WHEN IS MORE REASONING WORTH IT?

Anonymous authors
Paper under double-blind review

ABSTRACT

The next generation of reasoning models will not be the ones that think longest, but the ones that know when to think. Adapting reasoning effort to each question is a central open problem, and solving it would move the frontier between cost and accuracy. This work is a step in this direction. We build a controlled evaluation of effort selection and release its data: repeated responses from open models at every effort setting on the same questions. We first show that adapting effort to the question can save nearly half of the generated tokens with almost no loss in accuracy. We then compare two practical strategies, predicting from the question text and checking agreement among cheap-model answers, with every model call charged. We propose question-level cost forecasts that improve an existing selector without extra generation. Finally, we identify where further gains lie: knowing where extra reasoning helps matters beyond knowing where it is expensive, but this benefit is model specific and current forecasts capture little of it.

1 INTRODUCTION

Reasoning models can think longer to answer better, and most now expose this as a reasoning effort setting: low, medium, and high, or thinking off and on. The setting is typically fixed in advance for every question, yet the value of extra thinking varies widely between questions. For two of our models, Qwen3.5-27B and Gemma-4-31B, questions whose earlier thinking-off answers were all correct receive about half of the added tokens when thinking is enabled (51.3% and 49.1%), yet contribute an estimated 5.5% and 1.3% of the net accuracy gain. A model that knew when to think would avoid most of this spending and reserve its effort for the questions that need it. Adapting effort to each question is therefore a central open problem: it would move the frontier between cost and accuracy, and it is a building block for agents that decide for themselves how hard to think at each step.

Prior work predicts the benefit of additional computation (Damani et al., 2025), or question-specific success and cost for routing among models and reasoning budgets (Somerstep et al., 2025; Fernandez et al., 2026). CARROT compares question-specific and average costs under the same success predictor, finding little benefit in its setting (Somerstep et al., 2025); RADAR uses average costs (Fernandez et al., 2026). We ask what gain information adds beyond costs, how much of that value can be predicted for new questions, and when extra guidance is worth buying. Our contribution connects these questions through controlled comparisons and repeated responses, which also reveal whether question-specific gains are reproducible within models and shared across them.

An information framework for evaluating effort selection. Using the same response bank, we compare four sources of information. The first two sources serve as reference points: **S1)** The target's own earlier answers to a question give an oracle-like ceiling: they show what per-question information is worth when it comes for free. **S2)** Benchmark averages choose one setting for every question in a benchmark, providing a baseline/floor for question-specific selection. The other two sources guide decisions on new questions: **S3)** Predictors learned from the question text, including our adaptation of RADAR, show how much of the ceiling's value can be forecasted for new questions. **S4)** Answers from a cheap model, obtained before selection, test whether this information is worth the cheap model's cost. Within the gain/cost comparisons, we hold the selection rule fixed and replace question-specific gains, costs, or both with benchmark averages to isolate their value. The paid comparison then charges probes and target calibration, while separate analyses also charge reference-response acquisition. The ceiling and floor give reference points against which new methods for selecting reasoning effort can be evaluated.

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.

An open dataset of repeated responses at every effort setting. We release 597,744 scored responses from nine models on the same 2,518 questions across ten benchmarks, with 8 or 16 independent responses per question at every effort setting, together with token counts, question groups, and grading code. Repeated responses let us choose effort from one set and evaluate it on another. They also let future selectors be compared on identical questions without generating new evaluation answers.

What makes effort choices better: gains. With the target's own earlier answers (S1), choosing effort per question reduces tokens by 48% relative to always using high effort, for a 0.29-point accuracy loss. With the same per-question costs, adding per-question gains improves accuracy by 1.62 points and saves 370 tokens per answer at a fixed token penalty. Accuracy gains are reproducible within models but less shared across families: their correlation is 0.20, against 0.65 for added costs. At token use halfway between each model's cheapest and most expensive settings, measured gains add 1.66 accuracy points beyond the same question-specific costs. When gains must instead be predicted from new questions' text (S3), their added value is 0.36 points with 128 calibration questions per benchmark (Section 3).

Practical improvements from costs and cheap-model answers. Replacing each setting's average cost in our within-model RADAR adaptation with a question-specific cost improves accuracy by 0.69 points at equal token use, with RADAR success predictions unchanged. A simple prompt-length predictor alone obtains 0.67 points. When target token budgets are fixed before evaluation, cost forecasts improve mean accuracy by 1.15 points while using 3.6% fewer actual tokens (Section 4). Cheap-model answers (S4) can be competitive with learned selectors. In a Qwen case study, four low-effort Qwen 9B answers guide Qwen 27B: it uses low effort when all four agree and high otherwise, without correctness labels for the new question. After charging probes and target calibration, this agreement beats the best tested learned selectors by 0.38 points at equal cost when calibration is shared across 1,000 later answers per benchmark; sharing it across 10,000 makes learned selection more competitive (Section 5). Paid guidance should be judged against learned alternatives at the reuse level expected in deployment.

Together, these comparisons distinguish information that improves effort choices from information the tested selectors can predict or afford to acquire. They make model- and effort-specific gain prediction a concrete target for future methods.

2 Experimental design

We choose one effort setting for a fixed answering model (the *target*), then generate one new answer. Higher effort starts a fresh generation and does not continue a low-effort response.

Responses and benchmarks. We collect eight or sixteen independent responses at every setting from GPT-OSS 20B/120B, Qwen3.5-9B/27B, GLM-4.7-Flash, EXAONE-4.0-32B, Nemotron-3-Nano-30B-A3B, Granite-4.2-8B, and Gemma-4-31B-it on each question of each benchmark.

The ten benchmarks cover mathematics (GSM8K, MATH500), knowledge (GPQA Diamond, MMLU-Pro), narrative and argument reasoning (MuSR, ReClor), logic and state tracking (ProofWriter, ZebraLogic, CogniLoad), and programming (LiveCodeBench) (Cobbe et al., 2021; Hendrycks et al., 2021; Rein et al., 2024; Wang et al., 2024; Sprague et al., 2024; Yu et al., 2020; Tafjord et al., 2021; Lin et al., 2025; Kaiser et al., 2025; Jain et al., 2025). Programs are scored against tests; the other benchmarks use fixed final-answer rules. Related questions, such as problems from the same coding contest, stay together in training/evaluation splits and uncertainty calculations (Appendix A). We use *task* as a synonym for benchmark in tables and figures.

GPT-OSS offers low, medium, and high; Granite offers off, brief, and on; the other models offer thinking off and on. These model-provided controls are the *native settings*. Every setting remains available. In cross-model summaries, low denotes low/off and high denotes high/on. All 21 model–setting combinations have the same 81,920-token output limit. We measure actual token use: Granite brief is cheaper than off.

Gain and cost. For model m, question q, and setting e, let p_{mqe} be the probability of a correct answer and μ_{mqe} its expected generated tokens. Relative to a baseline setting L,

$$\Delta_{mqe} = p_{mqe} - p_{mqL}, \qquad \Delta\mu_{mqe} = \mu_{mqe} - \mu_{mqL}. \tag{1}$$

We call these quantities the *gain* and *added token cost*. Raising success from 40% to 60% gives a gain of 20 percentage points (pp). Token counts include reasoning and final answers, including incorrect and incomplete responses. We retain negative gains when higher effort reduces success, and negative token differences when it uses fewer tokens. Costs are tokenizer-specific generated tokens until the priced comparisons in Section 5.

Changing one input at a time. The gain/cost comparisons use the same selection rule,

$$\widehat{e}_q = \arg\max_{e \in E_m} \{\widehat{p}_{mqe} - \eta \widehat{\mu}_{mqe}\}. \tag{2}$$

Here E_m contains all settings of the target model, and hats denote estimates. The rule chooses the highest predicted success after penalizing token use, so a larger η places more weight on saving tokens. At $\eta = 10^{-5}$, one percentage point of predicted accuracy offsets 1,000 additional tokens. Subtracting the same baseline success and token estimates from every setting leaves the choice unchanged: the rule therefore compares accuracy gains against added token costs. We compare benchmark-average gains and costs with question-specific gains, costs, or both (Table 1). The benchmark-average version chooses one setting for all questions in a benchmark, which we call a *task default*. Each comparison uses the same underlying responses or fitted predictions. Section 4.1 separately holds RADAR's success predictions and selection criterion fixed while changing its costs.

Information and evaluation. We distinguish three sources of information available before choosing effort. Earlier-answer experiments use scored responses to the same question. Text predictors learn from other questions and then choose effort from the new question's text alone. Paid agreement uses extra cheap-model answers to the new question, without correctness labels. In every case, the responses used to evaluate the choice are separate from those used to make it (Table 64). We average questions within each benchmark, then benchmarks equally and, for aggregate results, target models equally.

We report paired 95% bootstrap confidence intervals. Related questions are resampled together within each benchmark, and every compared method uses the same resampled questions. The evaluated models and fitted procedures remain fixed during this resampling (Appendix C).

What fixed settings achieve, and where high effort still helps. For GPT-OSS 20B, medium captures 77% of the low-to-high accuracy gain using 26% of the added tokens; for 120B, the corresponding shares are 69% and 21%. Questions with both correct and incorrect earlier medium-effort answers represent 30% of the 20B questions and 21% of the 120B questions, yet account for about 69% of the independently measured medium-to-high gain in each model. An efficient intermediate setting and useful question-specific increases therefore coexist. Table 65 and Appendix B report all models' setting returns, benchmark differences, and supporting controls.

3 The opportunity for question-specific effort

3.1 Choosing effort from earlier answers nearly halves generation tokens

How much can earlier answers to the same question improve effort choices relative to always using high effort? For each model and question, half the responses at each setting—four or eight— estimate success rates and mean token use. We choose the setting with the highest estimated success rate, breaking ties by fewer tokens ($\eta = 0$ in Equation 2). Average correctness and token use in the other half then estimate the outcome of one new answer at that setting. We reverse the halves and average the two evaluations. Reported tokens count the subsequent answer, excluding the acquisition of the earlier responses.

Averaged over the nine models and ten benchmarks, these choices reach 81.50% accuracy with 4,694 tokens per answer, compared with 81.79% and 9,011 for always high. This reduces generated-token use by 47.91 [46.53, 49.37]% for an accuracy loss of only 0.29 [0.08, 0.50] points (Figure 1A). Thus, when earlier measurements are available, nearly half of the always-high token expenditure can be avoided while retaining almost all of its accuracy.

Substantial extra spending can bring little additional accuracy. For Qwen 27B, questions whose earlier off-mode answers are all correct receive 51.3% of the added tokens when reasoning is enabled, but contribute an estimated 5.5% of the net accuracy gain. Gemma's corresponding shares are 49.1% and 1.3%. The responses defining these groups are separate from those measuring their gains and tokens. Figure 16 shows every model. These comparisons locate where extra spending brings little measured improvement; the independently evaluated selector above measures the saving from actually changing effort choices.

3.2 What measured gains add beyond costs

Using the same earlier responses and settings, we compare four versions of Equation 2 (Table 1). Averaging gains or costs within each benchmark removes their question-specific information. In particular, costs-only selection still uses accuracy information: every question receives its benchmark's average gain for each setting.

Table 1: **Measured gains improve selection beyond the same cost estimates.** All four selectors use the same earlier responses and token penalty $\eta = 10^{-5}$. "Average" denotes a target–setting benchmark average. Accuracy and tokens describe a new answer evaluated on independent responses and the earlier responses are not charged. Entries are rounded independently, and paired differences are reported in the text.

	Estimates supplied		Independent outcomes	
Selector	Gains	Costs	Accuracy (%)	Tokens
Benchmark averages	Average	Average	80.08	5,414
Costs only	Average	Per question	79.38	4,220
Gains only	Per question	Average	81.32	4,438
Costs and gains	Per question	Per question	81.01	3,850

At the fixed token penalty $\eta = 10^{-5}$, adding measured gains to the same question-specific costs improves accuracy by 1.62 [1.37, 1.87] points while saving 370 [271, 470] tokens per answer. The evaluated accuracy-minus-token-penalty score improves for all nine models (Table 34). Compared with benchmark averages for both quantities, using both question-specific estimates raises accuracy from 80.08% to 81.01% and reduces tokens from 5,414 to 3,850 per answer (Table 1). Thus measured gains add useful information beyond knowing costs, not merely beyond using one setting for an entire benchmark.

The benefit also appears at equal token use. A fixed token penalty can lead selectors to spend different amounts. We therefore also compare accuracy at the same measured mean token use. For each method, we evaluate policies across token penalties, include all fixed settings, and allow random mixtures whose policy choice is independent of the question. We report the best evaluated accuracy at each expenditure. Here spending is matched after evaluation; Section 4.3 separately fixes budgets before evaluating answers.

For each target, the *midpoint* is halfway between the mean token counts of its cheapest and most expensive fixed settings. We match token use within each target before averaging accuracy differences. At this midpoint, question-specific costs improve accuracy over benchmark-wide defaults by 1.23 points, and adding measured gains contributes a further 1.66 points (Figure 1B).

3.3 Models differ in which questions benefit

Do the same questions benefit from more reasoning across models? Across model families, the correlation is 0.20 for gains, compared with 0.65 for added tokens and 0.74 for high-effort success (Table 2). Models are therefore much more similar in which questions they solve or spend tokens on than in which questions their effort increase helps.

We estimate each question's gain twice, using independent halves of the recorded responses. Questions with larger estimated gains in one half tend to have larger estimates in the other. Depending on the model, an estimated 51%–75% of the within-benchmark variation in these estimates reflects reproducible differences between questions rather than response-sampling noise (Appendix C).

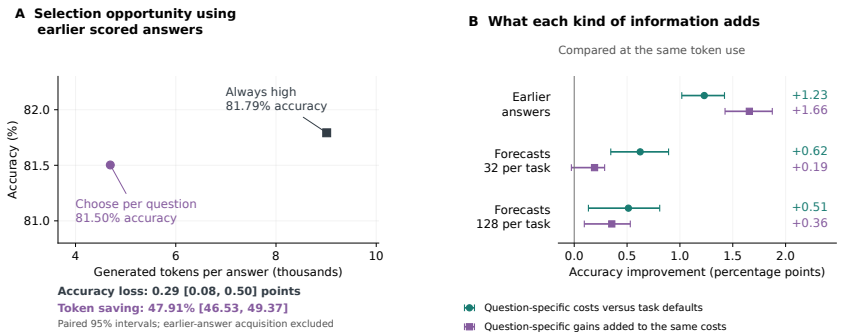


Figure 1: **Question-specific choices nearly halve generation tokens; gains add value beyond costs.** A: choices from earlier scored answers are evaluated on independent responses against always high, using $\eta = 0$. B: accuracy added by question-specific costs over benchmark defaults, then by gains over the same costs, at each target's midpoint token expenditure. Forecasts use 32 or 128 calibration questions per benchmark. Both panels average nine targets and ten benchmarks [paired 95%] while acquisition of earlier measurements and training data is excluded.

The particular effort increase also matters. For GPT-OSS across sizes (i.e., 20B and 120B), we compare gain patterns for the same increase (low-to-medium or medium-to-high) with patterns for successive increases within one size. On average, the same increase is more similar across sizes than successive increases within a size (Appendices C–D). Which questions benefit therefore depends on both the model and the effort settings being compared.

Table 2: **Models agree more about added costs than accuracy gains.** Signal correlations compare questions within benchmarks before pooling [paired 95%]. Independent response halves separate reproducible question-level variation from response-sampling noise.

Question-level quantity	Across-family correlation
High-effort success probability	$0.74\,[0.72, 0.75]$
Added tokens	$0.65\,[0.63, 0.67]$
Accuracy gain	$0.20\,[0.18, 0.22]$

3.4 Predicting gains before generating an answer

We now replace earlier-answer measurements with predictions trained on other questions, testing what gain forecasts add to the same cost forecasts.

Our predictor learns in two stages. First, a frozen embedding represents the question text, and kernel ridge regression predicts average success rates and token costs of other models, called *references*. Second, ridge regressions map these predictions to the target's gains and costs. We fit this second stage using 32 or 128 example questions per benchmark, with one target response at every setting scored for correctness and tokens (*target calibration*). Costs are fitted on a log scale. Once trained, the predictor reads only the new question and does not request reference answers to that question.

The reference models exclude the target's model family. For every model, training and calibration exclude the evaluation questions and all related questions in their groups. Increasing target calibration from 32 to 128 questions changes only the target-fitting data while the reference training data and first-stage predictors remain fixed (Appendix F.3).

At the midpoint token expenditure, cost forecasts improve accuracy over task defaults by 0.62 [0.35, 0.89] points with 32 calibration questions per benchmark. Adding gain forecasts contributes a further 0.19 points, with an interval including zero. With 128 questions, their contribution is 0.36 [0.10, 0.53] points, compared with 1.66 for measured gains (Figure 1B). Earlier answers provide several scored responses to each evaluated question, whereas forecasts must generalize from other questions. The gap therefore reflects information access and training as well as prediction quality.

Benefits depend on the budget and selection rule. At the 25% point of the fixed-setting token range, task defaults outperform cost-only forecasts. Separately, adding gain forecasts often lowers accuracy minus the token penalty when that penalty is fixed (Figure 10B; Table 31). Excluding ZebraLogic and CogniLoad from training and evaluation retains a 0.37 [0.07, 0.55]-point midpoint cost-only advantage with 32 calibration questions. The forecasts thus recover useful information at some, but not all, operating points.

4 Improving effort selection with token-cost predictions

4.1 Changing RADAR's costs, not its success predictions

We now test the complementary route to better effort choices: improving cost forecasts while retaining an existing selector's question-specific success predictions. This comparison isolates what cost prediction contributes without changing the predicted accuracy gain of any effort increase.

RADAR uses a logistic model of question difficulty and configuration ability to predict success, but assigns each configuration one average cost (Fernandez et al., 2026). For each target model separately, we let it choose only among that model's available reasoning settings. The baseline assigns a setting's average calibration cost to every question while our variant uses the question-specific cost forecasts from Section 3.4. Both use the same fitted RADAR success predictions, calibration responses, available settings, and selection criterion: the two-stage gain predictor above does not replace RADAR's success model. The additional cost predictor requires no extra answering-model calls on new evaluation questions. Appendix F.4 specifies the shared implementation and training adaptations.

At equal midpoint token use, the cost forecasts improve mean accuracy by 0.69 [0.58, 0.80] points under RADAR's Chebyshev criterion and by 0.38 points under its linear criterion. Refitting on 50 different calibration samples gives a mean Chebyshev improvement of 0.70 points, close to the original 0.69-point result (Appendix F.7). Matching accuracy instead of tokens gives an 8.0% token reduction at the midpoint of each target's fixed-setting accuracy range. Appendix G reports the full accuracy-matched comparison.

More high-effort answers for the same tokens. Within LiveCodeBench, cost forecasts improve GLM and Nemotron accuracy by 5.6 and 5.3 points at the same token expenditure. They increase the share of questions assigned high effort by 13.3 and 14.9 percentage points, respectively, by shifting high effort toward cheaper upgrades. For GLM, replacing average costs with question-specific forecasts makes high effort more likely on questions where it costs about 17,000 additional tokens, and less likely on questions where it costs about 37,000. The estimated mean accuracy gains are near 41 percentage points in both groups (Figure 2).

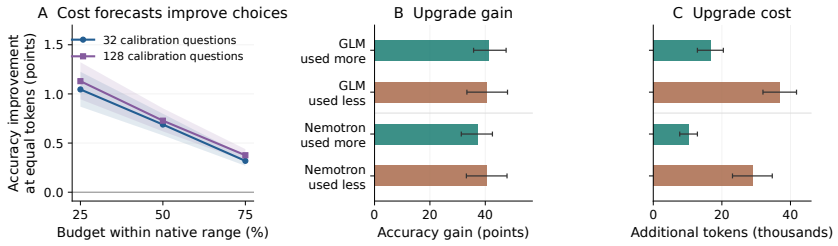


Figure 2: **Cost forecasts make more high-effort answers affordable.** A: accuracy improvement at equal observed tokens after replacing RADAR's average costs, with success predictions unchanged. B–C: measured gains and added tokens on LiveCodeBench for effort increases selected more or less often after replacement. Those selected more often are cheaper, with similar gain point estimates. Questions receive more weight when replacing the cost estimates changes their probability of receiving high effort by a larger amount. Shading and bars give paired 95% intervals.

4.2 A PROMPT-LENGTH MODEL CAPTURES MOST OF THE RADAR IMPROVEMENT

Setting averages and the prompt-length cost model use the same target responses to 32 calibration questions per benchmark. Instead of assigning every question the same mean cost, we fit a ridge regression for each effort setting to predict answer length from prompt length, both measured in tokens and transformed to a log scale. With RADAR's success predictions unchanged, it improves mean accuracy by 0.67 points at equal tokens, versus 0.69 for embedding-based costs (Table 41). This cost fit needs neither reference responses nor generation calls on new questions. Much of the observed benefit can therefore be obtained with a simple cost model. With benchmark-average gains in the earlier costs-only comparison, embedding-based costs instead improve accuracy by 0.62 points versus 0.18 for prompt length (Table 41, Figure 11).

Which models and benchmarks benefit? GLM and Nemotron account for most of the mean RADAR improvement as omitting them leaves 0.08 points. Separately omitting ZebraLogic and CogniLoad from evaluation also leaves about 0.08 points. For GPT-OSS, random mixtures of medium and high already match the best evaluated accuracy of either cost variant at the midpoint. The efficient medium setting is thus a strong alternative to finer selection for these targets. Table 4 reports every target.

