | MMLU-Pro | | MATH500 | |
|---|---|---|---|---|
| Signal | Pareto-AUC | Acc (%) @ CR 0.3/0.5/0.7 | Pareto-AUC | Acc (%) @ CR 0.3/0.5/0.7 |
| Entropy | 0.581 | 56.6 / 58.6 / 60.5 | 0.595 | 55.5 / 59.7 / 63.0 |
| Majority margin | 0.580 | 56.8 / 58.5 / 60.5 | 0.595 | 55.4 / 59.6 / 63.1 |
| Distinct-answer count | 0.581 | 57.1 / 58.6 / 60.0 | 0.594 | 56.2 / 59.5 / 62.5 |
| Confidence variance | 0.573 | 56.0 / 57.0 / 59.0 | 0.592 | 56.3 / 59.0 / 62.0 |

### A.7 Learned Selector Stability and ROC-AUC Distributions

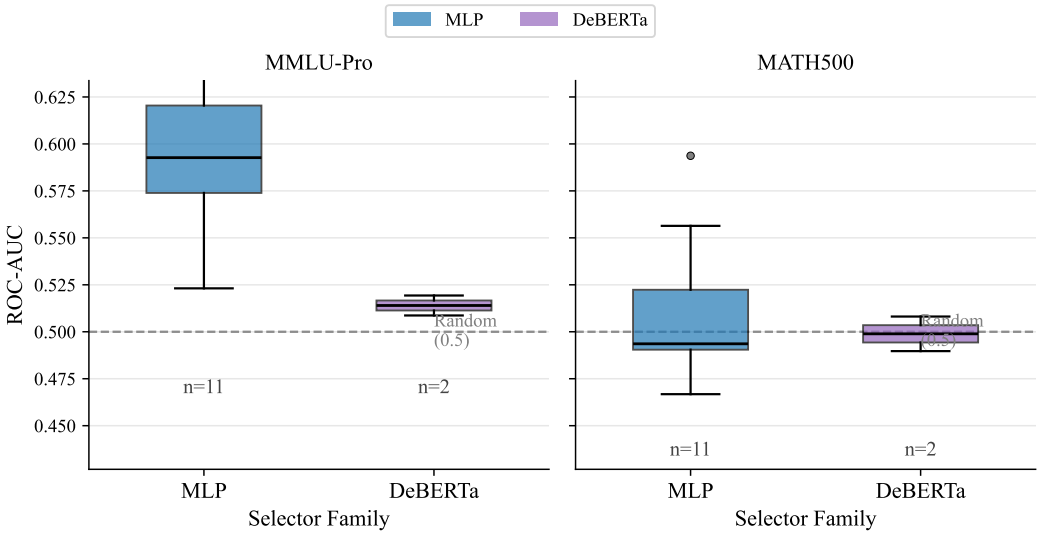



Figure 9: Distribution of ROC-AUC (binary PRM-only correct classification) across all runs within each selector family, on MMLU-Pro (left) and MATH500 (right). PRM-only label only. The dashed line marks random chance (0.5). Boxes span the interquartile range, whiskers extend to 1.5×IQR.

Figure 9 and Table 6 summarize binary classification performance and TTS stability across all PRM-only label selector runs. Two patterns stand out.

First, ROC-AUC on the PRM-only correct classification task is near-random or below chance for most families. On MATH500, the MLP family achieves only $0.510 \pm 0.036$ ROC-AUC, very slightly above chance, while DeBERTa falls to $0.499 \pm 0.009$, technically below random. On MMLU-Pro the picture is marginally better ($0.594 \pm 0.039$ for MLP, $0.514 \pm 0.005$ for DeBERTa), but still far from reliable discrimination. Despite these training-set sizes (the MLP family trains on approximately 5,175 MMLU-Pro questions and 1,125 MATH questions), discrimination remains limited under the evaluated setup. Under the representations, router families, and training-set sizes evaluated here, PRM-only correction remains substantially harder to discriminate than the alternative PRM/conf

Table 6: Binary classification and TTS stability for learned selector families at $\tau$=0.5. Values are mean $\pm$ standard deviation over all runs within each family (11 MLP, 2 DeBERTa), PRM-only label only. Degenerate: fraction of runs collapsing to constant predictions.

| | | MMLU-Pro | | MATH500 | |
|---|---|---|---|---|---|
| Family | | ROC-AUC | TTS Acc (%) | ROC-AUC | TTS Acc (%) |
| MLP | mean$\pm$std | $0.594 \pm 0.039$ | $55.61 \pm 2.11$ | $0.510 \pm 0.036$ | $57.29 \pm 2.54$ |
| | degenerate | 0/11 | | 0/11 | |
| DeBERTa | mean$\pm$std | $0.514 \pm 0.005$ | $59.28 \pm 0.87$ | $0.499 \pm 0.009$ | $56.50 \pm 1.10$ |
| | degenerate | 0/2 | | 0/2 | |

objectives (Appendix A.8.2): the set of instances where PRM succeeds but internal confidence fails is a relatively rare joint event that does not align strongly with the features evaluated here.

Second, MLP run outcomes vary substantially despite above-chance ROC-AUC. With near-random ROC-AUC on the PRM-only task, individual MLP run outcomes vary substantially: a TTS accuracy standard deviation of 2.11% on MMLU-Pro across 11 configurations indicates that cross-configuration choice materially affects downstream TTS performance. DeBERTa shows less variation across its two evaluated configurations ($\pm 0.87\%$ TTS on MMLU-Pro).

**Training-seed variability.** The variation above is across different input-feature configurations. To assess sensitivity to random initialization and data shuffling separately, we retrain each family's Table 1 feature configuration under 5 random seeds. The retraining script selects checkpoints by validation TTS accuracy rather than the criterion used to produce the original Table 1 runs, a difference known to prevent exact numeric reproduction on MATH500 (though not on MMLU-Pro) when the original seed is re-run under the new script. We therefore read the resulting mean $\pm$ std, reported in Table 7, as informative about seed-to-seed sensitivity under a closely related training setup rather than as an exact replication of the Table 1 protocol. Seed-to-seed variability is small relative to the cross-configuration variability above for three of the four family–dataset combinations, but is comparatively larger for DeBERTa on MATH500 ($\pm 3.22$ TTS, $\pm 0.265$ CR), indicating that this particular configuration is less robust to initialization than the others.