4.3 CHOOSING SETTINGS UNDER BUDGETS FIXED BEFORE EVALUATION

Unlike the preceding comparisons, this test fixes the budget before evaluating answers. For each target and benchmark, calibration sets an average-token budget halfway between the estimated cheapest and most expensive settings. Given a batch of new questions, a linear optimization problem assigns setting probabilities to maximize predicted success within the predicted-cost budget. Variants share the budget and RADAR success predictions and differ only in costs. We then measure accuracy and actual tokens, which may fall below or exceed the planned budget.

Embedding-based costs increase mean accuracy by 1.15 points while reducing actual tokens by 177 per answer, or 3.6%, relative to setting-average costs (Table 3). The averages combine different target-level choices: GLM and Nemotron gain accuracy and spend more, while several targets lose accuracy and spend less. Prompt-length costs also improve mean accuracy, with different actual spending. Thus cost forecasts improve the aggregate tradeoff even when effort choices are committed before their outcomes are evaluated.

Table 3: **Cost forecasts improve the average accuracy–token tradeoff with budgets fixed in advance.** All cost models use the same RADAR success predictions and budgets estimated from 32 calibration questions per benchmark. The mean planned budget is 4,975 tokens. Changes are relative to setting-average costs [paired 95%].

Cost input	Accuracy (%)	Tokens	Accuracy change (pp)	Token change
Average costs	75.92	4,990	—	—
Prompt length	77.14	5,041	$+1.22\,[+0.97, +1.48]$	$+51\,[-100, +204]$
Length + workload	77.11	5,033	$+1.19\,[+0.93, +1.45]$	$+43\,[-111, +195]$
Embedding transfer	77.07	4,813	$+1.15\,[+0.85, +1.42]$	$-177\,[-366, -5]$

The same test applied to the transfer predictor adds gain forecasts while keeping costs and budgets fixed. It gains 0.27 accuracy points for 156 additional actual tokens with 32 calibration questions, or 0.35 points for 59 tokens with 128. Appendix G.5 reports uncertainty and all budgets. These are comparisons at the same planned budget, not equal realized expenditure.

5 LEARNING FROM EXAMPLES OR BUYING GUIDANCE

Text predictors learn from scored examples and then choose effort from the question alone. Alternatively, a selector can buy extra answers to each new question and use their agreement. We compare these approaches for a fixed Qwen 27B target, using frozen API prices applied to local generations.

Table 4: **Replacing average costs with question-specific forecasts: all nine targets.** Both variants use the same RADAR success predictions and 32 calibration questions per benchmark. Left: accuracy change at equal observed tokens under Chebyshev selection. Right: accuracy and actual-token changes under the same budgets fixed before evaluation, using a linear optimization problem. Brackets give paired 95% intervals.

	Matched observed tokens	Advance budget	
Target	Accuracy change (pp)	Accuracy change (pp)	Actual token change
GPT-OSS 20B	$+0.00\,[0.00, 0.00]$	$+0.68\,[0.35, 1.04]$	$-715\,[-1126, -323]$
GPT-OSS 120B	$+0.00\,[0.00, 0.00]$	$-0.53\,[-0.84, -0.20]$	$-574\,[-885, -295]$
Qwen 9B	$+0.59\,[0.41, 0.80]$	$+0.30\,[0.02, 0.57]$	$-350\,[-559, -151]$
Qwen 27B	$+0.06\,[-0.03, 0.15]$	$-0.10\,[-0.34, 0.15]$	$-360\,[-571, -152]$
GLM	$+2.00\,[1.58, 2.43]$	$+3.96\,[3.20, 4.68]$	$+494\,[184, 821]$
EXAONE	$-0.03\,[-0.12, 0.06]$	$-0.98\,[-1.54, -0.42]$	$-303\,[-500, -105]$
Nemotron	$+3.64\,[3.07, 4.19]$	$+7.65\,[6.64, 8.67]$	$+654\,[370, 932]$
Granite	$+0.15\,[0.02, 0.27]$	$+0.85\,[0.36, 1.34]$	$-190\,[-431, 48]$
Gemma	$-0.21\,[-0.31, -0.08]$	$-1.53\,[-1.92, -1.18]$	$-252\,[-338, -175]$
Mean	$+0.69\,[0.58, 0.80]$	$+1.15\,[0.85, 1.42]$	$-177\,[-366, -5]$

Left matches tokens after averaging benchmarks within each target; right fixes a midpoint budget separately for each target–benchmark pair. All native settings remain eligible. Table 35 gives absolute matched-token outcomes; Table 3 gives aggregate advance-budget outcomes.

Agreement guides effort without target calibration. We request four low-effort Qwen 9B answers, called *probes*. If all four final answers can be extracted and match after normalization, Qwen 27B answers at low effort; otherwise it uses high effort. The target receives only the original question and generates one new answer. The rule needs neither scored target examples nor correctness labels for the new question. It improves accuracy by 0.81 [0.55, 1.09] points over random assignment of high effort to the same fraction of questions within each benchmark. This comparison measures which questions receive high effort while the following comparisons also pay for the probes.

The probes can earn back their cost. Four-probe agreement reaches 87.57% accuracy at \$12.96 per 1,000 final answers, including every probe and target call. Always high reaches 87.99% at \$16.27: agreement saves 20.3% of the dollar bill for a 0.42-point accuracy loss. At the same \$12.96 budget, it exceeds the best tested learned selectors by 0.38 [0.14, 0.56] accuracy points when calibration is shared across 1,000 later answers per benchmark (Table 5). These savings replace expensive target generation with cheaper reference generation: combined output increases from about 10,356 to 17,307 tokens per final answer. They concern model-call dollars, not a measured reduction in hardware work. Stopping probe requests at the first invalid or disagreeing answer preserves the decision and increases the dollar saving to 22.9%, with sequential rather than parallel acquisition.

Calibration is paid once while probes are paid for every question. Let N be the number of later answers per benchmark that share one calibration bill. The main comparison treats previously collected reference responses as already available and charges target calibration. Its mean model-call cost per answer is

$$\underbrace{C_T}_{\text{target answer}} + \underbrace{C_P}_{\text{probes}} + \underbrace{C_{\text{cal}}/N}_{\text{shared calibration}} .$$

Here C_T and C_P are average generation bills, while C_{cal} is the one-time calibration bill. Holding the fitted predictors and evaluation responses fixed, increasing N makes calibration cheaper per answer. At the same total cost, learned selectors can then afford more accurate effort choices. Agreement's probe bill stays fixed.

At $N = 10{,}000$, agreement's advantage over the best tested learned selectors falls to 0.01 [-0.26, 0.24] accuracy points. Reusing purchased reference answers is already competitive at $N = 1{,}000$. Recurring probes can therefore improve on learning from examples when the setup bill is shared over fewer answers, while more reuse changes the best affordable alternative. Appendix F.8 additionally charges reference-response acquisition.

Table 5: **Calibration reuse changes the comparison with paid agreement.** Accuracy at $12.96 per 1,000 final answers, including probes and target calibration; previously collected reference responses are treated as available. N is the number of later answers *per benchmark* sharing the calibration bill. For each baseline, we report the highest evaluated accuracy at this cost among its tested selection rules and their allowed random mixtures. Differences subtract four-probe agreement [paired 95%].

	Calibration reused over $N = 1{,}000$ answers		Calibration reused over $N = 10{,}000$ answers	
Procedure	Accuracy (%)	Difference from agreement (pp)	Accuracy (%)	Difference from agreement (pp)
Four-probe agreement	87.57	—	87.57	—
Random low/high	86.96	$-0.61\,[-0.82, -0.41]$	86.96	$-0.61\,[-0.82, -0.41]$
32-question task default	87.14	$-0.43\,[-0.62, -0.21]$	87.27	$-0.30\,[-0.49, -0.10]$
128-question task default	86.93	$-0.64\,[-0.92, -0.31]$	87.56	$-0.01\,[-0.26, +0.26]$
Best learned-policy curve	87.19	$-0.38\,[-0.56, -0.14]$	87.56	$-0.01\,[-0.24, +0.26]$
Paid answer-length selection	87.12	$-0.46\,[-0.67, -0.22]$	87.35	$-0.22\,[-0.45, +0.03]$
Reference-answer reuse curve	87.59	$+0.01\,[-0.22, +0.23]$	87.59	$+0.01\,[-0.22, +0.23]$

Learned selection includes both RADAR variants, simple prompt-cost controls, transfer/direct predictors, and exhaustive task defaults. Reference-answer reuse includes mixtures with native target settings.
Paid answer-length selection incurs calibration and recurring-probe costs; in contrast, agreement, random effort, and reference-answer reuse have no setup bill.

These comparisons hold Qwen 27B fixed as the target. When another answering model is allowed, high-effort Gemma reaches 87.95% accuracy at $1.35 per 1,000 answers under the same prices. Model choice can outweigh effort selection. Appendix J reports other targets, early stopping, and price sensitivity.

6 Related work and Conclusion

Relation to prior work. Besides the related work mentioned in the introduction, FrugalGPT, Hybrid LLM, and RouteLLM choose among answering models (Chen et al., 2024; Ding et al., 2024; Ong et al., 2025). Length prediction supports serving schedules (Zheng et al., 2023), while TALE and L1 control reasoning length (Han et al., 2025; Aggarwal & Welleck, 2025). Weak-model consistency supports cascades, and adaptive consistency and early stopping reduce sampling (Yue et al., 2024; Aggarwal et al., 2023; Li et al., 2024). Deferral analysis distinguishes confidence from the benefit of an alternative (Jitkrittum et al., 2023). We use answer agreement to compare paying for current-question evidence with learning from reusable examples and returning purchased answers.

Conclusion. Knowing when to think matters as much as being able to think. Our results show that the opportunity is large: with the right information, adapting effort to each question saves nearly half of the generated tokens with almost no loss in accuracy. Today's practical selectors capture part of this opportunity, and better cost predictions already bring them closer. The rest lies in knowing where extra reasoning helps. Because these gains are specific to each model, the most promising source of this knowledge is the model itself. Our repeated responses measure the gain of every question for every model, so they can serve both as training targets for models that learn to regulate their own effort and as a benchmark for testing them. A robust solution that lets a model self-regulate its compute would directly shift the Pareto frontier between inference cost and general intelligence, and would open the way to agents that decide at each step how hard to think.

REPRODUCIBILITY STATEMENT

The supplied supplementary archive includes scored responses, reproduction code, saved outputs, and instructions for checking the paper's main results. These checks run on CPU without additional model generations. We will publicly release the full dataset with the camera-ready version.

AI USE STATEMENT

OpenAI Codex and Anthropic Claude Clode supported analysis implementation, numerical verification, figures, and manuscript editing.

The idea for this research was fully developed by humans. The structure of the paper was designed by humans and the first draft written by hand. Subsequent iterations on the paper used coding agents to polish and improve the wording and clarity of paragraphs. Final edits of the manuscript were done by humans.

The authors take full responsibility for the results and scientific claims in the paper.

REFERENCES

Pranjal Aggarwal and Sean Welleck. L1: Controlling How Long A Reasoning Model Thinks With Reinforcement Learning. In *Conference on Language Modeling*, 2025. URL https://arxiv.org/abs/2503.04697.

Pranjal Aggarwal, Aman Madaan, Yiming Yang, and Mausam. 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. doi: 10.18653/v1/2023.emnlp-main.761. URL https://aclanthology.org/2023.emnlp-main.761/.

Lingjiao Chen, Matei Zaharia, and James Zou. FrugalGPT: How to use large language models while reducing cost and improving performance. *Transactions on Machine Learning Research*, 2024. URL https://arxiv.org/abs/2305.05176.

Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, Christopher Hesse, and John Schulman. Training Verifiers to Solve Math Word Problems. *arXiv preprint arXiv:2110.14168*, 2021. URL https://arxiv.org/abs/2110.14168.

Mehul Damani, Idan Shenfeld, Andi Peng, Andreea Bobu, and Jacob Andreas. Learning How Hard to Think: Input-Adaptive Allocation of LM Computation. In *International Conference on Learning Representations*, 2025. URL https://proceedings.iclr.cc/paper_files/paper/2025/hash/ff414825df833edb8b1839e3d5d495e9-Abstract-Conference.html.

Dujian Ding, Ankur Mallick, Chi Wang, Robert Sim, Subhabrata Mukherjee, Victor Ruhle, Laks V. S. Lakshmanan, and Ahmed Hassan Awadallah. Hybrid LLM: Cost-efficient and quality-aware query routing. In *International Conference on Learning Representations*, 2024. URL https://proceedings.iclr.cc/paper_files/paper/2024/hash/b47d93c99fa22ac0b377578af0a1f63a-Abstract-Conference.html.

Nigel Fernandez, Branislav Kveton, Ryan A. Rossi, Andrew S. Lan, and Zichao Wang. RADAR: Reasoning-Ability and Difficulty-Aware Routing for Reasoning LLMs. In *International Conference on Learning Representations*, 2026. URL https://arxiv.org/abs/2509.25426.

Tingxu Han, Zhenting Wang, Chunrong Fang, Shiyu Zhao, Shiqing Ma, and Zhenyu Chen. Token-budget-aware LLM reasoning. In *Findings of the Association for Computational Linguistics: ACL 2025*, pp. 24842–24855. Association for Computational Linguistics, 2025. doi: 10.18653/v1/2025.findings-acl.1274. URL https://aclanthology.org/2025.findings-acl.1274/.

Dan Hendrycks, Collin Burns, Saurav Kadavath, Akul Arora, Steven Basart, Eric Tang, Dawn Song, and Jacob Steinhardt. Measuring Mathematical Problem Solving With the MATH Dataset. In *Proceedings of the Neural Information Processing Systems Track on Datasets and Benchmarks*, volume 1, 2021. URL https://datasets-benchmarks-proceedings.neurips.cc/paper_files/paper/2021/hash/be83ab3ecd0db773eb2dc1b0a17836a1-Abstract-round2.html.

Naman Jain, King Han, Alex Gu, Wen-Ding Li, Fanjia Yan, Tianjun Zhang, Sida I. Wang, Armando Solar-Lezama, Koushik Sen, and Ion Stoica. LiveCodeBench: Holistic and Contamination Free Evaluation of Large Language Models for Code. In *International Conference on Learning Representations*, 2025. URL https://proceedings.iclr.cc/paper_files/paper/2025/hash/94074dd5a072d28ff75a76dabed43767-Abstract-Conference.html.

Wittawat Jitkrittum, Neha Gupta, Aditya Krishna Menon, Harikrishna Narasimhan, Ankit Singh Rawat, and Sanjiv Kumar. When does confidence-based cascade deferral suffice? In *Advances in Neural Information Processing Systems*, 2023. URL https://arxiv.org/abs/2307.02764.

Daniel Kaiser, Arnoldo Frigessi, Ali Ramezani-Kebrya, and Benjamin Ricaud. CogniLoad: A Synthetic Natural Language Reasoning Benchmark With Tunable Length, Intrinsic Difficulty, and Distractor Density. *arXiv preprint arXiv:2509.18458*, 2025. URL https://arxiv.org/abs/2509.18458.

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*, 2024. URL https://arxiv.org/abs/2401.10480.

Bill Yuchen Lin, Ronan Le Bras, Kyle Richardson, Ashish Sabharwal, Radha Poovendran, Peter Clark, and Yejin Choi. ZebraLogic: On the scaling limits of LLMs for logical reasoning. In *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pp. 37889–37905. PMLR, 2025. URL https://proceedings.mlr.press/v267/lin25i.html.

Isaac Ong, Amjad Almahairi, Vincent Wu, Wei-Lin Chiang, Tianhao Wu, Joseph E. Gonzalez, M Waleed Kadous, and Ion Stoica. RouteLLM: Learning to route LLMs with preference data. In *International Conference on Learning Representations*, 2025. URL https://arxiv.org/abs/2406.18665.

OpenRouter. Provider endpoint listings and reasoning-token billing, 2026. URL https://openrouter.ai/docs/api/api-reference/endpoints/list-all-endpoints-for-a-model. Endpoint quotes captured 17 September 2026. Billing documentation: https://openrouter.ai/docs/guides/best-practices/reasoning-tokens.

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

Seamus Somerstep, Felipe Maia Polo, Allysson Flavio Melo de Oliveira, Prattyush Mangal, Mírian Silva, Onkar Bhardwaj, Mikhail Yurochkin, and Subha Maity. CARROT: A cost aware rate optimal router. *arXiv preprint arXiv:2502.03261*, 2025. URL https://arxiv.org/abs/2502.03261.

Zayne Sprague, Xi Ye, Kaj Bostrom, Swarat Chaudhuri, and Greg Durrett. MuSR: Testing the Limits of Chain-of-Thought with Multistep Soft Reasoning. In *International Conference on Learning Representations*, 2024. URL https://arxiv.org/abs/2310.16049.

Oyvind Tafjord, Bhavana Dalvi Mishra, and Peter Clark. ProofWriter: Generating implications, proofs, and abductive statements over natural language. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pp. 3621–3634. Association for Computational

Linguistics, 2021. doi: 10.18653/v1/2021.findings-acl.317. URL https://aclanthology.org/2021.findings-acl.317/.

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. In *Advances in Neural Information Processing Systems, Datasets and Benchmarks Track*, 2024. URL https://arxiv.org/abs/2406.01574.

Weihao Yu, Zihang Jiang, Yanfei Dong, and Jiashi Feng. ReClor: A Reading Comprehension Dataset Requiring Logical Reasoning. In *International Conference on Learning Representations*, 2020. URL https://arxiv.org/abs/2002.04326.

Murong Yue, Jie Zhao, Min Zhang, Liang Du, and Ziyu Yao. Large language model cascades with mixture of thought representations for cost-efficient reasoning. In *International Conference on Learning Representations*, 2024. URL https://arxiv.org/abs/2310.03094.

Zangwei Zheng, Xiaozhe Ren, Fuzhao Xue, Yang Luo, Xin Jiang, and Yang You. Response length perception and sequence scheduling: An LLM-empowered LLM inference pipeline. In *Advances in Neural Information Processing Systems*, volume 36, 2023. URL https://proceedings.neurips.cc/paper_files/paper/2023/hash/ce7ff3405c782f761fac7f849b41ae9a-Abstract-Conference.html.

GUIDE TO THE APPENDIX

Table 6 identifies the information and evaluation protocol behind each result. All active text forecasts use the fixed-source protocol in Appendix F.3, with feature and fitting definitions in Appendix E.2. Earlier forecast outputs with a different training partition remain in the artifact.

Table 6: **Which protocol supports each main result?** The entries distinguish different information tests from changes in calibration samples. All use the same recorded response bank.

Main result	Information and evaluation protocol	Specification
Cross-model patterns (Section 3)	Independent response halves estimate within-benchmark signal covariances; comparisons balance model-family pairs.	Appendices C–D
Earlier-answer value (Section 3.1)	One response half supplies gains and costs on each exact question; the other evaluates the original choices and four information restrictions.	Appendices E.1, F.1
Main text forecasts (Section 3.4)	Fixed source training; predictions for calibration questions exclude their own groups; 32/128 target questions; other groups evaluate choices.	Appendix F.3; features in E.2
RADAR cost change (Section 4.1)	One fitted success predictor, two cost modules, matched calibration, and evaluated policy tradeoff curves.	Appendix F.4
Stronger cost controls	Prompt length/workload fits, all calibration-derived default switching points, and budgets fixed before evaluating answers.	Appendix G
Calibration stability	Refit target procedures on 50 shared sample selections; retain source fits and evaluation questions.	Appendix F.7
Paid guidance (Section 5)	Original two/four-probe rule and independent target outcomes; charge acquired calls and compare learned-policy envelopes.	Appendices F.8, H, J

Appendix A specifies the response bank and scoring. Appendix B reports every native upgrade and its token cost. Appendices H–J give the agreement controls, acquisition sensitivities, and unsuccessful cases. Appendix K collects additional displays. The reproducibility statement and dated artifact specifications record the analysis sequence.

A STUDY CONSTRUCTION, MODELS AND SCORING

Table 7: **All nine models are evaluated on the same questions in all ten benchmarks.** Repeats are independent responses per question and effort setting. The 21 settings produce 597,744 responses. Shared questions are counted once.

Benchmark	What it tests	Questions	Repeats
GSM8K	Arithmetic word problems	256	8
MATH500	Competition mathematics	256	8
GPQA Diamond	Graduate science	198	8
MMLU-Pro	Knowledge across subjects	336	16
MuSR	Narrative reasoning	192	16
ReClor	Argument analysis	256	16
ProofWriter	Logical deduction	256	8
ZebraLogic	Constraint puzzles	256	8
CogniLoad	Tracking changing attributes	256	8
LiveCodeBench	Programming	256	16

The complete matrix contains 2,518 questions and 2,321 question groups. ProofWriter variants share 237 source groups for 256 questions. LiveCodeBench problems share 78 contests for 256 problems.

The other tasks use one group per selected question. Group assignments are fixed before calibration or bootstrap calculations.

GSM8K and MATH500 each contribute 256 fixed questions, while GPQA contributes all 198 Diamond questions. ProofWriter uses True/False/Unknown deduction under open-world semantics. ZebraLogic requires a complete grid satisfying its constraints. CogniLoad asks for attributes after a sequence of state updates. The latter two supply controlled problem structures rather than a sample of everyday requests. MMLU-Pro selects 24 questions in each of 14 subjects. MuSR selects 64 distinct stories in each of three subtasks. ReClor selects 256 distinct validation passages. LiveCodeBench selects 256 contest problems from the October 2024–April 2025 additions, proportional to published difficulty strata. Publication dates do not establish absence from training data.

Table 8: **Model versions and native settings.** Revision prefixes identify immutable model snapshots. The model lock file supplies full hashes, chat templates, tokenizer versions, and serving images.

Model	Provider	Settings	Temp.	Top-p	Revision
GPT-OSS 20B	openai	low/medium/high	1.0	1.00	6cee5e81ee
GPT-OSS 120B	openai	low/medium/high	1.0	1.00	b5c939de8f
Qwen 9B	Qwen	off/on	1.0	0.95	c202236235
Qwen 27B	Qwen	off/on	1.0	0.95	fc05daec18
GLM	zai-org	off/on	1.0	0.95	7dd20894a6
EXAONE	LGAI-EXAONE	off/on	0.6	0.95	a1d54d1c14
Nemotron	nvidia	off/on	1.0	0.95	bf77c3174f
Granite	ibm-granite	off/brief/on	1.0	0.95	f8de16cdcd
Gemma	google	off/on	1.0	0.95	842da3794e

Table 9: **Additional sampling settings.** -1 disables top-k truncation. GPT-OSS uses its reasoning-effort control. The other models use their native thinking control.

Model	Top-k	Presence penalty
GPT-OSS 20B	-1	0.0
GPT-OSS 120B	-1	0.0
Qwen 9B	20	1.5
Qwen 27B	20	1.5
GLM	-1	0.0
EXAONE	-1	0.0
Nemotron	-1	0.0
Granite	-1	0.0
Gemma	64	0.0

The common context ceiling is 98,304 tokens and the output ceiling is 81,920. Native stopping tokens and all generated token IDs are retained. Counts include reasoning, final answers, special tokens and failed outputs. They measure tokenizer-specific expenditure, not equal FLOPs, latency or monetary prices. Independent request identities include model revision, rendered input, effort setting, repeat and sampler. Infrastructure failures are recovered under the same identity and do not count as incorrect model answers.

Thinking off disables a native control, not every form of explanatory text. In particular, the CogniLoad prompt requests sequential updates, and Qwen off-mode can write long rule-by-rule state traces. The two Qwen sizes share the flag and sampler, but their lengths need not match. Granite also offers a brief setting. We establish its position relative to off and on from measured tokens rather than the names of the controls.

A.1 Answer extraction and grading

Extraction reads the final-answer channel without consulting the gold answer. For category tasks, we accept an explicit, unambiguous terminal choice. An answer containing incompatible terminal choices remains unrecognized. Mathematical answers use the task's terminal-answer rule followed by Math-Verify 0.8.0 with pinned dependencies. ProofWriter categories are compared under the

task's three-way semantics. ZebraLogic checks the complete extracted grid. CogniLoad checks the queried state with the benchmark scoring rules. A submitted wrong answer differs from a missing or uninterpretable final answer.

LiveCodeBench uses the last complete fenced program in the final channel and the official checker pinned at commit `28fef95ea8c9f7a547c8329f2cd3d32b92c1fa24`. Programs run against all supplied public/private tests, with six seconds per test and 2 GiB memory in a non-root, network-disabled container. Generation cannot read private tests. Compiler errors, program timeouts, and failed tests count as incorrect answers. Checker infrastructure failures are resolved before grading. Qualification fixtures cover passing and wrong answers, syntax, signals, timeouts and native stdout. Identical question/program keys reuse a determinate checker verdict. No coding verdict remains unresolved.

The same extraction and grading rules apply to every model. Missing or unrecognized answers count as incorrect. The artifact includes exact prompts, response identities, extraction code, and grading versions.

A.2 Output-limit rates

Table 10 reports how often high-effort generations reach the 81,920-token ceiling on each benchmark. The marginal-price analysis separately uses finish reasons and repeats its comparison on questions where every GPT-OSS answer finishes normally (Appendix B). Full accuracy, token, and cap-rate tables for every setting are in the artifact's `complete_matrix/results/` directory.

Table 10: **High-effort output-cap rates on every model and benchmark.** Percent of responses reaching 81,920 tokens. GPT columns distinguish 20B and 120B, and Qwen columns distinguish 9B and 27B. Zeroes are rounded to one decimal place.

Benchmark	GPT20	GPT120	Q9	Q27	GLM	EXA	Nem	Gran	Gem
GSM8K	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
MATH500	0.2	0.0	0.2	0.0	0.1	0.0	0.3	0.1	0.0
GPQA	7.8	0.3	0.7	0.4	0.3	0.1	4.8	0.1	0.0
MMLU-Pro	1.3	0.2	0.0	0.0	0.3	0.0	1.1	0.0	0.0
MuSR	0.0	0.0	0.0	0.0	0.0	0.0	0.1	0.0	0.0
ReClor	0.0	0.0	0.0	0.0	0.0	0.0	0.1	0.0	0.0
ProofWriter	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
ZebraLogic	6.3	2.3	1.0	1.0	0.2	0.0	10.4	0.3	0.0
CogniLoad	0.3	0.0	0.1	0.3	1.4	0.2	1.6	0.0	0.0
LiveCodeBench	17.2	5.2	6.3	4.7	15.8	1.3	5.3	0.1	0.0

B Accuracy returns and their token costs

The following analyses measure the return to each native effort choice, examine why the later GPT-OSS increase has a higher marginal price, and locate additional tokens and gains across questions. The artifact contains the specifications and independent numerical checks.

B.1 Accuracy return, remaining errors, and reciprocal prices

For an effort increase, let C be its mean additional tokens per answer, G its accuracy gain as a probability, and $E = 1 - p_{\text{start}}$ its starting error rate. The accuracy return is G/C. Expressed in accuracy points per 1,000 additional tokens, it is $100{,}000G/C$. This is the average return between two measured settings, not a local derivative of accuracy with respect to length. Returns retain negative gains and are undefined when added tokens are nonpositive.

We first average questions within each task and then the ten tasks equally, and only then form ratios. The error-rate reduction relative to remaining errors is $R = G/E$. This is a net rate change between fresh generations at two settings, not a paired count of particular wrong answers corrected. The corresponding return is R/C, reported as $100{,}000G/(EC)$ percent of starting errors removed per

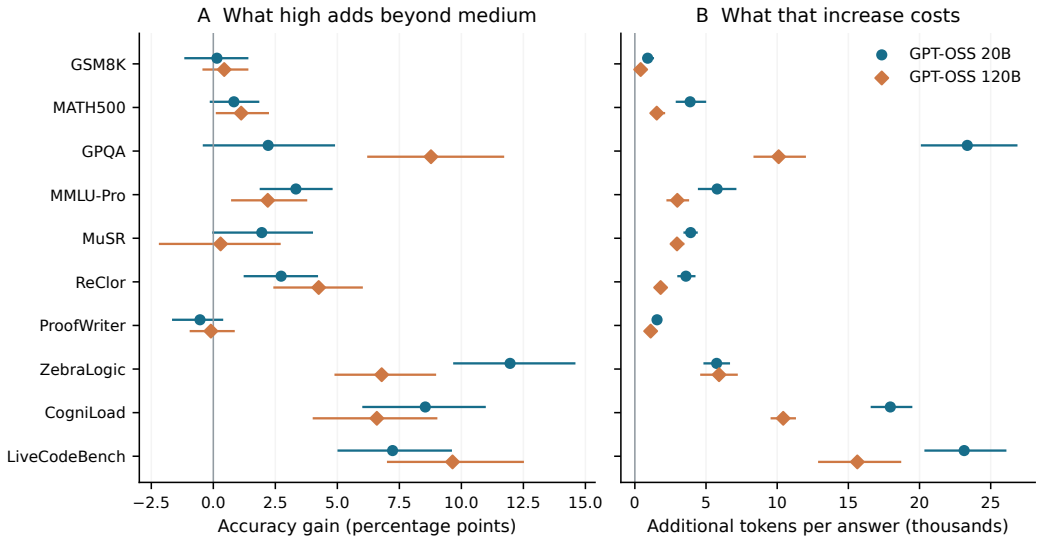


Figure 3: **An expensive final step can still be worth taking.** Medium-to-high accuracy gains and added tokens on all ten benchmarks, with paired 95% intervals. Coding and ZebraLogic retain substantial gains, while some other tasks gain little despite thousands of extra tokens. Both sizes use the same questions.

1,000 additional tokens. This also requires a positive starting error rate. The adjustment does not equate the difficulty of remaining errors or the native controls of different models.

The reciprocal view remains useful for expressing the cost of a correct answer. Over N questions, the increase costs NC tokens and produces NG net additional correct answers in expectation. Thus C/G is the extra token price per net additional correct answer when the gain is positive. It is not an expected retry-until-correct cost. Likewise, C/R is the reciprocal of the return relative to starting errors. The price tables below preserve these earlier summaries. The return summaries are recalculated from the same paired draws, without inverting rounded interval endpoints.

Table 11 gives starting accuracies, relative reductions, and relative returns for the choices in Table 65. Qwen 9B and 27B reduce their starting error rates by 23.98% [20.93, 26.87] and 24.47% [20.73, 28.07]. Per 1,000 added tokens, these are reductions of 3.06% [2.64, 3.49] and 3.07% [2.57, 3.56]. The 27B/9B accuracy-return ratio is 0.67 [0.55, 0.81], whereas the relative-return ratio is 1.00 [0.85, 1.19]. Similar pooled reductions do not establish equivalent performance or matching task profiles. For example, the relative reductions on LiveCodeBench are 19.4% and 39.8%, while those on ZebraLogic are 69.4% and 47.7%. The artifact retains all task-specific values and paired intervals.

For the first and second GPT-OSS increases,

$$\frac{C_2/G_2}{C_1/G_1} = \underbrace{\frac{C_2}{C_1}}_{\text{token increase}} \times \underbrace{\frac{E_1}{E_2}}_{\text{starting errors}} \times \underbrace{\frac{R_1}{R_2}}_{\text{relative reduction}} . \tag{3}$$

This is an accounting identity, not a causal decomposition. All three factors exceed one in the aggregate results (Table 13). The later accuracy return is 10.5% [8.6, 12.9] of the earlier return for 20B and 11.9% [9.8, 14.3] for 120B. After accounting for starting errors, the shares are 16.5% [13.5, 19.6] and 18.1% [15.2, 21.2]. Equivalently, the adjusted price ratios are 6.07 [5.10, 7.41] and 5.54 [4.71, 6.58]. Fewer errors left at medium therefore do not exhaust the measured decline in efficiency.

The intervals use 1,000 paired question-group bootstrap draws with common weights across models and settings. They condition on the response bank and model set. Recomputing each ratio within each draw preserves the dependence among its components and uses the same accuracy and token draws as the native-setting analysis. In the tables, L, M, and H denote low, medium, and high.

Table 11: **Starting accuracy, error reduction, and return per token.** Entries are percentages [paired 95%]. The final column is the percent of starting errors removed per 1,000 added tokens. Relative error reduction divides the equal-task gain by the equal-task starting error rate. Ratios are formed after averaging.

Model	Change	Starting accuracy	Relative error reduction	Relative return per 1,000 added tokens
GPT-OSS 20B	low → high	65.0 [63.6, 66.5]	47.0 [44.3, 49.7]	3.89 [3.54, 4.26]
GPT-OSS 120B	low → high	73.6 [72.2, 75.1]	49.3 [46.1, 52.3]	7.35 [6.63, 8.12]
Qwen 9B	off → on	76.2 [75.0, 77.3]	24.0 [20.9, 26.9]	3.06 [2.64, 3.49]
Qwen 27B	off → on	84.1 [83.0, 85.2]	24.5 [20.7, 28.1]	3.07 [2.57, 3.56]
GLM	off → on	52.5 [51.4, 53.6]	50.4 [48.4, 52.7]	5.93 [5.48, 6.44]
EXAONE	off → on	69.9 [68.5, 71.3]	34.3 [31.1, 37.4]	5.25 [4.70, 5.79]
Nemotron	off → on	42.1 [40.9, 43.5]	61.9 [60.0, 63.9]	7.99 [7.32, 8.74]
Granite	off → on	71.3 [69.8, 72.8]	15.0 [12.8, 17.4]	3.52 [2.94, 4.17]
Gemma	off → on	81.9 [80.5, 83.2]	33.4 [29.6, 37.3]	11.45 [10.07, 12.84]
GPT-OSS 20B	low → medium	65.0 [63.6, 66.5]	36.0 [33.6, 38.3]	11.59 [10.33, 12.98]
GPT-OSS 20B	medium → high	77.6 [76.4, 79.0]	17.2 [14.5, 19.8]	1.91 [1.58, 2.26]
GPT-OSS 120B	low → medium	73.6 [72.2, 75.1]	34.1 [31.5, 36.9]	24.09 [21.51, 26.92]
GPT-OSS 120B	medium → high	82.6 [81.4, 83.9]	23.0 [20.0, 26.3]	4.35 [3.70, 5.05]
Granite	brief → off	57.7 [56.1, 59.3]	32.1 [29.6, 34.7]	5.04 [4.57, 5.55]

Table 12: **Marginal accuracy and cost, with paired 95% intervals.** Errors removed are a percentage of the error rate at the earlier setting. The final column uses thousands of tokens per net additional correct answer.

GPT-OSS	Increase	Gain (pp)	Added tokens	Errors removed (%)	Price (1,000 tokens)
20B	L to M	12.59 [11.66, 13.44]	3,106 [2,861, 3,357]	36.0 [33.6, 38.3]	24.7 [22.2, 27.4]
20B	M to H	3.84 [3.24, 4.44]	8,981 [8,458, 9,490]	17.2 [14.5, 19.8]	233.9 [199.1, 281.8]
120B	L to M	8.99 [8.21, 9.82]	1,415 [1,319, 1,520]	34.1 [31.5, 36.9]	15.7 [14.2, 17.5]
120B	M to H	4.00 [3.35, 4.67]	5,288 [4,916, 5,683]	23.0 [20.0, 26.3]	132.2 [114.2, 154.6]

Table 13: **Three factors in the higher marginal price.** Each cell is a ratio [paired 95%]. Their pointwise product equals the price ratio in every draw.

GPT-OSS	C_2/C_1	E_1/E_2	R_1/R_2	Price ratio
20B	2.89 [2.70, 3.12]	1.56 [1.51, 1.62]	2.10 [1.79, 2.49]	9.48 [7.77, 11.65]
120B	3.74 [3.55, 3.92]	1.52 [1.46, 1.58]	1.48 [1.27, 1.75]	8.40 [6.99, 10.18]

B.2 Question groups within successive GPT-OSS increases

We group questions according to how often the model succeeds at the earlier setting. Even-numbered response attempts determine whether none, some, or all of these answers are correct, and odd-numbered attempts measure both settings' success rates and tokens. We then reverse the roles and average. Each selection half contains four answers per question on six tasks and eight on MMLU-Pro, MuSR, ReClor, and LiveCodeBench. Neither the later setting nor the evaluated gain determines a question's group.

Within each task, we sum a group's question weights, gains, and added tokens, then average across tasks. Dividing by the corresponding totals gives its shares of questions, gains, and tokens (Table 14). Dividing by the group's own question weight instead gives its independently measured success rates and mean tokens (Table 15). The label "all correct" describes the earlier sample and does not assert a true success probability of one. This analysis locates gains and costs in the recorded responses. Intervals hold the groups fixed and use common question weights so that the components continue to sum to their totals.

For medium to high, questions with some earlier correct answers account for 30% of questions in 20B and 21% in 120B. They supply about 69% of the net gain in each size while receiving 53% and 47% of the added tokens. This concentration concerns the second GPT-OSS step. The low/off-to-high/on comparisons below measure a different change.

Table 14: **Question groups, token shares, and gain shares.** All values are percentages [paired 95%]. Group labels refer only to the separate earlier-setting answers. Each set of three rows partitions its complete task-averaged question population, added tokens, and gain.

GPT-OSS	Increase	Selection answers	Questions	Added tokens	Accuracy gain
20B	L to M	None correct	18.2 [16.9, 19.7]	60.8 [56.5, 64.5]	34.8 [31.1, 38.8]
20B	L to M	Some correct	35.4 [33.7, 37.0]	30.8 [27.7, 34.6]	59.3 [55.4, 63.0]
20B	L to M	All correct	46.4 [44.8, 48.1]	8.4 [7.4, 9.5]	5.9 [4.1, 7.9]
20B	M to H	None correct	9.3 [8.2, 10.4]	23.7 [19.9, 27.5]	12.3 [7.5, 17.1]
20B	M to H	Some correct	29.8 [28.3, 31.3]	53.0 [49.5, 56.4]	69.0 [62.1, 76.1]
20B	M to H	All correct	60.9 [59.3, 62.5]	23.4 [21.2, 25.8]	18.7 [13.1, 23.9]
120B	L to M	None correct	14.6 [13.3, 15.9]	46.2 [41.8, 50.3]	36.7 [32.2, 41.4]
120B	L to M	Some correct	25.1 [23.7, 26.5]	37.7 [34.2, 41.4]	58.5 [54.1, 62.9]
120B	L to M	All correct	60.2 [58.5, 62.0]	16.1 [14.5, 17.8]	4.8 [2.3, 7.2]
120B	M to H	None correct	8.2 [7.2, 9.3]	23.6 [19.4, 28.0]	20.1 [14.7, 25.5]
120B	M to H	Some correct	21.0 [19.6, 22.3]	46.8 [43.4, 50.3]	68.6 [61.7, 75.5]
120B	M to H	All correct	70.8 [69.2, 72.5]	29.5 [26.7, 32.4]	11.3 [5.5, 16.7]

Table 15: **Independent outcomes within the same groups.** Success rates are percentages. Added tokens are measured per question in that group. Both settings are measured on answers excluded from group selection.

GPT-OSS	Increase	Selection answers	Earlier success	Later success	Added tokens
20B	L to M	None correct	7.2 [6.2, 8.4]	31.3 [28.4, 34.1]	10,373 [9,307, 11,495]
20B	L to M	Some correct	53.6 [52.0, 55.4]	74.7 [72.9, 76.5]	2,699 [2,490, 2,933]
20B	L to M	All correct	96.5 [96.0, 96.9]	98.1 [97.7, 98.5]	563 [522, 607]
20B	M to H	None correct	8.9 [7.4, 10.6]	14.0 [11.5, 16.6]	22,908 [19,904, 25,849]
20B	M to H	Some correct	59.3 [57.5, 61.2]	68.2 [66.2, 70.3]	15,949 [14,922, 17,042]
20B	M to H	All correct	97.1 [96.7, 97.4]	98.2 [97.9, 98.5]	3,445 [3,173, 3,718]
120B	L to M	None correct	6.1 [5.1, 7.2]	28.7 [25.6, 31.7]	4,464 [3,984, 4,992]
120B	L to M	Some correct	54.5 [52.5, 56.7]	75.4 [73.4, 77.5]	2,124 [1,967, 2,302]
120B	L to M	All correct	98.0 [97.8, 98.3]	98.8 [98.4, 99.0]	378 [356, 401]
120B	M to H	None correct	6.3 [5.0, 7.8]	16.1 [13.2, 19.2]	15,176 [12,530, 17,984]
120B	M to H	Some correct	60.1 [57.8, 62.2]	73.1 [70.6, 75.5]	11,793 [10,932, 12,767]
120B	M to H	All correct	98.2 [97.9, 98.4]	98.8 [98.5, 99.1]	2,207 [2,075, 2,344]

B.3 Where native upgrades pay off across all nine models

We apply the same independent-half grouping to every model's native low/off-to-high/on change. Granite's lower setting is off in this analysis. Only the lower-setting classification half determines whether none, some, or all answers are correct. The opposite half supplies both settings' outcomes, and we average the two orientations. All models and settings share the same question-group bootstrap weights, with 1,000 draws and seed $2026091623 + t$ for task index t starting at zero.

Tables 16 and 17 give the complete groups underlying Figure 16. Shares divide task-balanced contributions by task-balanced totals; they do not average within-task shares. Group masses sum to one, and the signed gain and token contributions sum to the original native-setting changes within every task and bootstrap draw. A separate implementation reconstructs group membership from raw response counts and bootstraps sums over question groups. It reproduces all 1,001 point/bootstrap entries for every model, including the previously saved native accuracy and token estimates.

Qwen 27B's earlier-all-correct group receives 51.30% [48.60, 54.11] of its added tokens and contributes 5.54% [-0.94, 11.06] of its net gain. Its independently measured lower-setting accuracy is 98.42% [98.19, 98.66]. It contains 7.31% [6.09, 8.54] of starting errors, calculated from its

task-balanced lower-setting error contribution divided by the total. Gemma's corresponding token and gain shares are 49.05% [46.41, 51.69] and 1.27% [−2.80, 5.05], with 4.67% [3.85, 5.66] of starting errors. These groups have little remaining accuracy to gain, but the uncertainty does not establish zero gain. EXAONE and Granite have negative net gains in this group, which remain signed. A gain share can therefore be negative or exceed 100%. The groups use earlier scored answers to describe where spending pays off; they do not specify a label-free selection rule or identify a removable token budget.

Table 16: **All three question groups across the nine native upgrades.** Entries are percentages [paired 95%]. None, Some, and All refer to correct answers in the separate lower-setting sample. Each model's three rows partition its question population, additional tokens, and net accuracy gain after equal-task averaging.