Table 7: Training-seed stability (mean $\pm$ std over 5 random seeds, fixed feature configuration, PRM-only label) for the Table 1 MLP and DeBERTa configurations.

| Dataset | Family | TTS Acc (%) | CR | Pareto-AUC |
|---|---|---|---|---|
| MMLU-Pro | MLP | $56.71 \pm 1.95$ | $0.411 \pm 0.185$ | $0.574 \pm 0.002$ |
| MMLU-Pro | DeBERTa | $58.12 \pm 0.90$ | $0.638 \pm 0.128$ | $0.570 \pm 0.003$ |
| MATH500 | MLP | $62.24 \pm 0.08$ | $0.640 \pm 0.003$ | $0.595 \pm 0.001$ |
| MATH500 | DeBERTa | $57.88 \pm 3.22$ | $0.412 \pm 0.265$ | $0.590 \pm 0.008$ |

We further compare the entropy heuristic with the exact MLP(best) and DeBERTa(best) configurations reported in Table 1. For each comparison, we compute the difference in excess AUC using 2000 paired bootstrap resamples. The entropy heuristic has excess AUC of $+0.0112$ on MMLU-Pro and $+0.0070$ on MATH500, as reported in Section 4.4 and Appendix A.9.

On MMLU-Pro, entropy has higher excess AUC than MLP(best) ($+0.0112$ vs. $+0.0054$), with a difference of $+0.0058$ (95% CI $[-0.0009, 0.0116]$). The difference is not statistically significant. Entropy also exceeds DeBERTa(best) ($+0.0112$ vs. $+0.0006$), with a difference of $+0.0105$ (95% CI $[0.0027, 0.0173]$).

On MATH500, MLP(best) slightly exceeds entropy ($+0.0079$ vs. $+0.0070$), with a difference of $-0.0009$ from the entropy-minus-MLP comparison (95% CI $[-0.0034, -0.0006]$). DeBERTa(best) has excess AUC of $+0.0030$, giving a difference of $+0.0040$ relative to entropy (95% CI $[-0.0092, 0.0155]$).

Across the four comparisons, a learned selector significantly outperforms entropy only for MATH500 MLP, with a margin of 0.0009. Entropy significantly outperforms DeBERTa on MMLU-Pro. These

comparisons are conditional on the configurations reported in Table 1. The bootstrap does not account for uncertainty from family-level configuration selection. The results are therefore consistent with the broader finding that the learned selectors evaluated here do not consistently improve over the entropy heuristic.

### A.8 Effect of Training Objective

Table 8: Best-configuration TTS accuracy (%), PRM call rate (CR), and tradeoff (Trd.) at $\tau$=0.5 for each training objective and selector family. Each cell reports the run with the highest TTS accuracy among all configurations within that family and label (11 MLP runs, 2 DeBERTa runs per label). Dashes: CR = 0 (tradeoff undefined). We repeat the entropy heuristic (training-free) and fixed baselines from Table 1 for reference.

| | | MMLU-Pro | | | MATH500 | | |
|---|---|---|---|---|---|---|---|
| Label | Family | Acc (%) | CR | Trd. | Acc (%) | CR | Trd. |
| Internal-only (Conf) | | 53.21 | 0.00 | — | 53.20 | 0.00 | — |
| External-only (PRM) | | 60.79 | 1.00 | 0.076 | 64.40 | 1.00 | 0.112 |
| Entropy heuristic | | 60.50 | 0.71 | 0.103 | 63.20 | 0.73 | 0.137 |
| | MLP | 59.52 | 0.71 | 0.089 | 62.40 | 0.65 | 0.142 |
| PRM-only | DeBERTa | 60.16 | 0.95 | 0.073 | 57.60 | 0.39 | 0.113 |
| | MLP | 60.40 | 0.97 | 0.074 | 62.40 | 0.77 | 0.120 |
| PRM | DeBERTa | 58.31 | 0.74 | 0.069 | 62.00 | 0.73 | 0.120 |
| | MLP | 59.18 | 0.58 | 0.103 | 62.40 | 0.66 | 0.140 |
| Conf | DeBERTa | 56.71 | 0.40 | 0.088 | 56.60 | 0.43 | 0.079 |

**How "best-configuration" entries are selected.** For every (label, family) cell in Table 8 and in Table 1 in the main text, per-configuration checkpoint selection uses the held-out validation split (via early stopping on validation loss). The reported operating point uses the default decision threshold ($\tau = 0.5$), not a validation-calibrated one. The reported entry is then the configuration with the highest test set accuracy among the (11 MLP / 2 DeBERTa) configurations evaluated for that label and family. This is a post-hoc upper-envelope summary of the evaluated family. It reports the best test performance observed across configurations with validation-selected checkpoints rather than a leakage-free generalization estimate of any single preselected configuration, and confidence intervals computed against one of these entries (e.g., the entropy–MLP comparisons in Appendix A.7) are conditional on this already-selected configuration and do not additionally account for the uncertainty of having selected it from among several.

We now ask whether the within-family patterns observed under the PRM-only label A.5 persist under alternative objectives. Table 8 compares the three training objectives across all selector families. The PRM-only label directly targets the complementary event where the PRM is correct and the internal signal is wrong. The "PRM" label instead predicts whether the external signal is correct, while the "conf" label predicts whether the internal signal is correct.

The results do not show a consistent ordering among these objectives. The PRM label generally invokes the external scorer more often than PRM-only while achieving a lower tradeoff (MLP MMLU-Pro: CR=0.97 vs. 0.71, Trd=0.074 vs. 0.089, DeBERTa MMLU-Pro: 0.069 vs. 0.073, MLP MATH500: 0.120 vs. 0.142). Its accuracy can nevertheless match or exceed that of PRM-only, as for MLP on MMLU-Pro (60.4% vs. 59.52%), suggesting that the additional calls do not translate into proportional accuracy gains. DeBERTa on MATH500 is an exception, where the PRM label attains a slightly higher tradeoff than PRM-only (0.120 vs. 0.113).