Model	Correct earlier	Questions	Added tokens	Net gain
GPT-OSS 20B	None	18.2 [16.9, 19.7]	47.3 [43.8, 50.8]	39.8 [36.1, 43.4]
GPT-OSS 20B	Some	35.4 [33.7, 37.0]	42.7 [39.4, 45.9]	55.0 [51.5, 58.5]
GPT-OSS 20B	All	46.4 [44.8, 48.1]	10.1 [8.9, 11.4]	5.1 [3.6, 6.8]
GPT-OSS 120B	None	14.6 [13.3, 15.9]	44.2 [39.5, 48.7]	40.0 [35.9, 44.1]
GPT-OSS 120B	Some	25.1 [23.7, 26.5]	38.8 [35.2, 42.8]	56.8 [52.7, 61.1]
GPT-OSS 120B	All	60.2 [58.5, 62.0]	16.9 [15.3, 18.8]	3.2 [1.2, 5.1]
Qwen 9B	None	10.0 [8.9, 11.0]	22.5 [19.0, 26.2]	16.1 [12.1, 20.5]
Qwen 9B	Some	30.6 [29.1, 32.1]	37.3 [34.7, 40.1]	68.7 [63.1, 74.0]
Qwen 9B	All	59.4 [57.8, 60.9]	40.2 [37.5, 42.9]	15.1 [11.2, 19.3]
Qwen 27B	None	8.0 [7.0, 9.1]	16.5 [13.5, 19.6]	27.3 [20.9, 34.2]
Qwen 27B	Some	18.2 [17.0, 19.5]	32.2 [29.6, 34.9]	67.2 [59.7, 74.6]
Qwen 27B	All	73.7 [72.3, 75.1]	51.3 [48.6, 54.1]	5.5 [-0.9, 11.1]
GLM	None	24.0 [22.8, 25.2]	56.8 [53.9, 59.9]	42.5 [39.8, 45.3]
GLM	Some	47.7 [46.2, 49.2]	38.6 [35.6, 41.4]	52.3 [49.4, 55.0]
GLM	All	28.3 [27.1, 29.5]	4.6 [3.8, 5.4]	5.2 [4.2, 6.3]
EXAONE	None	18.7 [17.4, 20.1]	39.6 [36.7, 42.7]	57.7 [52.4, 63.5]
EXAONE	Some	24.2 [22.7, 25.6]	28.3 [25.8, 30.7]	50.6 [45.2, 55.9]
EXAONE	All	57.1 [55.3, 58.7]	32.1 [29.9, 34.5]	-8.3 [-12.7, -4.3]
Nemotron	None	32.7 [31.3, 34.1]	63.3 [59.8, 66.5]	48.0 [45.9, 50.3]
Nemotron	Some	48.9 [47.3, 50.3]	32.9 [29.8, 36.1]	48.7 [46.4, 50.9]
Nemotron	All	18.5 [17.2, 19.8]	3.8 [3.1, 4.8]	3.2 [2.4, 4.0]
Granite	None	18.0 [16.5, 19.4]	40.0 [36.0, 44.1]	42.1 [34.8, 49.7]
Granite	Some	23.9 [22.5, 25.4]	32.9 [29.7, 36.3]	64.1 [56.2, 72.3]
Granite	All	58.1 [56.5, 59.8]	27.1 [24.2, 30.2]	-6.2 [-14.3, 0.3]
Gemma	None	11.0 [9.8, 12.2]	19.6 [17.2, 22.0]	40.8 [35.0, 46.4]
Gemma	Some	16.2 [15.0, 17.4]	31.4 [29.1, 33.6]	57.9 [52.2, 64.1]
Gemma	All	72.8 [71.4, 74.4]	49.1 [46.4, 51.7]	1.3 [-2.8, 5.1]

B.4 Long answers and completion

For a response of length T, tokens in the band $(a, b]$ equal $\min\{\max(T-a, 0), b-a\}$. Bands partition every response's tokens. The difference in each band's expected tokens between settings therefore partitions the full added bill. Tokens beyond position 16,384 account for 60% and 45% of the medium-to-high increase for 20B and 120B. Table 18 reports the fixed 8,192-, 16,384-, and 32,768-token thresholds. No response is rescored as though it had stopped earlier. This accounting locates expenditure, but does not identify the effect of a tighter output limit or an internal reasoning operation.

Table 19 separately partitions the extra bill by stop status and correctness. Negative changes are retained: normally stopped, unrecognized answers consume fewer tokens at high than at medium. Tokens from correct answers are not automatically unnecessary, since the frequency of correct answers also changes. High-effort length stops occur in 3.31% [2.66, 3.99] of 20B responses and 0.80% [0.51, 1.11] of 120B responses under equal task weights. Medium rates are 0.16% and zero, respectively. Length-stopped responses contribute 28.77% [23.71, 33.51] and 12.41% [8.37, 16.24] of the medium-to-high token increase.

Table 17: **Independent outcomes within the all-model question groups.** Both settings are evaluated on responses excluded from group membership. Accuracy is in percent and added tokens are per question in that group, with paired 95% intervals. Lower and higher mean low/off and high/on, respectively.

Model	Correct earlier	Lower accuracy	Higher accuracy	Added tokens
GPT-OSS 20B	None	7.2 [6.2, 8.4]	43.2 [39.9, 46.5]	31,369 [29,130, 33,846]
GPT-OSS 20B	Some	53.6 [52.0, 55.4]	79.1 [77.2, 81.1]	14,561 [13,577, 15,496]
GPT-OSS 20B	All	96.5 [96.0, 96.9]	98.3 [97.9, 98.7]	2,623 [2,396, 2,882]
GPT-OSS 120B	None	6.1 [5.1, 7.2]	41.6 [38.0, 45.1]	20,242 [17,923, 22,592]
GPT-OSS 120B	Some	54.5 [52.5, 56.7]	83.8 [81.8, 85.6]	10,355 [9,718, 11,084]
GPT-OSS 120B	All	98.0 [97.8, 98.3]	98.7 [98.4, 99.1]	1,886 [1,768, 2,008]
Qwen 9B	None	7.4 [6.1, 8.9]	16.6 [13.8, 19.7]	17,668 [15,030, 20,398]
Qwen 9B	Some	58.2 [56.5, 59.8]	71.0 [68.6, 73.1]	9,543 [8,942, 10,136]
Qwen 9B	All	97.0 [96.7, 97.3]	98.4 [98.1, 98.7]	5,292 [5,060, 5,548]
Qwen 27B	None	5.2 [3.9, 6.6]	18.4 [15.0, 22.6]	16,389 [13,846, 19,015]
Qwen 27B	Some	60.9 [58.6, 63.3]	75.2 [72.5, 77.8]	14,109 [13,185, 15,095]
Qwen 27B	All	98.4 [98.2, 98.7]	98.7 [98.4, 99.0]	5,555 [5,372, 5,741]
GLM	None	7.4 [6.5, 8.2]	49.8 [47.0, 52.7]	20,191 [18,749, 21,689]
GLM	Some	51.0 [49.6, 52.4]	77.2 [75.7, 78.8]	6,885 [6,388, 7,368]
GLM	All	93.4 [92.6, 94.2]	97.8 [97.1, 98.4]	1,377 [1,177, 1,582]
EXAONE	None	4.9 [4.1, 5.8]	36.7 [33.3, 39.7]	13,812 [12,971, 14,834]
EXAONE	Some	54.6 [52.7, 56.7]	76.2 [74.0, 78.6]	7,628 [7,105, 8,171]
EXAONE	All	97.7 [97.4, 98.0]	96.2 [95.5, 96.8]	3,676 [3,473, 3,887]
Nemotron	None	7.1 [6.3, 7.9]	59.8 [57.3, 62.2]	15,036 [13,669, 16,432]
Nemotron	Some	47.2 [45.8, 48.6]	83.0 [81.3, 84.6]	5,215 [4,712, 5,779]
Nemotron	All	90.6 [89.2, 91.8]	96.8 [95.8, 97.8]	1,599 [1,307, 1,954]
Granite	None	4.2 [3.4, 5.1]	14.2 [12.4, 16.2]	9,460 [8,500, 10,450]
Granite	Some	56.4 [54.3, 58.5]	67.9 [65.4, 70.3]	5,838 [5,332, 6,396]
Granite	All	98.1 [97.9, 98.4]	97.7 [97.2, 98.1]	1,984 [1,796, 2,178]
Gemma	None	5.3 [4.2, 6.5]	27.8 [24.2, 31.6]	5,202 [4,714, 5,667]
Gemma	Some	57.8 [55.5, 60.1]	79.4 [76.6, 81.9]	5,646 [5,335, 5,954]
Gemma	All	98.8 [98.6, 99.0]	98.9 [98.6, 99.2]	1,967 [1,896, 2,043]

Table 18: **The share of added tokens beyond fixed length thresholds.** Percentages [paired 95%] measure the increase in tail tokens divided by the entire token increase. They are not the fraction of answers exceeding that length.

GPT-OSS	Increase	Beyond token	Share of added tokens (%)
20B	L to M	8,192	47.0 [43.5, 50.1]
20B	L to M	16,384	27.3 [23.7, 30.4]
20B	L to M	32,768	11.7 [9.2, 14.1]
20B	M to H	8,192	77.8 [76.2, 79.2]
20B	M to H	16,384	59.6 [57.2, 61.8]
20B	M to H	32,768	33.9 [31.2, 36.3]
120B	L to M	8,192	24.9 [22.1, 27.4]
120B	L to M	16,384	4.5 [3.6, 5.5]
120B	L to M	32,768	0.0 [0.0, 0.0]
120B	M to H	8,192	67.0 [64.4, 69.6]
120B	M to H	16,384	45.3 [41.7, 48.8]
120B	M to H	32,768	20.2 [16.9, 23.3]

For a common completion sensitivity, we retain a question only if every recorded answer for both GPT-OSS sizes at all three settings stops normally. This retains 92.42% [91.30, 93.49] of equal-task question mass. The later accuracy return is 9.7% [7.9, 11.7] of the earlier return for 20B and 9.5% [7.4, 12.0] for 120B. The reciprocal price ratios are 10.36 [8.54, 12.69] and 10.51 [8.37, 13.53]. Within this common population, marginal prices are 14,735 versus 152,588 tokens for 20B and 11,597 versus 121,932 for 120B. The decline in return therefore persists among questions whose recorded

Table 19: **Completion and correctness partition the medium-to-high token increase.** Added tokens are contributions per question in the full population. Shares are percentages with paired 95% intervals. Signed components sum to the total bill.

GPT-OSS	Response category	Added tokens	Share (%)
20B	Finished, correct	5,177 [4,933, 5,455]	57.6 [54.1, 61.6]
20B	Finished, wrong answer	1,432 [1,197, 1,703]	15.9 [13.4, 18.7]
20B	Finished, no recognized answer	-211 [-337, -99]	-2.3 [-3.7, -1.1]
20B	Length stopped, correct	0 [0, 0]	0.0 [0.0, 0.0]
20B	Length stopped, incorrect	2,584 [2,059, 3,128]	28.8 [23.7, 33.5]
120B	Finished, correct	3,599 [3,412, 3,826]	68.1 [64.3, 72.3]
120B	Finished, wrong answer	1,069 [902, 1,259]	20.2 [17.2, 23.3]
120B	Finished, no recognized answer	-36 [-70, -3]	-0.7 [-1.3, -0.1]
120B	Length stopped, correct	2 [0, 6]	0.0 [0.0, 0.1]
120B	Length stopped, incorrect	654 [419, 909]	12.4 [8.4, 16.2]

answers all finish. Because this restriction changes the question population, it does not estimate the causal effect of removing the token cap.

B.5 Returns across models and tasks

Figure 3 shows the final GPT-OSS step on every benchmark. Table 20 broadens the comparison to all nine models' low-to-high changes. Ratios are calculated after equal-task averaging. When a gain is nonpositive, its marginal price is left undefined in the saved per-task tables. The aggregate GPT-OSS prices above are defined in every draw.

Table 20: **Endpoint prices across all nine models.** Each comparison uses native low/off versus high/on. Price is measured in thousands of tokenizer-specific tokens per net additional correct answer. Monetary and hardware costs are not equated across models.

Model	Gain (pp)	Added tokens	Price (1,000 tokens)
GPT-OSS 20B	16.43 [15.39, 17.43]	12,088 [11,437, 12,745]	74 [68, 80]
GPT-OSS 120B	12.99 [11.99, 14.00]	6,704 [6,261, 7,176]	52 [47, 57]
Qwen3.5-9B	5.72 [4.94, 6.40]	7,828 [7,444, 8,207]	137 [122, 159]
Qwen3.5-27B	3.89 [3.29, 4.55]	7,984 [7,676, 8,307]	205 [176, 244]
GLM-4.7-Flash	23.93 [22.83, 25.03]	8,511 [8,054, 9,012]	36 [33, 38]
EXAONE-4.0-32B	10.33 [9.23, 11.40]	6,532 [6,311, 6,771]	63 [57, 71]
Nemotron-3-Nano	35.83 [34.48, 37.10]	7,754 [7,195, 8,285]	22 [20, 24]
Granite-4.2-8B	4.30 [3.64, 5.07]	4,248 [4,019, 4,476]	99 [83, 118]
Gemma-4-31B-it	6.05 [5.25, 6.84]	2,920 [2,828, 3,012]	48 [43, 56]

Granite's brief setting is cheaper and less accurate than off. For its named brief-to-off and off-to-on choices, the corresponding prices are 46,883 [42,661, 51,900] and 98,849 [83,043, 118,407] tokens. Brief-to-off raises accuracy by 13.56 [12.32, 14.76] points for 6,357 [6,073, 6,638] additional tokens. We do not equate those controls with the GPT-OSS low/medium/high sequence. The saved results include all three pairwise Granite comparisons. Figure 4 shows how the low-to-high accuracy gain varies across all nine models and ten tasks.

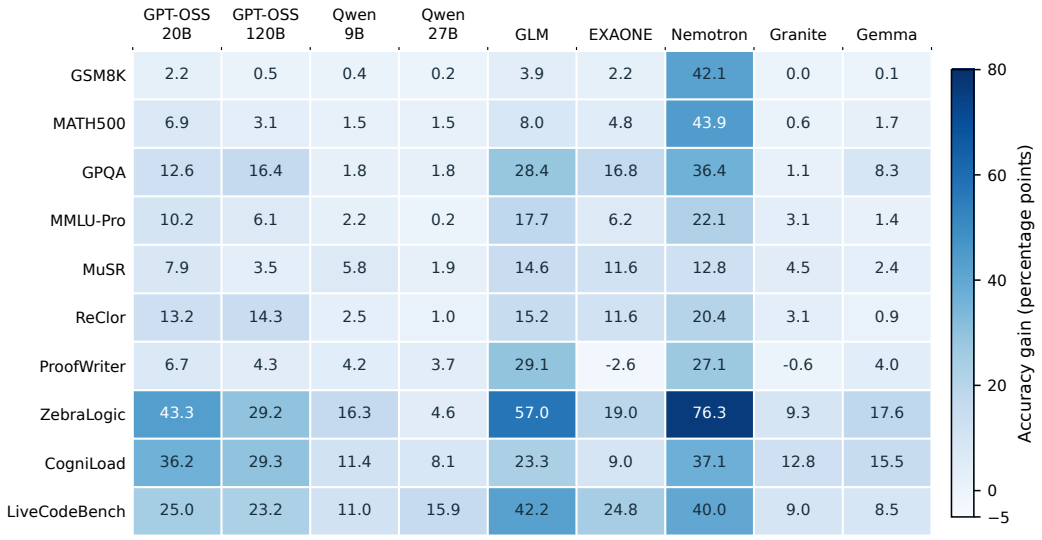


Figure 4: **Where high effort adds accuracy across the complete study.** Each cell is the high/on minus low/off accuracy difference on the same questions, in percentage points. Negative values are retained. The numbers are complete-benchmark estimates, not a selected set of benefiting questions. This map locates the gains; the native-setting comparison in Figure 15 and the successive-step comparison in Figure 3 show their token costs.

Table 21 reports gains and added tokens directly for every model–task comparison. This avoids magnifying small or uncertain gains through a price ratio. The artifact retains full-precision prices, undefined comparisons, and all per-task question-group outcomes in `all_model_effort_returns_20260918/results/`.

Table 21: **The native upgrade on every model and task.** Low/off to high/on accuracy gains and additional generated tokens per answer, with paired 95% intervals. Granite compares off with on.

Model	Task	Gain (pp)	Added tokens
GPT-OSS 20B	GSM8K	2.20 [0.49, 3.96]	1,177 [868, 1,576]
GPT-OSS 20B	MATH500	6.88 [4.88, 9.23]	4,981 [3,823, 6,380]
GPT-OSS 20B	GPQA	12.63 [8.27, 17.05]	26,806 [23,212, 30,763]
GPT-OSS 20B	MMLU-Pro	10.25 [7.66, 12.74]	6,519 [5,162, 7,856]
GPT-OSS 20B	MuSR	7.88 [4.17, 11.43]	4,570 [4,118, 5,007]
GPT-OSS 20B	ReClor	13.23 [9.50, 16.72]	4,213 [3,643, 4,831]
GPT-OSS 20B	ProofWriter	6.74 [4.22, 9.69]	2,135 [1,905, 2,349]
GPT-OSS 20B	ZebraLogic	43.26 [39.26, 47.32]	16,365 [13,878, 18,756]
GPT-OSS 20B	CogniLoad	36.23 [32.08, 39.84]	23,583 [21,851, 25,545]
GPT-OSS 20B	LiveCodeBench	25.02 [21.41, 29.11]	30,529 [26,701, 34,843]
GPT-OSS 120B	GSM8K	0.54 [-0.73, 1.90]	502 [428, 592]
GPT-OSS 120B	MATH500	3.08 [1.61, 4.88]	1,962 [1,485, 2,566]
GPT-OSS 120B	GPQA	16.35 [12.37, 20.52]	11,407 [9,577, 13,390]
GPT-OSS 120B	MMLU-Pro	6.12 [3.89, 8.48]	3,371 [2,629, 4,181]
GPT-OSS 120B	MuSR	3.52 [-0.26, 7.00]	3,268 [2,866, 3,713]
GPT-OSS 120B	ReClor	14.28 [10.81, 17.60]	2,133 [1,889, 2,393]
GPT-OSS 120B	ProofWriter	4.35 [2.02, 6.86]	1,463 [1,335, 1,585]
GPT-OSS 120B	ZebraLogic	29.20 [24.95, 33.25]	9,448 [7,612, 11,187]
GPT-OSS 120B	CogniLoad	29.30 [25.98, 33.35]	14,927 [13,673, 16,192]
GPT-OSS 120B	LiveCodeBench	23.19 [18.82, 27.54]	18,558 [15,395, 22,167]
Qwen 9B	GSM8K	0.44 [-0.73, 1.66]	3,594 [3,226, 4,030]
Qwen 9B	MATH500	1.51 [0.29, 2.78]	7,548 [6,660, 8,557]
Qwen 9B	GPQA	1.77 [-0.82, 4.23]	16,478 [14,950, 18,142]

Continued on next page

Table 21 continued.

Model	Task	Gain (pp)	Added tokens
Qwen 9B	MMLU-Pro	2.19 [0.73, 3.74]	5,776 [5,110, 6,446]
Qwen 9B	MuSR	5.83 [2.47, 9.31]	3,440 [3,262, 3,615]
Qwen 9B	ReClor	2.49 [1.27, 3.71]	2,637 [2,452, 2,818]
Qwen 9B	ProofWriter	4.25 [1.88, 6.72]	3,874 [3,681, 4,062]
Qwen 9B	ZebraLogic	16.31 [13.04, 19.29]	6,981 [6,001, 7,932]
Qwen 9B	CogniLoad	11.38 [8.40, 14.16]	10,868 [10,267, 11,511]
Qwen 9B	LiveCodeBench	10.99 [7.76, 13.72]	17,087 [14,238, 20,110]
Qwen 27B	GSM8K	0.24 [-1.07, 1.56]	2,573 [2,226, 2,920]
Qwen 27B	MATH500	1.46 [0.05, 3.03]	5,274 [4,632, 5,934]
Qwen 27B	GPQA	1.77 [-0.51, 4.23]	15,242 [13,956, 16,772]
Qwen 27B	MMLU-Pro	0.20 [-1.12, 1.54]	6,079 [5,503, 6,689]
Qwen 27B	MuSR	1.92 [-0.81, 4.49]	2,855 [2,714, 2,986]
Qwen 27B	ReClor	0.95 [0.17, 2.00]	2,372 [2,143, 2,573]
Qwen 27B	ProofWriter	3.71 [1.64, 5.90]	3,254 [3,044, 3,467]
Qwen 27B	ZebraLogic	4.59 [2.34, 7.03]	7,471 [6,675, 8,349]
Qwen 27B	CogniLoad	8.11 [6.05, 10.21]	14,691 [13,983, 15,399]
Qwen 27B	LiveCodeBench	15.94 [13.26, 18.90]	20,029 [17,837, 22,546]
GLM	GSM8K	3.86 [1.95, 5.86]	842 [623, 1,109]
GLM	MATH500	8.01 [5.81, 10.60]	1,711 [1,137, 2,429]
GLM	GPQA	28.41 [24.24, 32.83]	18,277 [15,988, 20,849]
GLM	MMLU-Pro	17.69 [14.58, 20.63]	5,783 [4,702, 6,867]
GLM	MuSR	14.55 [10.48, 18.52]	2,892 [2,703, 3,098]
GLM	ReClor	15.19 [12.62, 17.80]	2,217 [1,970, 2,477]
GLM	ProofWriter	29.05 [25.43, 33.06]	2,699 [2,505, 2,897]
GLM	ZebraLogic	57.03 [53.22, 61.08]	11,963 [10,608, 13,316]
GLM	CogniLoad	23.29 [19.43, 27.05]	12,016 [10,881, 13,238]
GLM	LiveCodeBench	42.24 [36.98, 47.23]	26,712 [23,407, 30,195]
EXAONE	GSM8K	2.20 [0.20, 4.49]	2,028 [1,812, 2,261]
EXAONE	MATH500	4.83 [3.03, 6.98]	5,109 [4,479, 5,816]
EXAONE	GPQA	16.79 [11.87, 21.53]	12,744 [11,577, 13,957]
EXAONE	MMLU-Pro	6.21 [3.27, 9.06]	4,848 [4,117, 5,521]
EXAONE	MuSR	11.56 [7.26, 16.11]	2,531 [2,398, 2,662]
EXAONE	ReClor	11.57 [8.30, 15.02]	1,748 [1,618, 1,885]
EXAONE	ProofWriter	-2.64 [-5.73, 0.63]	2,563 [2,408, 2,720]
EXAONE	ZebraLogic	19.04 [15.53, 22.36]	5,888 [5,243, 6,533]
EXAONE	CogniLoad	8.98 [4.59, 13.67]	9,772 [9,037, 10,483]
EXAONE	LiveCodeBench	24.76 [21.23, 28.67]	18,091 [16,736, 19,573]
Nemotron	GSM8K	42.09 [38.57, 45.80]	723 [508, 998]
Nemotron	MATH500	43.95 [39.84, 48.05]	3,145 [2,278, 4,204]
Nemotron	GPQA	36.36 [30.93, 41.54]	19,833 [17,190, 22,784]
Nemotron	MMLU-Pro	22.12 [18.77, 25.50]	4,959 [3,871, 6,151]
Nemotron	MuSR	12.83 [8.76, 17.09]	1,277 [1,056, 1,534]
Nemotron	ReClor	20.41 [16.70, 24.27]	2,299 [1,958, 2,645]
Nemotron	ProofWriter	27.15 [22.64, 31.27]	608 [443, 764]
Nemotron	ZebraLogic	76.32 [72.80, 80.13]	16,133 [13,358, 18,760]
Nemotron	CogniLoad	37.11 [31.93, 41.85]	10,127 [9,216, 11,195]
Nemotron	LiveCodeBench	39.99 [35.87, 44.18]	18,433 [15,025, 22,003]
Granite	GSM8K	0.00 [-1.12, 1.22]	1,445 [1,267, 1,630]
Granite	MATH500	0.59 [-0.54, 1.66]	1,203 [625, 1,857]
Granite	GPQA	1.14 [-1.01, 3.35]	6,225 [5,610, 6,840]
Granite	MMLU-Pro	3.09 [1.21, 4.71]	3,177 [2,812, 3,547]
Granite	MuSR	4.52 [1.37, 7.94]	1,687 [1,572, 1,809]
Granite	ReClor	3.10 [1.10, 5.35]	2,140 [1,888, 2,416]
Granite	ProofWriter	-0.59 [-1.77, 0.49]	2,005 [1,803, 2,218]
Granite	ZebraLogic	9.33 [6.49, 12.06]	-2,572 [-3,408, -1,795]
Granite	CogniLoad	12.79 [9.03, 16.46]	16,277 [14,711, 17,909]
Granite	LiveCodeBench	9.01 [6.48, 11.29]	10,895 [9,842, 12,024]
Gemma	GSM8K	0.15 [-0.39, 0.78]	948 [826, 1,091]
Gemma	MATH500	1.71 [0.63, 2.93]	1,726 [1,503, 1,975]
Gemma	GPQA	8.33 [4.67, 11.74]	4,632 [4,279, 5,006]
Gemma	MMLU-Pro	1.38 [-0.13, 2.81]	2,681 [2,434, 2,926]

Continued on next page

Table 21 continued.

Model	Task	Gain (pp)	Added tokens
Gemma	MuSR	2.44 [-1.24, 6.22]	1,786 [1,639, 1,939]
Gemma	ReClor	0.88 [0.24, 1.81]	1,073 [978, 1,179]
Gemma	ProofWriter	4.00 [1.51, 6.50]	1,227 [1,113, 1,340]
Gemma	ZebraLogic	17.58 [14.35, 21.09]	2,161 [1,831, 2,470]
Gemma	CogniLoad	15.53 [12.69, 18.41]	7,426 [6,821, 8,099]
Gemma	LiveCodeBench	8.52 [6.09, 11.13]	5,535 [5,276, 5,805]

C SIGNAL CORRELATIONS AND MEASUREMENT RELIABILITY

In addition to success rates p and mean tokens μ, we compare mean log tokens $c = \mathbb{E}[\log(1+T)]$, where T is one response's generated-token count. Log expansion is $c_H - c_L$, while absolute added tokens are $\mu_H - \mu_L$. These scales ask different questions about the same effort increase.

Table 22: **High-effort levels are more shared than accuracy gains.** Ten-task pooled signal correlations with 95% intervals. The last two columns compare sizes within the same family. Absolute added tokens remain shared even when relative token expansion is weakly aligned.

Quantity	Across families	GPT-OSS siblings	Qwen siblings
Low-effort accuracy p_L	0.60 [0.58, 0.62]	0.82 [0.80, 0.84]	0.84 [0.82, 0.86]
High-effort accuracy p_H	0.74 [0.72, 0.75]	0.85 [0.83, 0.87]	0.85 [0.83, 0.88]
Accuracy gain Δ	0.20 [0.18, 0.22]	0.58 [0.53, 0.63]	0.40 [0.32, 0.49]
Low-effort log tokens c_L	0.53 [0.51, 0.55]	0.87 [0.85, 0.89]	0.95 [0.94, 0.95]
High-effort log tokens c_H	0.83 [0.82, 0.84]	0.96 [0.95, 0.96]	0.93 [0.92, 0.94]
Log-token expansion Δc	0.25 [0.24, 0.27]	0.85 [0.83, 0.87]	0.74 [0.71, 0.77]
Added tokens $\Delta\mu$	0.65 [0.63, 0.67]	0.91 [0.90, 0.92]	0.85 [0.80, 0.88]

For estimates X^A and X^B from halves A and B, we average the two cross-half covariances on each benchmark b, then normalize by estimated signal variances:

$$\widehat{\Sigma}_{b,mm'} = \tfrac{1}{2}\left[\mathrm{Cov}_q(X^A_{bmq}, X^B_{bm'q}) + \mathrm{Cov}_q(X^B_{bmq}, X^A_{bm'q})\right],$$
$$\overline{\Sigma} = \frac{1}{10}\sum_b \widehat{\Sigma}_b, \qquad \widehat{r}_{mm'} = \frac{\overline{\Sigma}_{mm'}}{\sqrt{\overline{\Sigma}_{mm}\overline{\Sigma}_{m'm'}}}. \tag{4}$$

Write an estimated half-profile as $X^A_m = \xi_m + \epsilon^A_m$ and $X^B_m = \xi_m + \epsilon^B_m$. Here ξ_m is the stable question-level quantity being compared. Conditional on the question, each error has mean zero and the two halves are independent. The law of total covariance then gives

$$\mathbb{E}[\widehat{\mathrm{Cov}}_q(X^A_m, X^B_{m'})] = \mathrm{Cov}_q(\xi_m, \xi_{m'}),$$

with the corresponding finite-question centering convention. Symmetrizing the cross-half products produces Equation 4. This removes the expected within-half sampling-noise contribution to covariance. Normalizing estimated variances remains a noisy ratio operation and is not unbiased.

Each task contributes equally to pooled covariance. For each of the 21 distinct family pairs, we average all cross-family model-pair correlations, then average these family-pair means. This includes 34 cross-family model pairs. The two same-family pairs are reported separately. Resampling uses 1,200 draws of the original number of question groups, independently by task and paired across quantities and models within task. Signal variances at most 10^{-7} make a ratio undefined. Values outside $[-1, 1]$ are preserved. If any required pair is undefined, the complete average is also undefined. We do not replace it with an average over the available pairs.

All ten-task pooled main estimates have complete bootstrap support. Some task estimates do not. In ProofWriter, Granite's gain signal variance is slightly negative, so the complete task gain average has no point estimate. That is evidence of insufficient estimated signal in that cell, not a measured zero correlation. Hollow markers flag intervals with fewer than 90% defined draws. The numerical artifact records every support count.

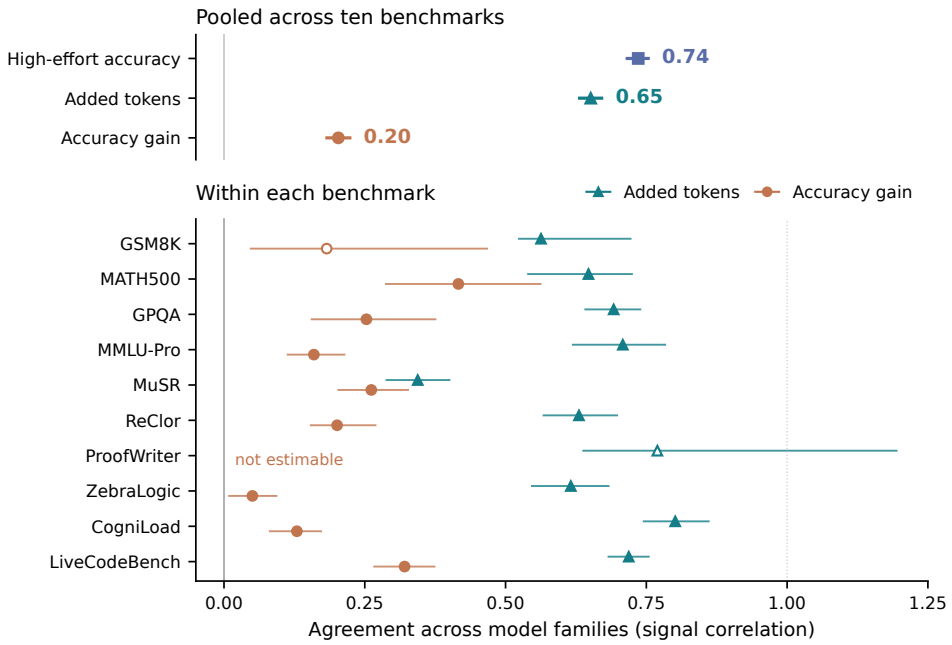


Figure 5: **Complete-bank agreement across quantities and benchmarks.** The top panel compares pooled signal correlations, and the lower panel shows each benchmark. Covariances are computed within benchmarks before pooling, with equal family-pair weights. Bars are 95% intervals. Estimates are not clipped and can exceed one. Hollow markers indicate fewer than 90% defined bootstrap draws. ProofWriter's gain average is not estimable because one model has too little stable gain variation.

Table 23: **Alternative repeat support, aggregation and effort controls.** The task-correlation mean is undefined if one task is undefined. Task-deletion ranges summarize points after each omission. Thinking off/on excludes both GPT-OSS sizes. †: fewer than 90% defined draws; —: undefined.

Quantity	First eight repeats	Mean task correlation	Task-deletion range	Off/on only
Low accuracy	0.60 [0.59, 0.62]	0.60 [0.58, 0.62]	0.58–0.62	0.59 [0.57, 0.60]
High accuracy	0.73 [0.71, 0.75]	0.73 [0.71, 0.76]†	0.72–0.75	0.74 [0.72, 0.76]
Accuracy gain	0.20 [0.18, 0.22]	–	0.18–0.23	0.18 [0.16, 0.21]
Low log tokens	0.53 [0.51, 0.55]	0.62 [0.59, 0.64]†	0.49–0.68	0.50 [0.48, 0.52]
High log tokens	0.83 [0.82, 0.84]	0.82 [0.81, 0.83]	0.81–0.85	0.83 [0.82, 0.84]
Log expansion	0.25 [0.24, 0.27]	0.24 [0.22, 0.26]	0.24–0.28	0.20 [0.19, 0.22]
Added tokens	0.65 [0.63, 0.67]	0.65 [0.63, 0.71]†	0.64–0.66	0.62 [0.60, 0.64]

Within-model reliability measures how much of the variation across questions is reproducible across independent response halves. We divide the covariance between the two half-sample estimates by their average variance, averaging these components across benchmarks before taking the ratio. Under the independent sampling assumptions above, the ratio estimates the fraction of variation in the half-sample estimates attributable to stable question differences rather than response-sampling noise. This measures repeatability within one model, not similarity between different models.

C.1 Starting accuracy, input length, and unfinished responses

The controls in Table 25 test whether agreement changes when accounting for starting accuracy, input demands, or missing answers. For submission probability a and correctness conditional on submission c, total accuracy is $p = ac$. Conditional correctness requires a submitted response in both halves for every model. This restriction estimates agreement among questions with submitted answers. It changes the question population and does not estimate the causal effect of eliminating delivery failures.

Table 24: **Reproducible question-level variation within each model.** Pooled split-half reliability, with the same model abbreviations as Table 10. Gain reliability is lower than accuracy-level reliability but remains positive in every model. This measures repeatability within a model, not agreement between models.

Quantity	GPT20	GPT120	Q9	Q27	GLM	EXA	Nem	Gran	Gem
Low accuracy	0.86	0.91	0.85	0.90	0.74	0.91	0.78	0.93	0.92
High accuracy	0.89	0.90	0.91	0.92	0.85	0.89	0.90	0.91	0.93
Accuracy gain	0.67	0.73	0.51	0.57	0.63	0.75	0.72	0.51	0.71
Low log tokens	0.88	0.93	0.96	0.97	0.70	0.95	0.61	0.93	0.97
High log tokens	0.97	0.98	0.95	0.95	0.96	0.98	0.96	0.96	0.98
Log expansion	0.88	0.91	0.78	0.85	0.69	0.94	0.78	0.87	0.88
Added tokens	0.96	0.96	0.89	0.90	0.89	0.88	0.93	0.78	0.86

Table 25: **Controls for the cross-model agreement comparison.** All use the complete model set. Counts show the number of questions retained. The final column reports defined draws out of 1,200. Residual high-effort accuracy and adjusted high-effort log tokens are different quantities from raw gains and added tokens.

Quantity or adjustment	Signal correlation [95%]	Questions	Defined draws
Accuracy gain	0.20 [0.18, 0.22]	2,518	1200/1200
Change in submission rate	–	2,518	0/1200
High-effort log tokens	0.83 [0.82, 0.84]	2,518	1200/1200
Gain on fully submitted questions	0.20 [0.16, 0.24]	1,388	1200/1200
Gain in correctness among submissions	0.21 [0.19, 0.24]	2,409	1200/1200
Jeffreys-probit gain	0.22 [0.19, 0.24]	2,518	1200/1200
Quadratic headroom residual	0.31 [0.28, 0.33]	2,518	1200/1200
Quartic headroom residual	0.29 [0.26, 0.32]	2,518	1200/1200
After input/workload adjustment	0.74 [0.72, 0.75]	2,518	1200/1200

To account for starting accuracy, we fit $p_H = f(p_L)$ separately by task and model, using other question groups for training. The degree-two and degree-four Bernstein polynomials use binomial moments to account for noise in measured starting accuracy: for $K \sim \mathrm{Binomial}(n, p)$,

$$\mathbb{E}\left[\binom{K}{j}\binom{n-K}{d-j}\Big/\binom{n}{d}\right] = B_{j,d}(p).$$

Degree-$2d$ moments form the feature products. The Gram matrix has an eigenvalue floor of 0.002, and coefficients are constrained to $[0, 1]$. Opposite halves give separate residuals $p_H - \widehat{f}(p_L)$. This bounded function class does not remove every possible starting-accuracy effect. The probit control instead transforms Jeffreys-smoothed half success rates. Neither residual agreement nor this finite-repeat transformation is a corrected version of the raw gain correlation.

For high-effort log tokens, five grouped folds predict from log input length, log annotated workload where available, and a workload-availability indicator. Missing workload is encoded as -1 after logging. The fixed gradient-boosted regressor uses 100 iterations, six leaves, minimum leaf size 15, learning rate 0.07, and ℓ_2 penalty four. These intervals hold the fitted adjustments fixed.

Table 26 removes any question with a capped response at the compared settings. The low/high comparison uses all nine models. The successive-step comparison uses both GPT-OSS sizes at low, medium, and high. The supports therefore differ. This tests whether the pattern persists among observed uncapped questions, rather than simulating a different output limit.

Table 26: **Agreement after excluding questions with capped outputs.** Counts show the common question support for each comparison. The step advantage compares matching transitions across sizes with successive transitions within a size. †: fewer than 90% defined draws; —: undefined.

Task	Endpoint questions	Added-token r	Gain r	Step advantage	Step questions
GSM8K	247	0.60 [0.54, 0.69]	0.20 [0.07, 0.47]†	0.74 [-0.10, 2.47]†	256
MATH500	233	0.59 [0.49, 0.65]	0.20 [-0.16, 0.31]†	–	253
GPQA	135	0.65 [0.57, 0.73]	0.21 [0.10, 0.34]	0.07 [-0.99, 0.55]	155
MMLU-Pro	308	0.60 [0.54, 0.66]	0.14 [0.09, 0.19]	0.04 [-0.18, 0.24]	325
MuSR	188	0.35 [0.30, 0.41]	0.26 [0.21, 0.33]	0.10 [-0.13, 0.28]	192
ReClor	247	0.60 [0.53, 0.67]	0.22 [0.17, 0.31]	0.15 [-0.10, 0.35]	256
ProofWriter	250	0.70 [0.65, 0.81]	–	–	256
ZebraLogic	186	0.38 [0.32, 0.43]	0.14 [0.07, 0.21]	0.53 [0.25, 0.88]	213
CogniLoad	169	0.85 [0.82, 0.88]	0.11 [0.05, 0.17]	0.40 [-0.02, 0.84]	250
LiveCodeBench	109	0.68 [0.63, 0.74]	0.38 [0.29, 0.48]	0.13 [-0.06, 0.28]	178
All ten tasks	2072	0.62 [0.59, 0.64]	0.20 [0.18, 0.23]	0.28 [0.17, 0.38]	2334

D SUCCESSIVE EFFORT INCREASES

Table 27: **Medium effort captures most of the accuracy improvement with a minority of the added tokens.** Shares divide the ten-task mean low-to-medium change by the ten-task mean low-to-high change. Brackets give 95% intervals, recomputing both means and their ratio in each resample.

Model	Gain captured (%)	Added tokens used (%)	Successive-step agreement
GPT-OSS 20B	77 [73, 80]	26 [24, 27]	0.26 [0.15, 0.39]
GPT-OSS 120B	69 [66, 73]	21 [20, 22]	0.16 [0.06, 0.26]

For each GPT-OSS size we estimate $p_M - p_L$ and $p_H - p_M$ separately in the two repeat halves. Cross-half covariance compares the first change from one half with the second from the other, then symmetrizes. The shared middle setting therefore contributes no shared decoding error across the two sides. The contrast in the main text is

$$\frac{r(20_{L\to M}, 120_{L\to M}) + r(20_{M\to H}, 120_{M\to H})}{2} - \frac{r(20_{L\to M}, 20_{M\to H}) + r(120_{L\to M}, 120_{M\to H})}{2}.$$

Each bootstrap draw reconstructs both sides from paired covariance estimates, preserving the dependence among overlapping model pairs. The effort-step analysis is a retrospective comparison of the three available settings. The direction is retained across every single-task omission, rather than being selected on a favorable benchmark.

Table 28: **Same-step agreement remains higher after each task omission.** Paired differences use the same resampling draws for all four terms.

Omitted task	Same-step advantage [95%]	Defined draws
GSM8K	0.26 [0.15, 0.36]	1200/1200
MATH500	0.29 [0.18, 0.39]	1200/1200
GPQA	0.30 [0.20, 0.40]	1200/1200
MMLU-Pro	0.29 [0.17, 0.40]	1200/1200
MuSR	0.30 [0.18, 0.41]	1200/1200
ReClor	0.29 [0.18, 0.40]	1200/1200
ProofWriter	0.24 [0.13, 0.34]	1200/1200
ZebraLogic	0.24 [0.12, 0.35]	1200/1200
CogniLoad	0.25 [0.14, 0.36]	1200/1200
LiveCodeBench	0.30 [0.18, 0.42]	1200/1200

D.1 EMPIRICAL BENEFIT REGIONS FROM REFERENCE-DEFINED DIFFICULTY

This descriptive check fixes five equally sized difficulty bins within each task using only scored responses from the six non-GPT model families. Difficulty is one minus the mean low/high reference

success rate, with equal family weights and deterministic question-ID tie breaking. Both GPT-OSS sizes are excluded from the ordering. Their low-to-medium and medium-to-high gains are then estimated from all their responses in those bins, and the ten task curves receive equal weight. Figure 6 displays all five bins for both sizes, without selecting tasks or bins using target outcomes.

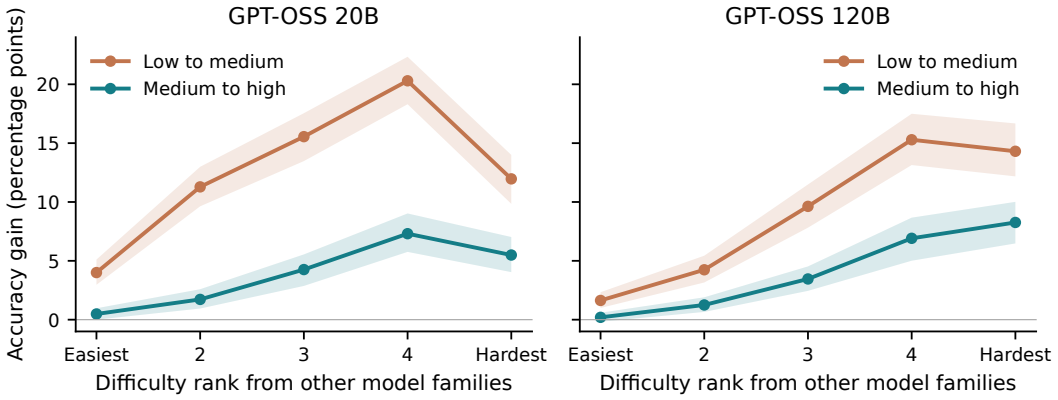


Figure 6: **Later gains are relatively more concentrated on questions the references find difficult.** Bins are fixed without GPT-OSS answers. Curves average the ten tasks. Shading gives conditional 95% question-group bootstrap intervals with the reference ordering fixed. The substantial overlap makes this a partial explanation, rather than evidence for disjoint benefit regions.