The "conf" label also performs competitively in some settings. On MMLU-Pro, it achieves the highest tradeoff among the three objectives for both learned families (MLP: 0.103, DeBERTa: 0.088, compared with 0.089 and 0.073 for PRM-only). TTS accuracy remains close for MLP (59.18% vs. 59.52%) but differs more for DeBERTa (56.71% vs. 60.16%). On MATH500, PRM-only gives the highest tradeoff for MLP (0.142), whereas the PRM label does so for DeBERTa (0.120).

Although PRM-only directly targets the rescue event, it does not consistently outperform the other supervision objectives across datasets and selector families.

### A.8.1 Within-Family Ablation by Label Objective

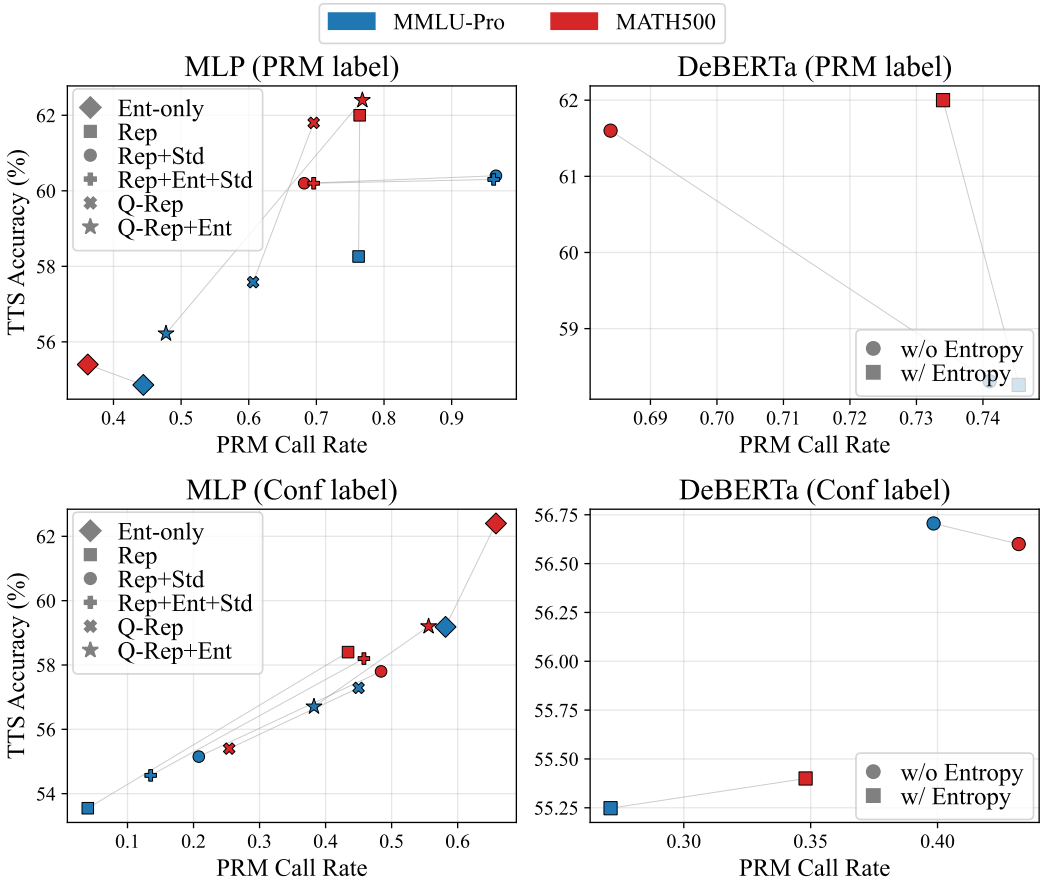



Figure 10: Within-family ablation at $\tau$=0.5 for the PRM label (top row) and conf label (bottom row). Each point is one configuration, gray lines connect the same configuration across datasets (blue: MMLU-Pro, red: MATH500). Compare to Figure 8 (PRM-only label).

Figure 10 shows within-family feature ablations under PRM and conf label objectives.

Under the PRM label, configuration ordering for MLP and DeBERTa is similar to PRM-only: entropy-based features remain the strongest signal, though representation-based variants are somewhat less disadvantaged than under PRM-only training, consistent with PRM correctness being more aligned with the evaluated features than PRM-only correction.

Under the conf label, the entropy-only configuration retains its leading position for MLP on both benchmarks, mirroring the PRM-only label pattern. This is consistent with both labels sharing the same structural predictor: entropy captures the regime transition in internal signal reliability regardless of which signal's correctness is targeted. DeBERTa's behavior is also similar: appending entropy increases the call rate without a proportionate accuracy gain, a pattern consistent with entropy acting as a broad alarm rather than a selective filter.

### A.8.2 ROC-AUC Distributions by Label Objective

Figure 11 and Table 9 reveal that PRM and conf objectives are substantially easier to learn than PRM-only. However, this learnability does not translate to better routing efficiency.