To summarize the relative shift, assign ranks $u_b \in \{0.1, 0.3, 0.5, 0.7, 0.9\}$ to the bins. For each step, its gain-weighted mean rank is $\sum_b u_b \bar{\Delta}_b / \sum_b \bar{\Delta}_b$. The denominator is positive for every pooled estimate and all 1,200 bootstrap draws. The mean rank rises for the later step in both sizes (Table 29). The 20B curves peak in the same coarse bin, so the result describes a shift in relative concentration, not a separate peak for every model. Reference difficulty and model ability remain observational quantities, and this comparison does not equalize computational doses.

Table 29: **The later effort step shifts gain weight toward higher reference difficulty.** Ranks run from zero to one. Paired 95% intervals resample question groups while keeping reference-defined bins fixed.

Model	Low-to-medium mean rank	Medium-to-high mean rank	Shift [95%]
GPT-OSS 20B	0.58	0.66	0.08 [0.05, 0.12]
GPT-OSS 120B	0.66	0.72	0.06 [0.02, 0.09]

The same-step comparison describes gain agreement. The artifact retains a separate allocation test of matching transitions and shared-difficulty response-model checks; neither is needed to implement the agreement rule studied here.

E THE VALUE OF QUESTION-SPECIFIC EFFORT CHOICES

Appendix E.1 defines the earlier-answer selection experiment. Appendix E.2 specifies the text features and regressions used by the main fixed-source protocol in Appendix F.3. Earlier forecast outputs with a different training/calibration partition remain in the artifact rather than being mixed with the active comparison.

E.1 CHOICES AND INDEPENDENT EVALUATION

For each model and question, one parity of response attempts estimates success rates and mean generated tokens at every native setting. The other parity evaluates the selected setting. Each half contains four responses per setting on six tasks and eight on the other four. We reverse the assignment and average both directions. For a fixed token penalty η, the choice is

$$\widehat{e}_q = \arg\max_{e \in \mathcal{E}_m} \{\widehat{p}_{mqe} - \eta \widehat{\mu}_{mqe}\}, \tag{5}$$

where $\mathcal{E}_m$ includes all native settings. Exact ties favor fewer predicted tokens, then the fixed configuration order. The six penalties are $\eta \in \{0, 1, 2, 5, 10, 20\} \times 10^{-6}$. Subtracting any common baseline success rate gives the equivalent gain-based objective. At $\eta = 10^{-5}$, 1,000 extra tokens require one percentage point of predicted gain.

The task-wide control averages those same training-half measurements across the task and chooses a single setting by the same objective. It receives exactly the same earlier answers. A low/high-only control removes intermediate settings while keeping measurements and the decision rule fixed. All choices precede the outcomes used to evaluate them. Reported answer tokens exclude the cost of acquiring the earlier measurements. The procedure estimates the value of finite, question-specific information rather than a perfect-information optimum or a deployable new-question selector.

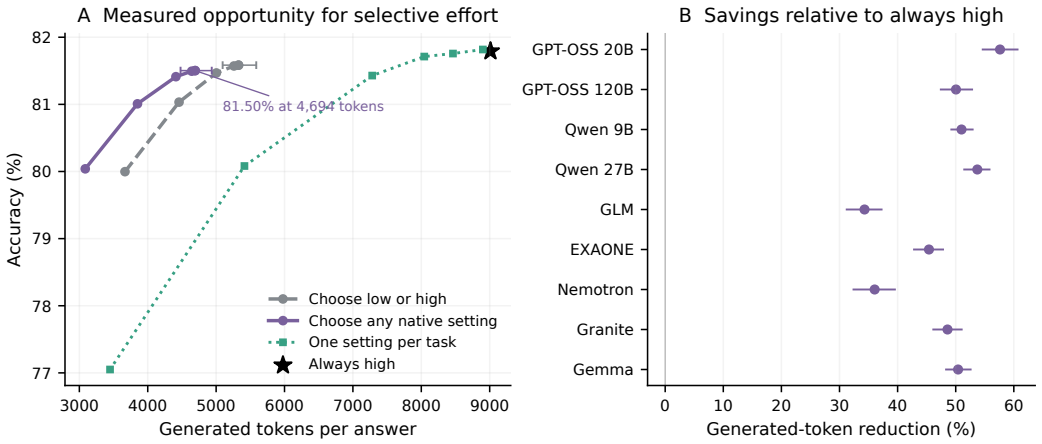


Figure 7: **Earlier answers contain useful question-specific selection information.** A: independently evaluated question-specific and task-wide choices across the six fixed penalties. Both use the same earlier measurements. B: token reductions for all-setting choices at zero penalty, averaging tasks within each target. These are token reductions, not a claim of equal accuracy for every target. Acquisition of earlier measurements is excluded.

Intermediate settings contribute to the saving. At $\eta = 10^{-5}$, restricting individual choices to low and high gives 81.03% accuracy at 4,458 tokens, compared with 81.01% at 3,850 when all native settings are available. This comparison keeps the earlier measurements and the token penalty fixed.

All results average questions within benchmark, then benchmarks and target models equally. The 1,000 paired bootstrap draws resample question groups within each benchmark, sharing weights across settings and targets. They hold the selection/evaluation assignments and resulting choices fixed. Related ProofWriter variants and LiveCodeBench contest problems remain grouped.

E.2 Prompt features and regression fits

The target's entire family is removed from source responses. Reference success rates and token means average models within family and then families equally. Predictors map question text to those quantities without observing responses to evaluation questions. Target fitting uses separate calibration answers, and all target attempts on held-out groups evaluate the selected setting. Calibration counts describe each fitted set, not the complete cross-validation acquisition. Appendix F.3 gives the training folds and calibration procedure.

Representations and fitting. Semantic features use frozen `Qwen/Qwen3-Embedding-0.6B` weights (revision `97b0c614be`) and unit-normalized last-token embeddings of the canonical prompt. No prompt exceeds the 16,384-token input limit. A radial-basis kernel uses the median nonzero training squared distance as its scale. The word-based alternative uses TF–IDF unigrams/bigrams, minimum document count two, at most 6,000 features, sublinear term frequency, and eight standardized log-count features for text length and character categories. It uses a linear kernel.

Kernel ridge predicts reference low/high success rates and $\log(1+\mu)$ at both settings. Three inner group folds choose the summed-squared-error penalty from $\{0.01, 0.1, 1, 10, 100\}$ separately for each output. Preprocessing is refitted within each split. Success predictions are clipped to $[0.001, 0.999]$. A target ridge fit maps predicted reference log costs to each target setting's log tokens. Exponentiated predictions are bounded between one and 81,920 before scaling to the training mean. This scaling and the penalty selection also use only inner training data.

Gain predictors use reference difficulty, reference gain, or both reference success rates. The last uses gain d, midpoint s, and their product ds. All share the same predicted target costs and calibration answers. Slopes are standardized and shrunk toward zero, the intercept is unpenalized, and grouped validation selects the penalty on the same grid. Gains are clipped to $[-1, 1]$. A direct target-text predictor learns from target calibration labels without reference correctness. The cost-only control predicts token use but assigns every question the target's mean calibration gain. All select one native setting using Equation 5.

The active results cover all six token penalties, both calibration sizes, and complete target comparisons in Appendix F. The original forecasting module and its unchanged outputs remain available for historical reproduction in the artifact.

F Separating cost information from gain information

This appendix gives the procedures behind three comparisons: what measured gain and cost information add, what learned predictions recover on new questions, and what changes when RADAR uses different cost estimates. It also specifies calibration variability and acquisition accounting. All comparisons retain native settings, negative gains, incomplete answers, and their costs. The module `information_sources_20260918` contains the specifications, saved predictions and decisions, training identities, complete results, and independent numerical checks.

F.1 Four matched information restrictions

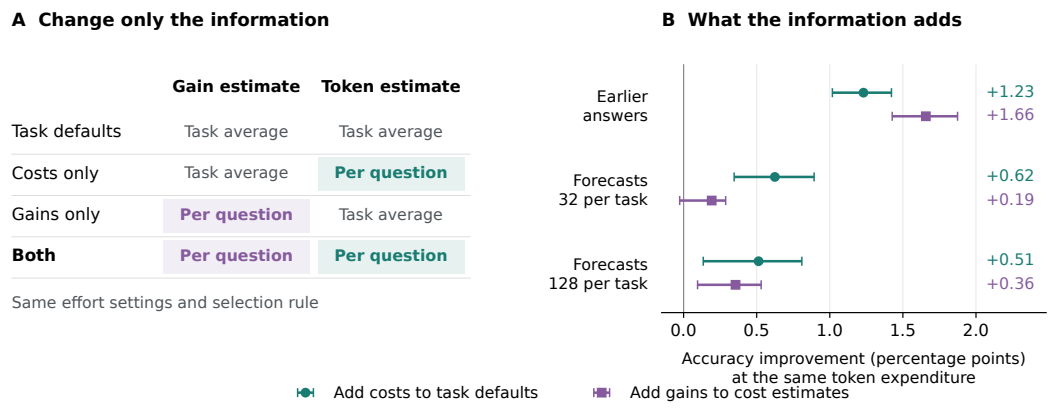


Figure 8: **Gain information helps beyond knowing costs, but forecasts recover less of its value.** A: four versions of the same selector change which estimates vary by question. B: each point shows the accuracy added by one kind of information at the same token budget. Earlier answers supply measured estimates; independent answers evaluate the choices. Forecasts learn from other questions, including 32 or 128 target calibration questions per task. Results average nine models at the midpoint budget defined in Section 2, with paired 95% intervals. Figure 10 gives the full budget comparison.

For each fixed token penalty, the selected setting maximizes estimated success minus that penalty times estimated tokens. We separately replace either vector with its benchmark mean. Ties use the cost estimate available to that row and then the native setting index. In particular, the average-cost rules do not receive question costs through a tie-break. For the earlier-answer diagnostic, means and question estimates come from the same even or odd response half. The opposite half evaluates the

choice. Reversing and averaging reproduces both the original opportunity and its task-default control exactly.

Table 30: **The four selectors at a fixed token penalty.** Each pair reports accuracy (%) and generated tokens per final answer. The penalty values one accuracy point at 1,000 tokens. Columns use earlier answers to the exact question, or forecasts from prompt embeddings calibrated on 32/128 separate target questions per benchmark. Acquisition is excluded from this diagnostic.

	Earlier answers		Text: 32		Text: 128	
Information	Acc.	Tokens	Acc.	Tokens	Acc.	Tokens
Task averages	80.08	5,414	79.52	5,601	79.90	5,502
Question costs	79.38	4,220	78.98	4,944	79.30	4,804
Question gains	81.32	4,438	79.24	5,471	79.81	5,458
Costs and gains	81.01	3,850	78.83	4,893	79.34	4,786

Table 31: **What adding gain estimates contributes across all six token penalties.** Joint minus cost-only penalized score $100(A - \eta T)$ [paired 95%], where A is accuracy as a proportion and T is mean generated tokens per answer. Positive values favor using both estimates; negative values favor cost-only. Zero penalty compares accuracy alone, while larger penalties put more weight on token savings. Text columns use the semantic predictor based on both reference success rates; calibration counts are per benchmark. Acquisition is excluded.

Penalty (10^{-6})	Earlier answers	Text: 32	Text: 128
0	−0.31 [−0.50, −0.11]	-0.18 [-0.24, -0.11]	-0.18 [-0.24, -0.12]
1	0.13 [−0.07, 0.32]	-0.23 [-0.30, -0.17]	-0.20 [-0.27, -0.14]
2	0.51 [0.32, 0.71]	-0.25 [-0.31, -0.18]	-0.17 [-0.24, -0.10]
5	1.50 [1.29, 1.74]	-0.18 [-0.24, -0.11]	-0.04 [-0.12, 0.04]
10	1.99 [1.78, 2.20]	-0.09 [-0.16, -0.03]	0.06 [-0.02, 0.14]
20	2.42 [2.21, 2.61]	0.04 [-0.03, 0.11]	0.23 [0.15, 0.33]

Table 31 reports every fixed penalty. Its negative zero-penalty contrast concerns a plug-in rule using finite estimates, not negative value for optimally used information: a rule with additional information could always ignore it. Likewise, comparing the earlier-answer and forecast columns does not hold supervision fixed and does not identify the cause of the forecast gap.

F.2 Information comparisons at equal token expenditure

For each target, we average questions within tasks and then tasks equally. Cost-only and joint forecasts supply six evaluated policies, one per token penalty. Task defaults include every distinct choice obtained by varying the penalty over their calibration estimates. We add every native setting and form the best supported accuracy from random mixtures. At each reported fraction of the native token range, we match expenditure within targets before averaging their paired accuracy differences. The complete current comparisons are in Figure 10 and Table 42. These nested restrictions measure separate information increments, not additive shares of the original selection opportunity.

The artifact preserves the original six-penalty default comparison, its numerical tables, and its figure in `paper/archived_information_defaults.tex`. The exhaustive default is the baseline for the submitted claims. Fixed-penalty gain comparisons in Table 31 remain relevant because their question is different: what happens when the accuracy–token preference itself is held fixed.

Forecasts trained on source responses also improve on direct target-text cost forecasts (Table 32). Both use the same calibration access and frozen text features, with different fitted representations. Source acquisition is excluded from this diagnostic.

F.3 Forecasts for new questions: training and calibration

This protocol supports Figure 1B and Table 31, using the representations and regularized predictors in Appendix E.2. Five whole-group outer folds separate evaluation questions from training. Within

Table 32: **Source-trained versus direct target-text cost forecasts.** Accuracy of source cost-only minus direct cost-only at equal tokens [95%]. Both use task-average gains; source acquisition is excluded.

Budget (%)	32 calibration questions	128 calibration questions
25	0.71 [0.57, 0.89]	0.39 [0.29, 0.49]
50	1.13 [0.92, 1.33]	0.65 [0.49, 0.80]
75	0.14 [0.07, 0.27]	0.02 [−0.02, 0.07]

the training questions, three group folds give out-of-fold source predictions for every possible target-calibration question. A final source model trained on the full outer training set predicts its test fold. Thus no source response to a predicted question, or its group, enters that prediction. Source training stays fixed when target calibration grows from 32 to 128 questions.

The source summaries balance reference families and exclude the target family. They predict low/high success and log(1+mean tokens) from frozen semantic or lexical features. Source and target penalties use the five-point grid and grouped inner validation, including refitting standardization and cost scaling. Target calibration takes one response per setting on each selected question, using three outcome-blind orders and attempts 0/1. Samples grow in whole groups. The original forecasting experiment excluded calibration groups from source training. Its outputs remain in the artifact, separate from this fixed-source protocol.

We retain difficulty, reference-gain, both-success-rate and direct target-text predictors, all six penalties, per-target results and equal-family summaries. The sibling-only comparison uses the other GPT-OSS or Qwen size as reference. The primary transfer comparison excludes those siblings. The complete new tables are `forecast_outcomes.csv` and `forecast_contrasts.csv`.

F.4 Controlled RADAR adaptation

Success model and relation to the transfer predictor. RADAR's published formulation routes among model–budget configurations and estimates a separate success probability for each query–configuration pair (Fernandez et al., 2026). We restrict the decision set to the native settings of one answering target, retaining its logistic response model. This is not the compact transfer gain regressor in Appendix F.3: the latter first predicts family-averaged reference outcomes, whereas the RADAR fit models configuration responses through shared question parameters and configuration abilities. For a fixed target, suppressing its model index,

$$\widehat{p}_{qe} = \sigma\big(a_q(\theta_e - b_q)\big), \qquad a_q = x_q^\top w_a, \quad b_q = x_q^\top w_b.$$

Here b_q is difficulty, a_q is the question-dependent slope (discrimination), θ_e is setting ability, and $\sigma(z) = 1/(1+\exp(-z))$. The vector x_q is the existing normalized 1,024-dimensional Qwen3-Embedding-0.6B representation plus an intercept. We pool source questions across tasks, excluding the target family and the outer test groups in every task. Source correctness means supply the binomial cross-entropy objective, with equal task weights. The displayed linear parameterization permits signed discrimination. It therefore need not be monotone in ability when $a_q < 0$; this is separate from not constraining abilities to follow the native effort order.

Three grouped source-validation folds select regularization of the projection weights and abilities. The initial grid $(10^{-4}, 10^{-3}, 10^{-2})$ selected its largest value and hit the optimizer limit. Before evaluating target decisions, we extended it to $10^{-1}, 1, 10$ and increased the iteration limit from 300 to 1,000. The original fits and diagnostics are retained. Deterministic L-BFGS uses analytic gradients. All final fits converge. A bounded bisection solver finds the convex target-ability optimum; an independent scalar optimizer checks it. The superseded clipped Newton fits are retained in the artifact after numerical validation detected nonconvergence. Target calibration estimates only each setting's scalar ability, with summed cross-entropy, a 10^{-3} quadratic penalty, and bounds $[-20, 20]$.

Two cost modules, one fixed success model. The average-cost baseline uses each target setting's mean token count on its calibration responses. The question-cost variant instead uses the source-to-target cost predictor defined in Appendices E.2 and F.3: prompt features predict reference log token costs, and a target ridge fit maps them to each setting's log token cost. Predictions are transformed

Table 33: **The RADAR comparison changes costs, not success predictions.** Both versions adapt RADAR to one target's native settings with identical data and selection criteria. Costs are expected generated tokens.

Component	Average-cost baseline	Question-cost variant
Success predictions	Fitted RADAR $\widehat{p}_{qe}$	The same $\widehat{p}_{qe}$
Token cost estimate	Setting average $\bar{\mu}_e$	Question forecast $\widehat{\mu}_{qe}$

and scaled using training data only, as specified there. The variant changes this cost input without replacing or refitting the RADAR success predictions. Both cost modules use the same source access and target calibration responses; the extra regressor does not require additional reasoning-model calls on evaluation questions. Small-predictor computation is not thereby a zero-cost operation.

Compared with the published RADAR experiments, the embedding model, native settings, optimizer, fixed rather than adaptive target calibration, and dataset differ. Both the average-cost baseline and the question-cost variant share these adaptations. The reported contrast therefore isolates the cost-module substitution within this implementation, rather than comparing a new full system with RADAR's published benchmark scores.

Unchanged selection criteria. Let $\bar{c}_e$ and $\widehat{c}_{qe}$ denote normalized average and question-specific costs, respectively. The two choices are

$$\widehat{e}_q^{\text{avg}} = \arg\max_{e \in E_m} f_w(\widehat{p}_{qe}, \bar{c}_e), \qquad \widehat{e}_q^{\text{question}} = \arg\max_{e \in E_m} f_w(\widehat{p}_{qe}, \widehat{c}_{qe}).$$

Only the second argument changes. For success p and normalized cost c, we evaluate the published linear objective $wp - (1-w)c$ and Chebyshev objective $-\max\{w(1-p), (1-w)c\}$ over $w \in \{0, .05, \ldots, 1\}$. Both cost modules use the minimum and maximum calibration configuration means for normalization; question costs are clipped to $[0, 1]$ in this sensitivity. The six original token penalties use untruncated arithmetic costs and are also retained. Supported same-cost comparisons include all native configurations and their mixtures. A target's midpoint is halfway between its cheapest and most expensive native mean token costs. We match within targets before averaging the paired accuracy differences.

The success estimates imply effort gains through $\widehat{p}_{qH} - \widehat{p}_{qL}$. Under an additive objective, subtracting a common baseline success rate cannot change the selected setting, as in Equation 2. The Chebyshev criterion is not generally invariant to that shift. We consequently retain the original success probabilities for both RADAR criteria; the cost-module experiment is not a gain-only replacement of RADAR. Its two versions use exactly the same predicted success probabilities.

Table 34: **The two information tests have different target patterns.** The first column is the earlier-answer gain-information increment in penalized-score points at $\eta = 10^{-5}$. The remaining columns are the RADAR cost-module accuracy improvements at the midpoint, using Chebyshev selection. Brackets give conditional paired 95% intervals, not simultaneous guarantees across targets.

Target	Gain information	Cost module: 32	Cost module: 128
GPT-OSS 20B	0.66 [0.11, 1.14]	0.00 [0.00, 0.00]	-0.00 [-0.02, 0.01]
GPT-OSS 120B	1.58 [1.06, 2.06]	0.00 [-0.00, 0.00]	0.00 [0.00, 0.01]
Qwen 9B	1.88 [1.44, 2.34]	0.59 [0.41, 0.80]	0.65 [0.50, 0.94]
Qwen 27B	1.84 [1.45, 2.22]	0.06 [-0.03, 0.15]	0.02 [-0.12, 0.16]
GLM	1.00 [0.49, 1.54]	2.00 [1.58, 2.43]	2.08 [1.62, 2.55]
EXAONE	3.77 [3.25, 4.33]	-0.03 [-0.12, 0.06]	-0.08 [-0.17, 0.04]
Nemotron	0.73 [0.22, 1.27]	3.64 [3.07, 4.19]	3.99 [3.38, 4.60]
Granite	4.29 [3.61, 4.99]	0.15 [0.02, 0.27]	0.17 [0.03, 0.31]
Gemma	2.20 [1.83, 2.57]	-0.21 [-0.31, -0.08]	-0.28 [-0.41, -0.11]

Table 35: **Absolute outcomes of the matched-token RADAR comparison.** Both cost modules retain the same success predictions, Chebyshev criterion, and 32-question calibration. All native settings and their mixtures are eligible. Differences are paired [95%].