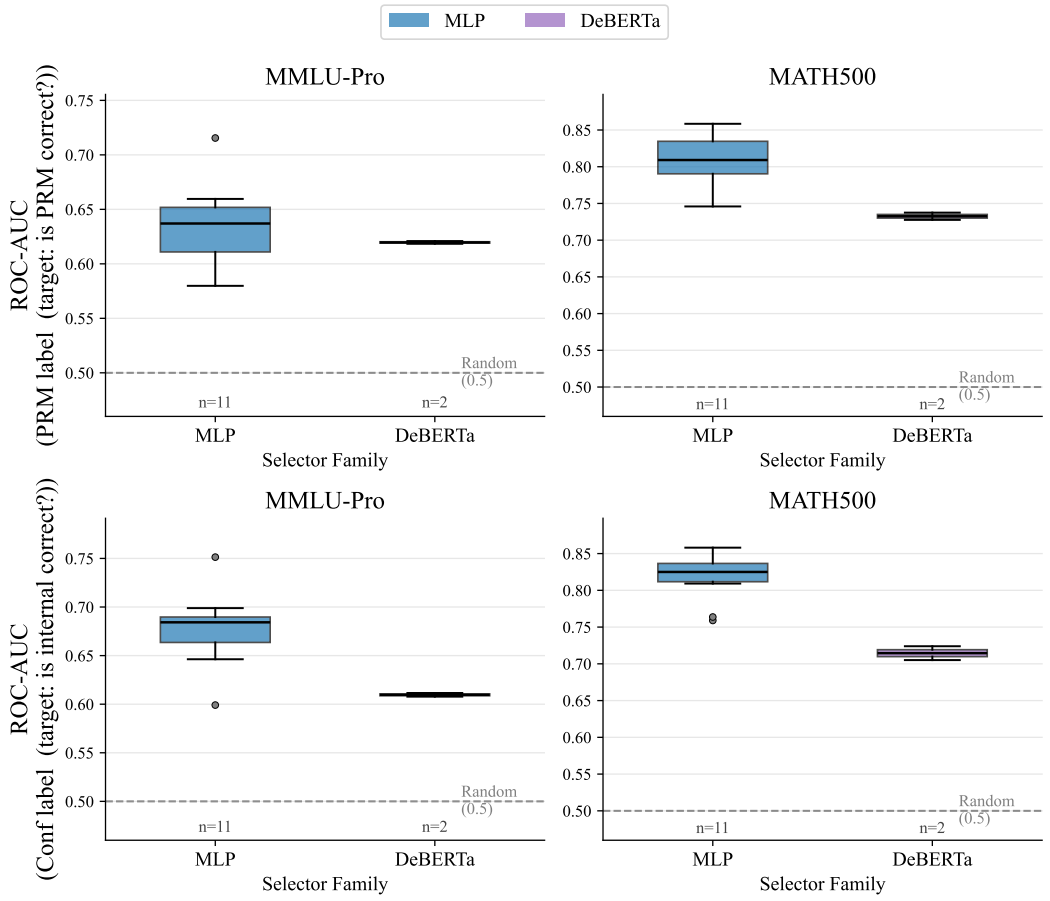



Figure 11: Distribution of ROC-AUC for PRM label (top row) and conf label (bottom row) across selector families, on MMLU-Pro (left) and MATH500 (right). The dashed line marks random chance (0.5). Compare to Figure 9 (PRM-only label, where MLP ROC-AUC is 0.510–0.594).

Table 9: ROC-AUC mean $\pm$ std for PRM and conf label objectives across selector families and datasets (11 MLP, 2 DeBERTa runs per label). For reference, PRM-only label ROC-AUC from Table 6 is reproduced in the last block.

| | | MMLU-Pro | MATH500 |
|---|---|---|---|
| Label | Family | ROC-AUC | ROC-AUC |
| PRM | MLP | $0.634 \pm 0.037$ | $0.808 \pm 0.033$ |
| | DeBERTa | $0.620 \pm 0.001$ | $0.733 \pm 0.005$ |
| Conf | MLP | $0.678 \pm 0.036$ | $0.818 \pm 0.030$ |
| | DeBERTa | $0.610 \pm 0.002$ | $0.715 \pm 0.009$ |
| PRM-only (ref.) | MLP | $0.594 \pm 0.039$ | $0.510 \pm 0.036$ |
| | DeBERTa | $0.514 \pm 0.005$ | $0.499 \pm 0.009$ |

The learnability gap between PRM-only and the other two objectives is large. On MATH500, MLP ROC-AUC rises from $0.510 \pm 0.036$ to $0.808 \pm 0.033$ (PRM) and $0.818 \pm 0.030$ (conf), an absolute increase of about 0.30, i.e. nearly 30 percentage points. On MMLU-Pro the improvement is more modest but consistent: MLP rises from $0.594$ (PRM-only) to $0.634$ (PRM) and $0.678$ (conf). DeBERTa shows an analogous gain. The PRM-only task is empirically harder for the evaluated selectors: it targets a rarer event ($\approx$14% positive rate) than the PRM and conf labels, which are more frequent and correlate more strongly with the evaluated features, including answer entropy.

**The conf-vs-PRM learnability gap is family-dependent.** For MLP, the conf objective yields slightly higher ROC-AUC than the PRM objective on both benchmarks (MMLU-Pro: 0.678 vs. 0.634, MATH500: 0.818 vs. 0.808). This pattern does not hold for DeBERTa, where conf is slightly lower than PRM on both datasets (MMLU-Pro: 0.610 vs. 0.620, MATH500: 0.715 vs. 0.733).

**High learnability does not imply high routing efficiency.** Despite much higher ROC-AUC, the PRM label generally yields lower routing efficiency than PRM-only (Table 8), with MATH500 DeBERTa as the exception noted above, and the conf label matches PRM-only in tradeoff for MLP on MATH500 (0.140 vs. 0.142) despite an absolute ROC-AUC increase of ≈30 percentage points. This decoupling arises because ROC-AUC measures the ability to rank instances by label probability, while routing efficiency depends on whether the decision boundary correctly separates PRM-only correct instances from the many instances where both signals succeed. Predicting PRM correctness is substantially easier for the evaluated selectors than identifying the marginal value of a PRM call.

### A.8.3 Net-Utility-Aligned Regression Objective

The PRM-only label used above rewards external-only correction ($E = 1, I = 0$) but does not explicitly penalize the complementary failure mode, internal-only correction being overridden ($I = 1, E = 0$). One possible explanation for the limited routing efficiency of learned selectors is that this asymmetric label, rather than any deeper limitation, is the bottleneck. To test this, we define a scalar net-utility target $u = \mathbf{1}[E{=}1] - \mathbf{1}[I{=}1] \in \{-1, 0, +1\}$, train the same MLP and DeBERTa architectures with an MSE regression loss against $u$ (routing decision: call the external signal iff the predicted $\hat{u} \geq 0$), and otherwise hold the search space identical to the PRM-only setting: the same 11 MLP feature combinations $\times$ 3 dropout rates and 2 DeBERTa configurations $\times$ 3 dropout rates, on the same data splits and candidate sets.

For this comparison, we select one configuration per objective using held-out validation TTS accuracy under its default decision rule ($p \geq 0.5$ for rescue and $\hat{u} \geq 0$ for utility), without using test-set performance for selection. We then evaluate the selected model once on the test set and report its full accuracy–call-rate frontier and excess AUC (Appendix A.4).

Both objectives use the same feature and architecture grid, consisting of 11 MLP feature combinations $\times$ 3 dropout rates and 2 DeBERTa configurations $\times$ 3 dropout rates. The data splits and candidate sets are shared, while only the supervision target and loss differ. We exclude repeated-seed rescue runs from the seed-stability analysis from the selection pool so both objectives select from the same 33 MLP and 6 DeBERTa runs.

Table 10: Net-utility-aligned regression objective vs. the PRM-only (rescue) objective, both under validation-based configuration selection from an identical-size pool (33 MLP / 6 DeBERTa runs per objective). Excess AUC relative to call-rate-matched random routing (Appendix A.4). 95% CI from 2000 paired bootstrap resamples.

| Dataset | Family | Utility | Rescue | Utility−Rescue | 95% CI |
|---|---|---|---|---|---|
| MATH500 | MLP | $+0.0013$ | $-0.0065$ | $+0.0078$ | $[-0.0025, 0.0176]$ |
| MATH500 | DeBERTa | $-0.0026$ | $+0.0045$ | $-0.0071$ | $[-0.0158, 0.0013]$ |
| MMLU-Pro | MLP | $-0.0005$ | $+0.0059$ | $-0.0064$ | $[-0.0126, -0.0000]$ |
| MMLU-Pro | DeBERTa | $-0.0010$ | $+0.0011$ | $-0.0021$ | $[-0.0067, 0.0031]$ |

Table 10 provides little evidence that the utility-aligned objective improves over rescue. Three of the four confidence intervals include zero. The remaining comparison, MMLU-Pro MLP, favors rescue, with an upper confidence bound of $-1.7 \times 10^{-5}$. This result is also marginal and is one of four comparisons without multiplicity correction. The validation-selected utility configurations also do not improve over the entropy heuristic. Their excess AUC is lower than entropy's on both benchmarks and for both selector families. On MMLU-Pro, the differences are statistically significant for both MLP ($-0.0120$, 95% CI $[-0.0192, -0.0048]$) and DeBERTa ($-0.0125$, 95% CI $[-0.0193, -0.0052]$). On MATH500, the differences are not statistically significant. These results do not imply that net utility is inherently unlearnable or that supervision choice is unimportant. They show only that the scalar utility-regression objective evaluated here does not consistently improve routing under the current

architectures, features, and training setup. The limitation observed in Section 4.3 therefore cannot be attributed solely to the rescue-only supervision target.

### A.9 Robustness of Routing Value Across External Scorers

We fix the same 32-candidate pool and internal signal used throughout, and replace only the external scorer used for BoN selection, so that any change in excess AUC (Appendix A.4) reflects the scorer alone. On MATH500, in addition to the main Qwen2.5-Math-PRM-7B setting, we score the same candidates with a larger process reward model (Qwen2.5-Math-PRM-72B) and two outcome reward models (Qwen2.5-Math-RM-72B and AceMath-7B-RM). On MMLU-Pro we report only the primary VersaPRM setting for reference, since this specific robustness check (alternative scorers at a different scale and supervision type) was only run on MATH500.

Table 11: Excess AUC over call-rate-matched random routing for entropy-heuristic routing under different external scorers. 95% CI from 2000 paired bootstrap resamples.

| External scorer | Scale | Supervision | External-only Acc. | Excess AUC | 95% CI |
|---|---|---|---|---|---|
| *MATH500* | | | | | |
| Qwen2.5-Math-PRM-7B | 7B | Process | 64.40% | $+0.0070$ | $[-0.0014, 0.0154]$ |
| Qwen2.5-Math-PRM-72B | 72B | Process | 65.80% | $+0.0095$ | $[0.0012, 0.0180]$ |
| Qwen2.5-Math-RM-72B | 72B | Outcome | 66.40% | $+0.0127$ | $[0.0038, 0.0220]$ |
| AceMath-7B-RM | 7B | Outcome | 67.00% | $+0.0164$ | $[0.0081, 0.0254]$ |
| *MMLU-Pro* | | | | | |
| VersaPRM (main) | 7B | Process | 60.79% | $+0.0112$ | $[0.0064, 0.0160]$ |

The point estimate for excess AUC is positive for every scorer evaluated. The 95% CI excludes zero for all three alternative MATH500 scorers and for the MMLU-Pro primary scorer. For the MATH500 primary scorer used in the main experiments, the point estimate is positive ($+0.0070$) but the 95% CI includes zero ($[-0.0014, 0.0154]$), so we do not claim significance for that setting in isolation. We similarly do not interpret the increasing point estimates on MATH500 ($+0.0070 \rightarrow +0.0095 \rightarrow +0.0127 \rightarrow +0.0164$) as evidence that a stronger scorer makes routing easier: model identity, parameter scale, and process-vs-outcome supervision all change simultaneously across these four rows, so we report only that the three alternative scorers yield larger excess-AUC point estimates than the primary 7B PRM baseline, without a causal claim about which factor drives the difference.

### A.10 Generalization Across Base Models and PRM Variants

The entropy calibration analysis in Section 4.3 uses Llama-3.1-8B-Instruct as the generator and, respectively, VersaPRM (MMLU-Pro) and Qwen2.5-Math-PRM-7B (MATH500) as the external scoring models. To assess whether the entropy–utility relationship holds beyond this specific configuration, we replicate the calibration analysis under three additional generator–PRM combinations.

**Choice of entropy regimes.** In the main text, Table 3 summarizes the decile trends using three entropy regimes: low entropy $(0, 0.35]$, medium entropy $(0.35, 0.88]$, and high entropy $(0.88, 1.0]$. These boundaries are descriptive rather than tuned decision thresholds. The lower boundary $\hat{H} = 0.35$ aligns with the 20th-percentile boundary of both held-out test distributions: the second decile edge is 0.342 on MMLU-Pro and 0.350 on MATH500. Below this region, PRM-only rescue is consistently rare, so the internal signal is usually sufficient. The upper boundary $\hat{H} = 0.88$ corresponds to the high-entropy transition point on MATH500, where the seventh decile edge is approximately 0.887 and the PRM-only rate reaches its peak before declining at the highest entropy levels. Using the same regime boundaries across benchmarks and replications allows us to compare whether the same qualitative structure persists: internal correctness dominates at low entropy, PRM-only rescue becomes more frequent at medium entropy, and neither signal succeeds reliably at very high entropy.