		Accuracy (%)		
Target	Tokens	Average costs	Question costs	Difference (pp)
GPT-OSS 20B	6,645	78.88	78.88	$0.00\,[0.00, 0.00]$
GPT-OSS 120B	4,045	84.08	84.08	$0.00\,[-0.00, 0.00]$
Qwen 9B	6,421	79.36	**79.95**	$+0.59\,[0.41, 0.80]$
Qwen 27B	6,364	86.57	86.63	$+0.06\,[-0.03, 0.15]$
GLM	5,521	64.53	**66.52**	$+2.00\,[1.58, 2.43]$
EXAONE	4,490	75.52	75.49	$-0.03\,[-0.12, 0.06]$
Nemotron	4,162	60.12	**63.76**	$+3.64\,[3.07, 4.19]$
Granite	5,628	69.03	**69.18**	$+0.15\,[0.02, 0.27]$
Gemma	2,365	**85.83**	85.61	$-0.21\,[-0.31, -0.08]$

F.5 Which benchmarks benefit from question-cost forecasts?

The benchmark comparison reuses the fitted models and all saved decisions from Appendix F.4. It changes only where expenditure is matched. For each target–benchmark pair, we form the supported accuracy–token curves from the 21 evaluated preference weights and all native target settings. The comparison budget is halfway between that pair's cheapest and most expensive native mean token costs. We interpolate both curves at that budget and average their paired accuracy differences equally across the nine targets. This differs from Table 4, which first averages benchmarks and then matches each target's total expenditure.

We repeat the calculation within each of the original 1,000 question-group bootstrap draws, including the native budget and interpolation. Common weights preserve pairing across targets and cost modules. Table 36 reports both calibration counts and selection criteria. The artifact additionally gives every 25%, 50%, and 75% budget, absolute accuracy and tokens for all 90 target–benchmark pairs, and their paired intervals. These are evaluated tradeoff comparisons, with the interpretation in Appendix F.9.

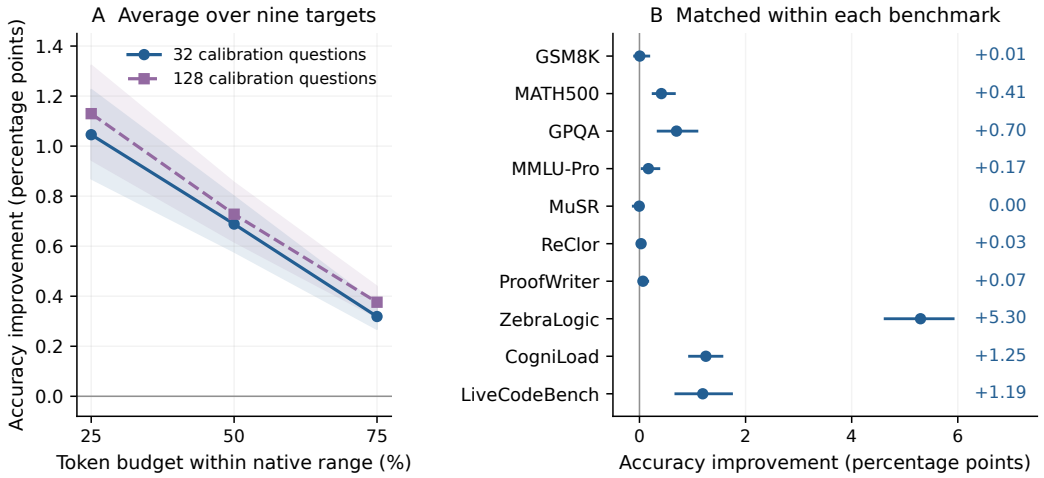


Figure 9: **The complete benchmark comparison.** A matches spending within each target at three budgets. B matches separately within every target–benchmark at its midpoint and averages all nine targets. Positive values favor question-cost forecasts. Chebyshev selection is used throughout, with paired 95% intervals.

ZebraLogic has the largest mean improvement under the primary Chebyshev criterion. The effect remains positive under the linear criterion but falls from 5.30 to 1.14 points with 32 calibration questions. The corresponding LiveCodeBench changes are 1.19 and 0.31 points, with the linear interval

Table 36: **Benchmark improvements with both selection criteria.** Accuracy differences (percentage points) at matched token expenditure within each target–benchmark, averaged equally over nine targets. Brackets give paired 95% intervals. Calibration counts are per benchmark.

	Chebyshev		Linear	
Benchmark	32 questions	128 questions	32 questions	128 questions
GSM8K	+0.01 [−0.09, 0.17]	+0.02 [−0.00, 0.12]	+0.01 [−0.08, 0.18]	+0.04 [0.00, 0.17]
MATH500	+0.41 [0.26, 0.66]	+0.45 [0.24, 0.77]	+0.38 [0.20, 0.70]	+0.45 [0.24, 0.79]
GPQA	+0.70 [0.35, 1.08]	+1.02 [0.59, 1.54]	+0.99 [0.65, 1.41]	+1.70 [1.19, 2.26]
MMLU-Pro	+0.17 [0.05, 0.36]	+0.35 [0.12, 0.57]	+0.12 [0.03, 0.28]	+0.23 [0.07, 0.45]
MuSR	−0.00 [−0.12, 0.04]	−0.00 [−0.13, 0.06]	+0.02 [−0.08, 0.09]	+0.04 [−0.05, 0.13]
ReClor	+0.03 [−0.02, 0.08]	+0.02 [−0.06, 0.04]	+0.06 [0.01, 0.09]	+0.03 [−0.02, 0.05]
ProofWriter	+0.07 [−0.00, 0.16]	+0.10 [0.02, 0.22]	+0.17 [0.07, 0.32]	+0.26 [0.05, 0.47]
ZebraLogic	+5.30 [4.63, 5.91]	+5.54 [4.85, 6.27]	+1.14 [0.74, 1.73]	+0.90 [0.55, 1.58]
CogniLoad	+1.25 [0.94, 1.55]	+1.43 [1.09, 1.82]	+1.31 [0.96, 1.61]	+1.66 [1.19, 2.03]
LiveCodeBench	+1.19 [0.69, 1.73]	+1.31 [0.73, 1.89]	+0.31 [−0.17, 0.81]	+0.25 [−0.21, 0.78]

including zero. Both criteria improve several benchmarks, with different magnitudes depending on how success and cost enter the decision.

Table 37: **The benchmark pattern is heterogeneous across targets.** Question costs minus average costs in accuracy points at each target–benchmark's midpoint, with 32-question calibration and Chebyshev selection. All settings remain eligible, including GPT-OSS medium and Granite brief. Complete absolute outcomes and paired intervals are in `radar_benchmarks_20260922/results/within_benchmark.csv`.

	GPT-OSS 20B	GPT-OSS 120B	Qwen 9B	Qwen 27B	GLM	EXAONE	Nemo.	Granite	Gemma
GSM8K	+0.00	+0.00	−0.00	−0.00	+0.00	−0.01	+0.06	+0.00	−0.00
MATH500	+0.00	+0.00	−0.01	+0.04	−0.03	−0.35	+4.10	+0.00	−0.02
GPQA	+0.00	+0.00	−0.14	−0.09	+0.85	+0.23	+4.98	+0.56	−0.11
MMLU-Pro	+0.04	+0.00	+0.00	+0.00	+0.07	+0.65	+0.61	+0.13	−0.00
MuSR	+0.00	+0.00	+0.00	+0.01	−0.01	−0.04	+0.02	−0.02	−0.00
ReClor	+0.00	+0.00	−0.00	−0.01	−0.06	−0.06	+0.40	+0.00	+0.00
ProofWriter	+0.00	+0.00	+0.00	+0.00	+0.18	−0.04	+0.47	+0.00	−0.01
ZebraLogic	+4.47	−0.09	+3.94	+0.44	+17.61	−1.98	+23.81	+1.45	−1.97
CogniLoad	+0.00	+0.00	−0.00	−0.08	+3.36	+0.65	+7.60	+0.00	−0.26
LiveCodeBench	+0.00	−0.02	+0.11	−0.02	+5.59	−0.16	+5.35	+0.00	−0.12

The coding improvements for GLM and Nemotron are 5.59 [3.24, 7.92] and 5.35 [3.06, 7.43] points, respectively. On GPQA, Nemotron improves by 4.98 [2.85, 6.82] points. ZebraLogic improves for several targets but declines for EXAONE and Gemma (Table 37). Changing the cost input establishes its decision value in these comparisons; the breakdown does not by itself identify which property of a benchmark or target causes the difference.

How these results relate to Table 4. We also decompose the original Table 4 comparison without changing its mixture weights. Each target's weights, obtained from the equal-benchmark aggregate curves, are applied to the individual benchmarks' outcomes. Averaging these contributions reproduces all 1,001 original point/bootstrap entries for every target. Table 38 reports accuracy and token changes together, because only aggregate expenditure is matched in this calculation. ZebraLogic contributes about 0.46 of the aggregate 0.69-point accuracy increase. LiveCodeBench and CogniLoad receive more tokens, while several other benchmarks receive fewer. The separate within-benchmark comparison above shows that the improvement is not solely a redistribution between tasks.

F.6 Which effort decisions change?

The saved RADAR decisions allow a direct account of the cost-module change without refitting either predictor. For each question, we average the six calibration views and apply the compared

Table 38: **Benchmark outcomes of the original Table 4 policies.** Both columns are question costs minus average costs, averaged over nine targets with paired 95% intervals. These preserve the original aggregate mixture weights. Token changes average to zero across benchmarks, but individual benchmarks need not have equal spending.

Benchmark	Accuracy difference (pp)	Token difference per answer
GSM8K	−0.51 [−0.67, −0.40]	−94 [−151, −40]
MATH500	+0.14 [−0.18, 0.27]	−219 [−445, −61]
GPQA	+0.51 [0.19, 0.88]	−306 [−478, −103]
MMLU-Pro	−0.19 [−0.39, −0.03]	−141 [−268, −103]
MuSR	−0.43 [−0.59, −0.25]	−39 [−84, −13]
ReClor	−0.17 [−0.62, −0.09]	−99 [−167, −55]
ProofWriter	−0.67 [−0.88, −0.46]	−112 [−152, −88]
ZebraLogic	+4.61 [4.08, 5.46]	−46 [−178, 80]
CogniLoad	+1.64 [1.19, 2.00]	+264 [81, 440]
LiveCodeBench	+1.96 [1.63, 2.72]	+792 [605, 1258]

tradeoff mixtures. Let r_q be the resulting probability of high effort with question costs minus that with average costs. For a two-setting target, its accuracy and token changes are exactly

$$\delta A = \mathbb{E}_q[r_q(p_{qH} - p_{qL})], \qquad \delta T = \mathbb{E}_q[r_q(\mu_{qH} - \mu_{qL})].$$

We separate positive and negative r_q. The gains and costs in Figure 2B–C average the measured upgrade outcomes with weights proportional to $|r_q|$ in each group. They describe upgrades receiving more or less weight, rather than a binary partition of all questions. For three-setting targets, we apply the same identity across both ordered setting boundaries, retaining GPT-OSS medium and ordering Granite as brief/off/on by measured mean cost. These boundary contributions sum exactly to the original outcomes.

Table 39: **Coding upgrades receiving more and less weight.** LiveCodeBench, 32-question calibration, Chebyshev selection, and expenditure matched within each target–benchmark. Mass is the total change in high-effort probability in each direction, as a percentage of questions. Gains and costs describe one upgrade under those weights, with paired 95% intervals.

Target	High effort used	Mass (%)	Gain (pp)	Additional tokens
GLM	More	24.3	41.4 [35.8, 47.5]	16,656 [12,863, 20,429]
GLM	Less	11.0	40.7 [33.4, 48.0]	36,838 [31,977, 41,763]
Nemotron	More	23.1	37.5 [31.3, 42.6]	10,262 [7,714, 12,867]
Nemotron	Less	8.2	40.6 [33.2, 47.9]	29,006 [23,093, 34,693]

GLM increases high-effort use by a net 13.3 [9.9, 16.5] points and Nemotron by 14.9 [11.7, 18.3], at unchanged mean token expenditure. Their upgrades used less often cost 2.21 [1.77, 2.93] and 2.83 [2.12, 3.85] times as much as upgrades used more often. The corresponding difference in measured gain is 0.7 [−8.0, 9.2] points for GLM and −3.1 [−12.9, 6.1] for Nemotron. The cost contrasts are clear; these gain intervals do not establish equivalence.

The module `cost_decisions_20260922` saves every task and target, both criteria, both calibration counts, and both aggregate and within-task midpoint comparisons. Mixtures, direction groups and weighted outcomes are recomputed in every paired draw. Independent reconstruction verifies all sum identities and the original accuracy differences. This accounts for how the evaluated decisions improve, not for the model's internal reasoning mechanism.

Every single-benchmark and single-family omission retains a positive interval for the primary Chebyshev contrast. Nevertheless, GLM and Nemotron account for most of its magnitude: omitting both leaves 0.08 [0.04, 0.12] points. Equal-family weighting gives 0.84 [0.70, 0.98]. The full omission results and 25/75% budgets are retained in the artifact.

Table 40: **How much do concentrated benefits drive the mean?** Midpoint RADAR accuracy improvements [95%], with 32-question calibration. Task omissions recompute native budgets and mixtures after omitting the task. Family omissions reaggregate the remaining targets without refitting.

Aggregation	Chebyshev	Linear
All models and tasks	0.69 [0.58, 0.80]	0.38 [0.28, 0.49]
Equal-family weights	0.84 [0.70, 0.98]	0.45 [0.33, 0.58]
Omit ZebraLogic	0.17 [0.09, 0.28]	0.25 [0.18, 0.34]
Omit Nemotron family	0.32 [0.26, 0.39]	0.14 [0.07, 0.22]
Omit GLM and Nemotron	0.08 [0.04, 0.12]	0.10 [0.06, 0.14]

F.7 Fresh calibration and uncertainty

We refit on 50 additional outcome-blind whole-group orders (1000–1049), alternating target response attempts 0/1. Every compared procedure uses the same selections and evaluation folds. Source fits remain fixed. We refit target transfer regressions, direct text fits, RADAR abilities, 32/128-question defaults, and all four paid length controls. Length policies are evaluated over all recorded probe subsets. Both RADAR tradeoff rules are included in the fresh-calibration and paid comparisons.

The paired evaluation intervals use 1,000 question-group bootstrap draws with the original seeds. Economic comparisons use the common weights of the agreement study, rather than subtracting arrays from different bootstrap runs. Each frontier is reconstructed inside each draw. Central calibration stability ranges instead describe variability over the 50 actual fits. They are not population confidence intervals. At the reference preference, the semantic joint-minus-cost-only score averages -0.04 [-0.18, 0.11] over these fits with 32 questions and 0.05 [-0.05, 0.13] with 128. The midpoint RADAR Chebyshev changes average 0.70 [0.62, 0.80] and 0.74 [0.66, 0.82], respectively. Complete ranges and conditional intervals are saved separately.

F.8 Fully priced learned alternatives

For N later answers per task, model-call cost is the mean final-answer bill plus recurring probe costs plus one-time acquisition divided by N. Source success and cost heads share calls, which are charged once. Direct target-text methods require no source responses. Calibration acquisition is averaged over the fitted sets, matching the original task-default accounting. The 32/128 default decisions and their setup bills reconstruct exactly under the new calculation.

In the existing-bank scenario, previously acquired source calls are sunk. Cold-start transfer must purchase all source endpoint responses used in fitting. RADAR uses all source native configurations, and its pooled acquisition is spread over the stated ten-task workload. EXAONE has no frozen price. Its complete cold-start source bill is therefore undefined. The fully priced sensitivity refits after excluding EXAONE; it does not assign that model a zero or invented price. For Qwen targets, this quoted source bank costs approximately $30 per task for the endpoint forecasts and $32 for the configuration-response fit. Target calibration is additional.

At the four-probe Qwen bill, the source-trained cold-start curves at the 1,000-answer horizon are outside supported cost ranges. Direct target-text and task-default controls remain affordable and are included in the strongest learned envelope. That envelope also includes both source banks when already acquired, every information ablation, both calibration sizes, and both RADAR tradeoff criteria. The four-probe accuracy advantage is 0.41 [0.16, 0.57] points with an existing bank and 0.43 [0.18, 0.62] under cold-start accounting. At 10,000 answers both comparisons are near 0.01 and their intervals include zero. Unsupported individual comparisons remain undefined, with support counts in the artifact.

The preceding figures concern the original learned collection. The simple-cost and exhaustive-default extension in Appendix G strengthens it further; Table 5 uses that extension. Over 50 fresh calibration selections of the original collection, agreement's advantage averages 0.30 [0.03, 0.49] points with an existing bank at 1,000 answers. Under cold-start accounting it averages 0.36 [0.08, 0.59]. Both longer-horizon stability ranges span zero. These ranges retain the same fixed agreement evaluation and vary the acquired calibration sample.

The dollar accounting covers model calls at frozen listed and full-allowance quotes. Encoder and small-predictor computation, and the creation of correctness labels, are not assigned an API-token price. The frozen embedding record reports 1.41 million input tokens and about 630 seconds for the complete bank on the recorded local device. This is an acquisition/runtime component to consider, not a zero-cost or hosted-latency claim.

F.9 Interpreting the comparisons

Measured information and forecast information. The earlier-answer diagnostic uses multiple scored responses to the exact question whose next answer is evaluated. The forecasting experiments instead use sparse target calibration on other questions and must generalize to excluded question groups. Their difference combines information access, repeat depth, and prediction; it does not isolate a representation limitation or establish that gains are intrinsically unlearnable. In the compact transfer model, the current evidence also does not locate a loss of information specifically in reference averaging, source regression, or target calibration. The negative zero-penalty contrast in Table 31 concerns a finite-sample plug-in selector, rather than negative value for optimally used information. The gain and cost contrasts can interact and are not additive shares of an efficiency improvement.

The RADAR comparison. The component test establishes an improvement over the average-cost version of a matched native-setting adaptation. Both variants retain the same fitted success predictions and selection criteria. It does not claim that the separate transfer gain predictor is a better success model than RADAR, nor that RADAR's question-specific success estimates are dispensable. Table 4 includes all nine targets: GLM and Nemotron contribute most of the mean improvement, and Gemma deteriorates. These results identify where the variant helps; the current comparison does not isolate the reason for those target differences. The implementation changes relative to the published multi-model RADAR experiments are listed in Appendix F.4.

Evaluated tradeoffs and advance budget selection. The cost-matched comparisons interpolate evaluated fixed policies and their mixtures. The best supported boundary describes the accuracy–cost tradeoff of that policy collection. These interpolation weights are not chosen using a separate calibration set in advance to meet an unseen budget. Section 4.3 supplies a separate advance-budget evaluation with fixed effort probabilities. The underlying per-question choices exclude their evaluation outcomes. Confidence intervals condition on the selected analysis, policy collection, and fitted procedures; they do not account for the broader history of inspecting this response bank and choosing later analyses. The 50 target-calibration refits vary a particular source of fitting uncertainty while keeping the source fits and evaluation population fixed.

Acquisition and the choice of answering model. The paid comparison holds Qwen 27B fixed and uses frozen provider rates to price local generations. It does not measure hosted-system accuracy, latency, or hardware computation. Four probes reduce expensive target output tokens while increasing total output across reference and target models from approximately 10,356 to 17,307 tokens per final answer (Appendix J.3). The broader screen finds positive paid comparisons concentrated on Qwen 27B; native intermediate settings defeat some other rules even with free probes, while other useful signals cost more than their selection benefit. Returning a unanimous reference answer is competitive. Under the same reported prices, always-high Gemma reaches 87.95% accuracy at $1.35 per 1,000 answers, so the fixed-Qwen improvement does not establish the best unrestricted answering system. Appendix J retains the complete screen, reuse controls, early stopping, and price sensitivities. Encoder and small-predictor computation, as well as correctness-label creation, are not assigned an API-token price.

G Simple cost controls and advance budgets

This appendix asks how simple the cost predictor can be, how thoroughly the task-default baseline must be evaluated, and whether cost forecasts help when the budget is set in advance.

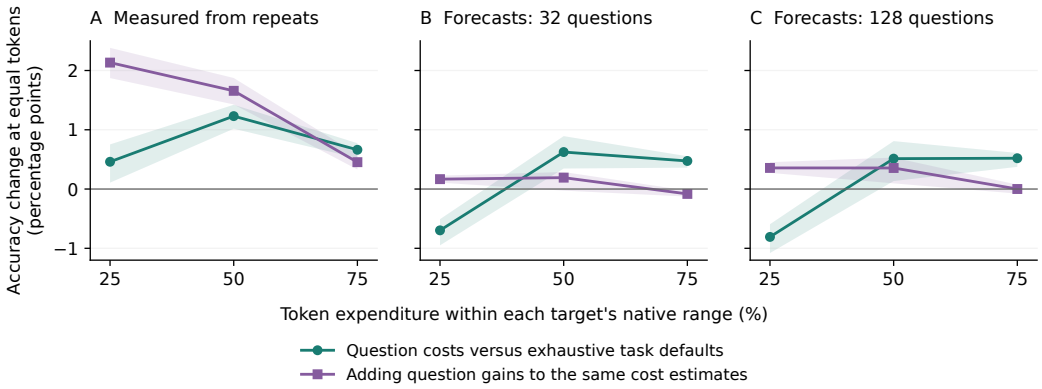


Figure 10: **The value of gain and cost information changes with the token budget.** Green adds question-specific costs to task defaults. Purple adds question-specific gains to those costs. A estimates these quantities from earlier answers to each question and evaluates choices on independent answers. B–C forecast from text, calibrated on separate target questions. Tokens are matched within each target before averaging all nine models. Shading gives paired 95% intervals. At the tighter forecast budget, task defaults outperform cost-only selection.