**Same base model, alternative PRM (Figure 12).** We replace Qwen2.5-Math-PRM-7B with LlamaPRM800K on MATH500 while keeping Llama-3.1-8B-Instruct as the generator. The decile-wise outcome profiles are nearly identical to the main calibration figure (Figure 4): internal correctness

decreases with entropy, PRM-only rescue peaks in the medium-to-high entropy region, and neither-correct outcomes dominate the highest-entropy bins. This suggests that the entropy regimes are not an artifact of the particular PRM used in the main MATH500 experiment.

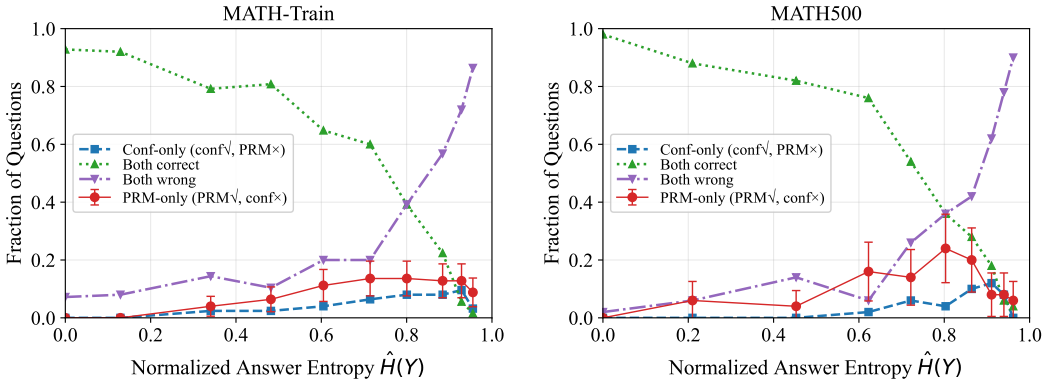



Figure 12: Outcome rates by entropy decile for Llama-3.1-8B-Instruct with an alternative PRM, LlamaPRM800K: MATH-Train (top) and MATH500 (bottom). Compare with Figure 4, which uses Qwen2.5-Math-PRM-7B on MATH500. The three-regime pattern is preserved under this PRM substitution.

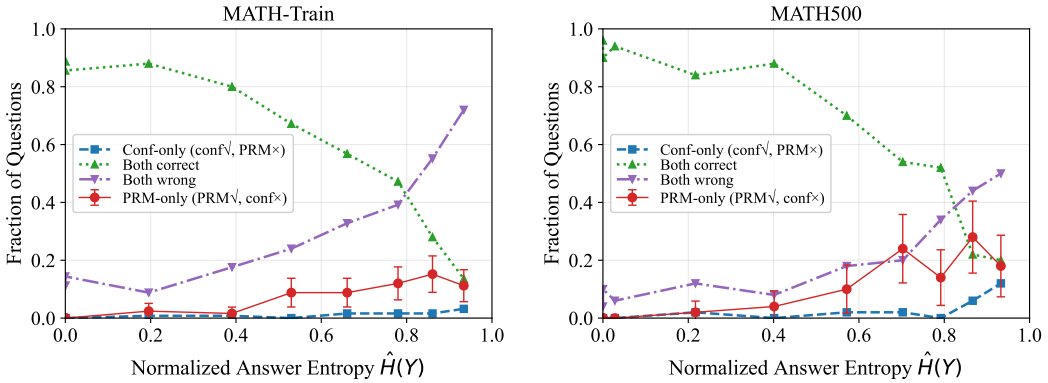



Figure 13: Outcome rates by entropy decile for Qwen2.5-7B-Instruct with Qwen2.5-Math-PRM-7B, on MATH-Train (left) and MATH500 (right). All available questions in each pool are used (1,250 and 500, respectively), irrespective of any internal train/validation/test partitioning of the underlying data. The three-regime pattern reproduces for this different base model.

**Different base model (Figures 13–14).** We use Qwen2.5-7B-Instruct as the generator on both MATH-Train and MATH500, paired separately with Qwen2.5-Math-PRM-7B and LlamaPRM800K. Despite the different generator and the inclusion of training-split data, both figures reproduce the same monotone decline in both-correct and the same PRM-only correct peak in the mid-entropy region. The both-wrong rate rises sharply at high entropy on MATH500 under both PRMs, consistent with the main finding.

Together, these results suggest that the qualitative three-regime pattern is not specific to the primary generator–PRM pairing and recurs across the additional combinations evaluated here. When answer entropy is low, the model samples consistently and its internal scoring is reliable, when entropy rises into the medium range, the external PRM provides incremental benefit while the internal signal degrades, when entropy is very high, both signals struggle. This is consistent with the calibration relationship in Section 4.3 being tied to the generator's sampling behavior rather than to a specific choice of PRM.

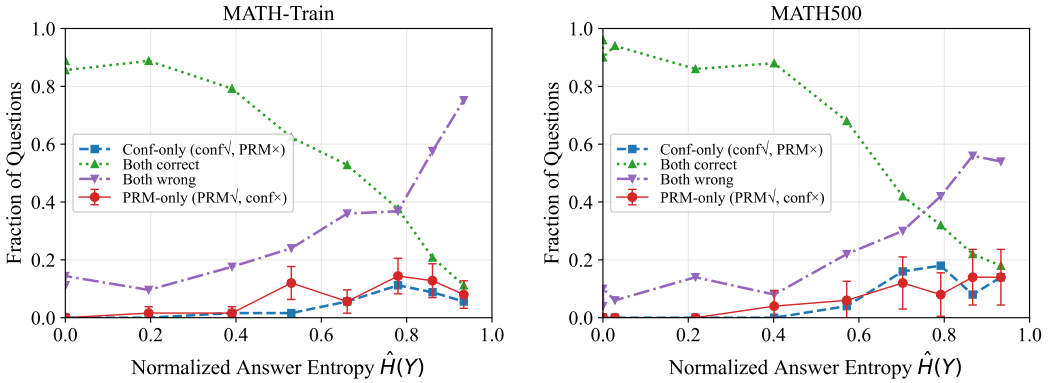



Figure 14: Outcome rates by entropy decile for Qwen2.5-7B-Instruct with LlamaPRM800K, on MATH-Train (left) and MATH500 (right). The pattern is consistent with Figure 13, supporting robustness across the evaluated PRM choices for the Qwen2.5-7B-Instruct generator.

Table 12: TTS accuracy (%), PRM call rate (CR), and tradeoff (Trd.) at $\tau$=0.5, PRM-only label, matching the Table 1 protocol. Best observed configuration per learned family (post-hoc family upper envelope.

| | Qwen2.5-7B-Instruct + Qwen2.5-Math-PRM-7B | | | Llama-3.1-8B-Instruct + LlamaPRM800K | | |
|---|---|---|---|---|---|---|
| Method | Acc | CR | Trd. | Acc | CR | Trd. |
| Internal-only (Conf) | 69.40 | 0.00 | — | 53.20 | 0.00 | — |
| External-only (PRM) | 77.00 | 1.00 | 0.076 | 59.60 | 1.00 | 0.064 |
| Entropy heuristic | **76.60** | 0.50 | 0.145 | **58.80** | 0.73 | **0.077** |
| MLP (best) | **76.60** | 0.53 | 0.136 | 58.00 | 0.67 | 0.072 |
| DeBERTa (best) | 74.80 | 0.36 | **0.149** | 54.00 | 0.12 | 0.066 |
| Oracle | 79.40 | 0.10 | 1.000 | 63.80 | 0.11 | 1.000 |

**Full selection comparison across generator–PRM combinations (Table 12 and Figures 15–16).** To verify that the selection results from Section 4.2 also generalize, we replicate the full comparison of selector families under two additional generator–PRM pairings: Qwen2.5-7B-Instruct with Qwen2.5-Math-PRM-7B, and Llama-3.1-8B-Instruct with LlamaPRM800K. Both use the same 1,250-problem MATH-Train split (90%/10% train/validation), evaluated on the full 500-question MATH500 test set. MLP and DeBERTa architectures, objectives, and hyperparameters follow the main experiments exactly (Section 4.1). The MLP input dimension is set to 3584 for Qwen2.5-7B-Instruct.

Both settings reproduce the main qualitative finding that disagreement-based routing remains competitive with the evaluated learned selectors. For Qwen2.5-7B-Instruct with Qwen2.5-Math-PRM-7B, the entropy heuristic reaches within 0.4 percentage points of the external-only (PRM) accuracy (76.60% vs. 77.00%) at half the call rate (CR = 0.50, Trd. = 0.145). The best MLP configuration reaches the same accuracy at a similar call rate (Trd. = 0.136), while the best DeBERTa configuration trades lower accuracy and call rate for the highest tradeoff among the three (Trd. = 0.149). For Llama-3.1-8B-Instruct with LlamaPRM800K, LlamaPRM800K is a weaker external signal than Qwen2.5-Math-PRM-7B (external-only accuracy 59.60% vs. 64.40% in the main experiments). The entropy heuristic achieves the highest accuracy and tradeoff among the evaluated selectors (58.80%, Trd. = 0.077), and the learned selectors offer no clear advantage. In both cases, the Pareto frontiers (Figures 15–16) reproduce the pattern from Figure 3: the entropy heuristic and best MLP curves are closely matched across the call-rate range, showing that the same competitive pattern recurs in the additional generator–PRM pairings evaluated here.

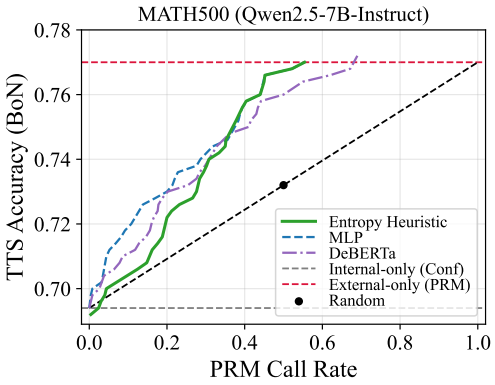



Figure 15: Pareto frontiers for Qwen2.5-7B-Instruct + Qwen2.5-Math-PRM-7B on MATH500.

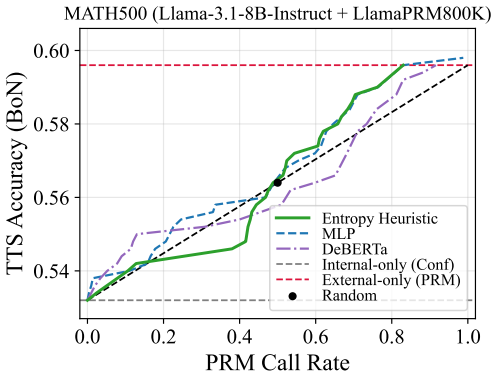



Figure 16: Pareto frontiers for Llama-3.1-8B-Instruct + LlamaPRM800K on MATH500.

### A.11 Sensitivity to Candidate Subset Composition

The main experiments fix a single sampled candidate set per question. Here we ask a narrower question: for the same question, does the scorer-relative outcome change if a different subset of candidates is sampled? We reuse the existing $N = 8/16/32/64$ sampling-budget robustness pool (a shared 64-candidate pool per question, generated once) and repeatedly resample $N$-sized subsets from it (20 repeats per $N \in \{8, 16, 32\}$, $N = 64$ uses the full pool and has no subsets to compare), recording, for each question and each resample, the four-way outcome (internal-only, external-only, both, or neither correct).

We report three complementary flip statistics: preference flip (whether some resamples for a question yield $E{=}1, I{=}0$ while others yield $I{=}1, E{=}0$, i.e., whether the same question can favor either signal depending on which candidates were sampled), routing-relevance flip (whether a question's outcome is ever a rescue or harm event in some resamples but neither in others), and four-way outcome flip (any change among the four outcome categories across resamples), together with an ANOVA-corrected estimate of the fraction of total outcome variance attributable to within-question (candidate-subset) variation rather than between-question variation. We estimate this via a one-way random-effects ANOVA decomposition of the rescue indicator over the $R{=}20$ resampled subsets per question: $\mathrm{MS}_{\text{within}}$ is the unbiased within-question variance, the between-question variance is recovered as $(\mathrm{MS}_{\text{between}} - \mathrm{MS}_{\text{within}})/R$ (clipped at zero), and $R_{\text{within}}$ is the within-question component divided by their sum.

Table 13: Candidate-subset sensitivity, by sampling budget $N$. Preference/routing-relevance/four-way are flip rates (%) across 20 resampled subsets per question. $R_{\text{within}}$ is the estimated within-question share of total variance after finite-repeat correction.

| Dataset | $N$ | Preference flip | Routing-relevance flip | Four-way flip | $R_{\text{within}}$ |
|---|---|---|---|---|---|
| MMLU-Pro | 8 | 27.41% | 69.14% | 69.39% | 77.87% |
| | 16 | 21.19% | 61.71% | 61.90% | 67.71% |
| | 32 | 12.10% | 51.90% | 52.09% | 54.35% |
| MATH500 | 8 | 15.80% | 49.80% | 51.00% | 74.45% |
| | 16 | 10.00% | 46.40% | 47.00% | 65.52% |
| | 32 | 5.20% | 37.40% | 37.80% | 48.57% |

Table 13 shows that all three flip rates decrease as $N$ increases, as expected: a larger sample is less sensitive to which particular candidates happen to be drawn. Under repeated subsampling from the fixed 64-candidate pool, scorer outcomes vary across candidate subsets, and this variation is still present, though smaller, at the paper's main $N{=}32$ setting: scorer-preference reversals occur

for 12.10% of MMLU-Pro and 5.20% of MATH500 questions, and the estimated within-question variance share remains close to half (54.35% and 48.57%, respectively).

We therefore report this only as a sensitivity property of the labels under subset resampling and do not use it to explain the learned-selector results.

### A.12 Routing Value Across Sampling Budgets

We additionally ask how the entropy heuristic's excess AUC (Appendix A.4) over call-rate-matched random routing changes as the sampling budget $N$ varies, using the same $N = 8/16/32/64$ pool as Appendix A.11 (at $N = 64$, the full pool, there is a single realization rather than resampled subsets). This 64-candidate pool is an independently regenerated sample for the same questions used in the main experiments (Section 4.1), not the same candidates underlying Table 1. Its $N = 32$ point is therefore a separate generation-run realization and is not expected to exactly reproduce the main-text $N = 32$ excess-AUC values. Only the qualitative trend across $N$ within this pool is of interest here.

Table 14: Entropy heuristic's excess AUC (Appendix A.4) by sampling budget $N$: mean $\pm$ std over 20 resampled subsets per $N < 64$, a single realization at $N = 64$ (the full pool, no resampling).

| Dataset | $N = 8$ | $N = 16$ | $N = 32$ | $N = 64$ |
|---|---|---|---|---|
| MMLU-Pro | $+0.0058 \pm 0.0025$ | $+0.0050 \pm 0.0022$ | $+0.0028 \pm 0.0015$ | $+0.0009$ |
| MATH500 | $+0.0097 \pm 0.0030$ | $+0.0111 \pm 0.0026$ | $+0.0136 \pm 0.0032$ | $+0.0183$ |

Table 14 shows that, on MATH500, where the external signal's advantage over the internal signal is largest, excess AUC increases with $N$, from $+0.0097$ at $N = 8$ to $+0.0183$ at $N = 64$. On MMLU-Pro, the internal and external signals are closer in overall strength, and excess AUC narrows toward $+0.0009$ at $N = 64$, indicating that routing value is not uniformly increasing in sampling budget across benchmarks. The reported standard deviations for $N < 64$ reflect variability in which subset of this 64-candidate pool was drawn (the same source of variability as Appendix A.11), not independent generation-run variability, so they should not be read as an estimate of run-to-run reproducibility.

### A.13 Robustness to PRM Step-Score Aggregation

This is distinct from the answer-aggregation rules (BoN/WMV/Borda) in Appendix A.2: here we vary how a single PRM combines its own step-level scores into one per-candidate score, holding BoN answer selection fixed. The main experiments aggregate a PRM's step-level scores by the minimum operator. We additionally compute excess AUC (Appendix A.4) for the entropy heuristic under two alternative aggregation rules, mean and last-step, using the same candidate pools and PRMs. On MMLU-Pro, excess AUC is positive for all three rules (min: $+0.0112$, reproducing the main-text primary-scorer result exactly, since the main experiments already use min aggregation, mean: $+0.0085$, last: $+0.0103$), and the 95% CI excludes zero for all three. On MATH500, all three point estimates are again positive but smaller (min: $+0.0070$, mean: $+0.0061$, last: $+0.0068$), and the 95% CI includes zero for all three, the same pattern of positive point estimates with intervals overlapping zero noted for MATH500 in Appendix A.9. We report the positive point estimates without treating them as significant on MATH500 in isolation.

### A.14 Robustness to Internal Signal Choice

The main experiments use Self-Certainty as the internal signal. We additionally evaluate the entropy heuristic when the internal signal is instead the sequence-level average log-probability (AvgLogP) of the sampled response under the base model. Self-Certainty is the substantially stronger internal signal in absolute BoN accuracy (by roughly 10 percentage points on both benchmarks), but excess AUC for the entropy heuristic remains positive under either internal signal. With AvgLogP on MATH500, however, excess AUC is budget-dependent rather than uniformly positive: it is below the random-routing baseline at a call rate of 0.3 and only turns positive at call rates of 0.5 and 0.7. We do not observe this budget-dependence with Self-Certainty, so routing value under a weaker internal signal should not be assumed to hold uniformly across the call-rate range.