Table 41: **Prompt length supplies useful cost estimates within RADAR.** Accuracy changes (pp) at the midpoint budget with 32-question calibration [paired 95%]. Column 1 uses task-average gains and compares cost-only selection with task defaults. Columns 2–3 retain RADAR's success predictions and Chebyshev criterion, changing only average costs to the named predictor. The last column omits two evaluation tasks while keeping the source success fit fixed.

Cost forecast	Task-average gains All ten tasks	RADAR success All ten tasks	RADAR success Without Zebra/CogniLoad
Prompt length	$+0.18\,[-0.09, +0.39]$	$+0.67\,[+0.58, +0.76]$	$+0.08\,[+0.05, +0.14]$
Length + workload	$+0.16\,[-0.11, +0.37]$	$+0.70\,[+0.61, +0.80]$	$+0.10\,[+0.06, +0.16]$
Source length + workload	$+0.21\,[-0.06, +0.43]$	$+0.70\,[+0.61, +0.80]$	$+0.09\,[+0.06, +0.15]$
Embedding transfer	$+0.62\,[+0.35, +0.89]$	$+0.69\,[+0.58, +0.80]$	$+0.08\,[+0.03, +0.17]$

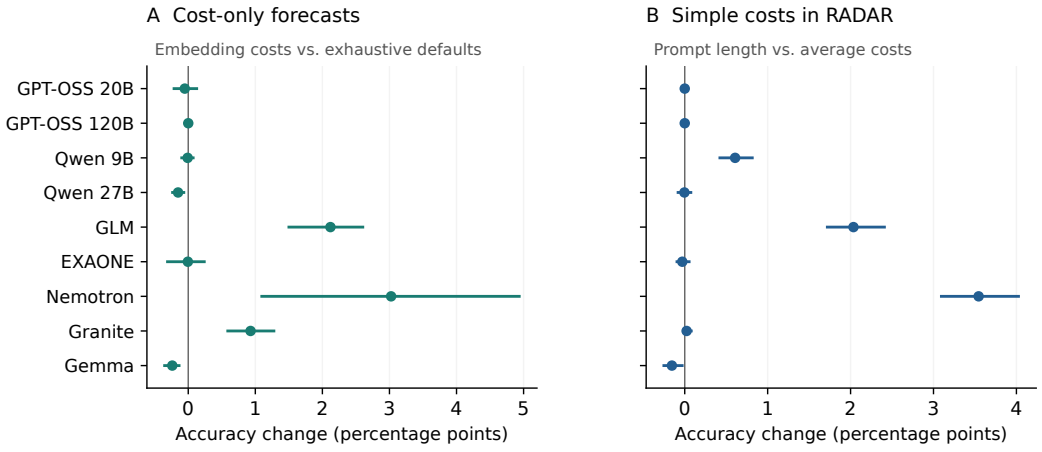


Figure 11: **The value of cost forecasts varies across targets.** A: embedding-based cost forecasts versus task defaults, with gains held at task averages. B: prompt-length versus average costs in RADAR, retaining its success predictions. Both compare accuracy at the midpoint budget after 32-question calibration. Bars show paired 95% intervals; horizontal scales differ.

G.1 Length and workload controls

The direct length predictor uses $\log(1 + \text{input tokens})$. The workload version adds $\log(1 + \text{annotated workload})$ and an availability indicator, with missing log workload encoded as -1. These are prompt attributes, not measurements of the target's generated response. Each setting's log output tokens are fitted by ridge regression with an unpenalized intercept. Standardization, penalty selection from $\{0.01, 0.1, 1, 10, 100\}$, and arithmetic-mean scaling use training data only, including inside grouped validation. Both controls acquire the same target calibration responses as the original forecaster.

The source-length control first regresses family-balanced reference costs on the same simple features. It excludes the target family and the evaluation groups. Calibration predictions are out of fold, and a target ridge fit adapts them using the original calibration sample. Its source pool is fixed between 32 and 128 questions. All these cost fits are task-local. Consequently, removing ZebraLogic and CogniLoad removes them from cost/gain training as well as evaluation. The separate RADAR omission reuses its pooled source success fit and is explicitly an evaluation omission.

G.2 Exhaustive task defaults

For each calibration view, a default chooses the setting maximizing $\bar{p}_e - \eta\bar{\mu}_e$. Its choice changes only at a nonnegative crossing of two setting scores. We enumerate the union of those crossings across benchmarks, folds, and calibration views for each target, evaluate every distinct policy, and retain its supported mixtures with all native settings. The same construction uses the two response halves in the repeat-measurement diagnostic. This removes dependence on the original six penalties for the default baseline, while retaining the original comparison in the artifact history described in Appendix F.2.

At the midpoint, the embedding cost-only contrast falls from 1.42 to 0.62 points with the stronger default. At the tighter budget, the exhaustive default does better. This is an effect of the policy collection: an identical coarse penalty grid need not cover every method's useful operating points equally well.

G.3 Matched accuracy and benchmark exclusions

For a matched-accuracy comparison we use 25%, 50%, or 75% of each target's native accuracy range, and minimize tokens over the supported mixtures attaining that accuracy. Percent savings are the reduction in summed target tokens divided by the baseline sum, after matching accuracy within each target. These accuracy positions differ from the token positions in Figure 10. Neither interpolation extrapolates beyond support, and each is recomputed inside every paired draw.

Table 42: **Budget and accuracy level change the comparison.** Embedding cost-only versus exhaustive defaults, and RADAR question costs versus average costs, with paired 95% intervals. Column 4 matches tokens at the stated fraction of the native token range. Columns 5–6 match accuracy at that fraction of the native accuracy range. Negative savings mean greater token use. "Eight" omits ZebraLogic and CogniLoad, recomputing budgets and curves.

Tasks	Cal.	Position	Cost-only accuracy (pp)	Cost-only token saving (%)	RADAR token saving (%)
Ten	32	25	$-0.70\,[-0.95, -0.51]$	$-14.12\,[-16.88, -12.05]$	$+14.76\,[+12.37, +17.93]$
Ten	32	50	$+0.62\,[+0.35, +0.89]$	$-2.94\,[-5.70, -1.04]$	$+8.02\,[+6.53, +9.87]$
Ten	32	75	$+0.47\,[+0.36, +0.55]$	$+5.03\,[+2.93, +6.03]$	$+3.38\,[+2.51, +4.17]$
Eight	32	25	$-1.06\,[-1.31, -0.85]$	$-19.86\,[-23.64, -16.73]$	$+1.75\,[+0.41, +4.97]$
Eight	32	50	$+0.37\,[+0.07, +0.55]$	$-8.41\,[-12.47, -5.81]$	$+0.40\,[-0.58, +1.94]$
Eight	32	75	$+0.33\,[+0.23, +0.41]$	$+1.56\,[-0.84, +2.95]$	$+0.03\,[-0.34, +0.76]$
Ten	128	25	$-0.81\,[-1.08, -0.59]$	$-14.55\,[-17.69, -12.31]$	$+15.55\,[+12.74, +19.11]$
Ten	128	50	$+0.51\,[+0.13, +0.81]$	$-4.51\,[-7.83, -2.45]$	$+8.69\,[+6.96, +10.91]$
Ten	128	75	$+0.52\,[+0.37, +0.61]$	$+3.30\,[+0.77, +5.13]$	$+3.67\,[+2.64, +4.99]$
Eight	128	25	$-1.10\,[-1.36, -0.86]$	$-21.41\,[-25.61, -17.89]$	$+2.76\,[+0.96, +6.31]$
Eight	128	50	$+0.54\,[+0.15, +0.69]$	$-8.82\,[-13.50, -5.98]$	$+0.75\,[-0.63, +2.38]$
Eight	128	75	$+0.40\,[+0.26, +0.48]$	$+1.30\,[-1.92, +3.65]$	$-0.29\,[-0.83, +1.05]$

The cost-only curves cross: their midpoint-token accuracy improvement does not imply savings at every requested accuracy. RADAR's all-task midpoint-accuracy saving is approximately 8%, with most of that magnitude absent after omitting ZebraLogic and CogniLoad. Complete results retain every benchmark, target, feature control, calibration size, and both RADAR criteria.

G.4 Budget choices made before evaluation

For each calibration view and held-out benchmark group, fix B at a stated fraction between the cheapest and most expensive calibration-average native costs. Use only predicted success and cost to choose probabilities x_{qe} solving

$$\max_{x} \frac{1}{Q}\sum_{q,e} x_{qe}\widehat{p}_{qe} \quad \text{subject to} \quad \frac{1}{Q}\sum_{q,e} x_{qe}\widehat{\mu}_{qe} \le B, \qquad \sum_{e} x_{qe} = 1, \quad x_{qe} \ge 0.$$

The unlabeled batch's prompts are available. A dual search over the linear penalty locates the relevant switching point and mixes the adjacent choices when needed. This gives fixed effort probabilities without using target evaluation correctness or output tokens. A single answer can then be drawn under the selected setting. If predicted minimum cost exceeds B, the rule uses its cheapest predicted setting and records infeasibility. Saved results report predicted budget errors as well as actual accuracy and tokens. Independent linear programs verify the allocation solution.

Setting probabilities remain fixed during the paired evaluation bootstrap. For the 32-question midpoint RADAR embedding-cost rule, 12 of 2,700 target–task–fold–view batches have an infeasible predicted budget; their cheapest-setting outcomes are retained. These counts concern fitted evaluation batches, not proportions of individual questions or an estimate of future budget violations.

The table in the main text uses RADAR's success predictions with this linear constrained allocator, not the normalized Chebyshev criterion. It also fixes separate budgets within benchmarks. It therefore answers an operational question alongside the original equal-token comparison, rather than estimating the same quantity a second time. Realized expenditure can exceed or undershoot the predicted budget. The 128-question, tight-budget, generous-budget, and per-target outcomes remain in the artifact.

Table 43: **The stronger controls retain heterogeneous targets.** All use 32 calibration questions. The first two columns are midpoint equal-token accuracy changes. The final two evaluate a calibration-set midpoint budget fixed before answers, comparing RADAR's success predictor with embedding versus average costs. Brackets are paired 95% intervals.

Target	Cost-only change (pp)	RADAR length change (pp)	Advance-budget accuracy (pp)	Advance-budget tokens
GPT-OSS 20B	$-0.05\,[-0.21, +0.13]$	$+0.00\,[+0.00, +0.00]$	$+0.68\,[+0.35, +1.04]$	$-715\,[-1126, -323]$
GPT-OSS 120B	$+0.00\,[-0.02, +0.01]$	$+0.00\,[+0.00, +0.00]$	$-0.53\,[-0.84, -0.20]$	$-574\,[-885, -295]$
Qwen 9B	$-0.01\,[-0.10, +0.07]$	$+0.61\,[+0.42, +0.81]$	$+0.30\,[+0.02, +0.57]$	$-350\,[-559, -151]$
Qwen 27B	$-0.15\,[-0.23, -0.07]$	$-0.00\,[-0.08, +0.07]$	$-0.10\,[-0.34, +0.15]$	$-360\,[-571, -152]$
GLM	$+2.12\,[+1.50, +2.60]$	$+2.03\,[+1.72, +2.41]$	$+3.96\,[+3.20, +4.68]$	$+494\,[+184, +821]$
EXAONE	$-0.01\,[-0.31, +0.24]$	$-0.03\,[-0.09, +0.05]$	$-0.98\,[-1.54, -0.42]$	$-303\,[-500, -105]$
Nemotron	$+3.02\,[+1.10, +4.93]$	$+3.54\,[+3.10, +4.02]$	$+7.65\,[+6.64, +8.67]$	$+654\,[+370, +932]$
Granite	$+0.93\,[+0.59, +1.28]$	$+0.02\,[-0.01, +0.08]$	$+0.85\,[+0.36, +1.34]$	$-190\,[-431, +48]$
Gemma	$-0.24\,[-0.35, -0.14]$	$-0.15\,[-0.25, -0.03]$	$-1.53\,[-1.92, -1.18]$	$-252\,[-338, -175]$

G.5 Gain forecasts under the same advance-budget procedure

The saved advance-budget experiment also contains task defaults, cost-only forecasts, and joint gain/cost forecasts. All use the linear constrained allocator above, the same calibration samples, native settings, and planned budget. Defaults use task averages for success and cost. Cost-only forecasts replace costs with question estimates while retaining task-average gains. Joint selection changes only the success estimates to the original predictor based on both reference success rates. Its cost inputs are identical to cost-only selection. No evaluation outcomes enter these decisions.

Table 44 reports actual accuracy and tokens alongside planned spending. Table 45 gives paired joint-minus-cost contrasts. Gains improve accuracy at each planned budget, but often increase actual

spending as well. At the midpoint, the 32-question change is +0.27 [0.15, 0.40] points and +156 [106, 209] tokens; the 128-question change is +0.35 [0.18, 0.51] points and +59 [−17, 142] tokens. These are operating outcomes, not equal-realized-token improvements. Table 46 shows their target dependence.

Table 44: **Three information restrictions with budgets chosen in advance.** All nine targets and ten tasks, equally weighted. Calibration is per task; position is the fraction of the calibration-derived native token range. Accuracy and tokens are measured after committing choices. Planned spending is common to the three rules.

Cal.	Position (%)	Information	Accuracy (%)	Actual tokens	Planned tokens
32	25	Task default	71.94	2,893	3,056
32	25	Costs only	74.31	3,080	3,056
32	25	Costs and gains	74.46	3,172	3,056
32	50	Task default	75.21	4,531	4,975
32	50	Costs only	77.15	4,679	4,975
32	50	Costs and gains	77.42	4,835	4,975
32	75	Task default	78.15	6,108	6,895
32	75	Costs only	79.06	6,043	6,895
32	75	Costs and gains	79.23	6,169	6,895
128	25	Task default	72.17	3,003	3,112
128	25	Costs only	74.74	3,245	3,112
128	25	Costs and gains	75.03	3,303	3,112
128	50	Task default	75.60	4,838	5,075
128	50	Costs only	77.73	5,119	5,075
128	50	Costs and gains	78.08	5,179	5,075
128	75	Task default	78.70	6,644	7,038
128	75	Costs only	79.77	6,729	7,038
128	75	Costs and gains	79.93	6,691	7,038

Table 45: **What gain forecasts add to the same cost forecasts.** Joint minus cost-only under the advance-budget procedure [paired 95%]. A shared planned budget need not produce equal actual expenditure.

Cal.	Position (%)	Accuracy change (pp)	Actual token change
32	25	$+0.15\,[0.04, 0.26]$	$+92\,[58, 129]$
32	50	$+0.27\,[0.15, 0.40]$	$+156\,[106, 209]$
32	75	$+0.17\,[0.06, 0.27]$	$+125\,[70, 178]$
128	25	$+0.29\,[0.15, 0.43]$	$+58\,[11, 110]$
128	50	$+0.35\,[0.18, 0.51]$	$+59\,[-17, 142]$
128	75	$+0.16\,[0.02, 0.30]$	$-39\,[-113, 39]$

Table 46: **Target outcomes of adding gain forecasts at the planned midpoint.** Joint minus cost-only, with paired 95% intervals. The artifact retains every target at all three planned budgets.

	32 calibration questions		128 calibration questions	
Target	Accuracy (pp)	Tokens	Accuracy (pp)	Tokens
GPT-OSS 20B	$+0.30\,[0.06, 0.51]$	$+347\,[187, 494]$	$+0.08\,[-0.17, 0.32]$	$-2\,[-207, 202]$
GPT-OSS 120B	$+0.40\,[0.15, 0.65]$	$+177\,[81, 267]$	$+0.27\,[-0.01, 0.56]$	$+81\,[-50, 230]$
Qwen 9B	$+0.14\,[-0.07, 0.33]$	$+436\,[336, 540]$	$+0.01\,[-0.29, 0.30]$	$+258\,[91, 439]$
Qwen 27B	$+0.28\,[0.11, 0.45]$	$+364\,[257, 474]$	$+0.55\,[0.26, 0.83]$	$+341\,[163, 536]$
GLM	$+0.32\,[0.07, 0.58]$	$-64\,[-147, 12]$	$+0.39\,[0.04, 0.75]$	$-37\,[-170, 106]$
EXAONE	$+0.49\,[0.03, 0.90]$	$+70\,[-41, 181]$	$+0.84\,[0.25, 1.37]$	$+9\,[-134, 157]$
Nemotron	$+0.05\,[-0.26, 0.37]$	$-67\,[-139, 7]$	$+0.51\,[0.12, 0.91]$	$-49\,[-136, 39]$
Granite	$-0.19\,[-0.47, 0.11]$	$+54\,[-53, 143]$	$-0.21\,[-0.62, 0.19]$	$-57\,[-197, 80]$
Gemma	$+0.66\,[0.37, 0.96]$	$+87\,[27, 154]$	$+0.68\,[0.25, 1.12]$	$-10\,[-87, 71]$

These allocations were computed with the original cost-control module.

G.6 Interpreting target differences

The exact midpoint zeros for GPT-OSS concern supported policy curves. Both cost modules' best mixture there uses the native medium/high alternatives. Some learned policies improve the lower-cost region, so the zeros do not mean the cost forecasts never change useful decisions. Conversely, a positive dispersion of costs does not ensure a better selector: useful gains, predictive ordering, and the baseline's available settings all matter.

We also examined 90 target–benchmark cells using cross-half covariances to estimate reproducible gain and added-token variance. Gains remain signed; we do not take their logarithms. Descriptive Spearman associations of RADAR benefit with mean gains, cost dispersion, and gain dispersion are modest (approximately 0.31, 0.27, and -0.29). The relative gain-dispersion statistic is undefined in two cells with nonpositive estimated signal variance. These same-bank associations do not establish a predictive rule for a new target. Complete cells and family-omission associations are retained in the artifact.

G.7 Strong controls in the dollar comparison

The new direct length/workload costs and exhaustive task defaults also enter Table 5. RADAR uses its original priced-reference success fits. Every method pays for its target calibration; the RADAR source bank is sunk in the main existing-bank scenario and charged in the cold-start scenario. A default needs no source bank. The source-length transfer fit that includes unquoted EXAONE is not assigned an invented cold-start price. Question-bootstrap weights match the agreement experiment's seed and task order. All unchanged historical guidance comparisons remain in their original modules.

The enlarged learned comparison raises its shorter-horizon accuracy from 87.16% to 87.19%. Agreement retains a 0.38 [0.14, 0.56]-point advantage there. The previously reported 50-refit comparison concerns the older learned-policy collection; it does not assess calibration variation for these newly added simple controls. We retain its provenance rather than relabel it as a refit of the expanded comparison.

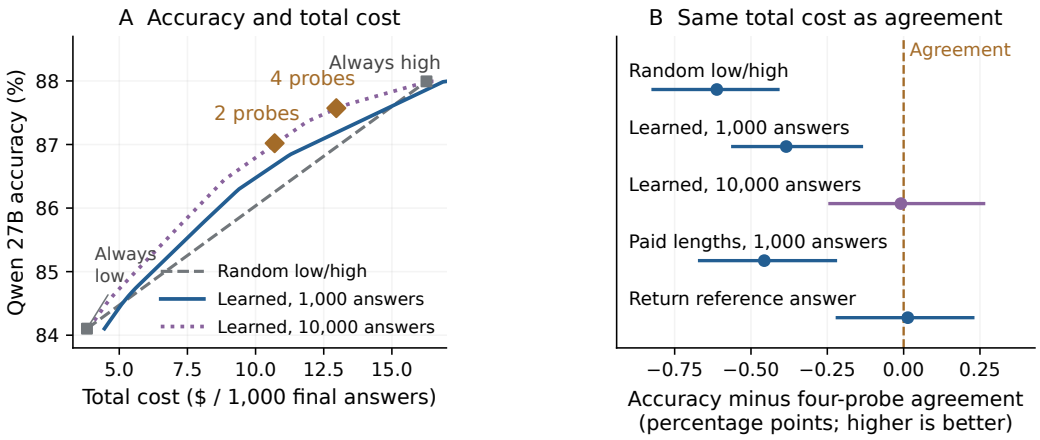


Figure 12: **Agreement's advantage depends on the alternative and workload.** A: Qwen 27B accuracy versus total cost, including probes and amortized calibration. Learned curves include the simple-cost controls and exhaustive task defaults. B: each alternative's accuracy minus four-probe agreement at the same bill [paired 95%]. Left of zero means lower accuracy.

H Answer agreement: procedure, evidence, and controls

This appendix specifies the fixed agreement rule, its expected-outcome evaluation, and the controls used to test its information and cost. The original two- and four-probe rules apply uniformly across

tasks. The reproducibility statement records the analysis sequence, and the artifact preserves every specification.

H.1 Information available and complete cost accounting

The fixed rule observes normalized final-answer identities and whether extraction succeeded. All acquired identities must be recognized and identical to select the lower setting. Otherwise it selects high. No target answer, target token count, reference correctness, or gold label is used to decide. The reference answers are neither returned as the final answer nor passed to the target. Both probe counts apply uniformly across tasks, without benchmark-specific thresholds. Labeled benchmark answers are used only to evaluate the rule. For GPT-OSS, both low/high and medium/high versions were specified, and all native-setting mixture controls include medium.

Answer identities are computed without gold answers. Extraction first produces a category, mathematical answer string, structured grid/set, or program. Category tasks compare extracted choices. ZebraLogic grids are keyed by house, and CogniLoad compares attribute sets. Mathematical strings remove whitespace, dollar signs and commas and lowercase text, while preserving decimal points. For programs, a separate hash uses the last complete code block after trimming its outer whitespace. Case, punctuation, and internal whitespace remain intact. These are task-specific identities, not a mathematical or program-equivalence solver. Appendix H.5 audits the archived normalizer and verifies that preserving decimal points leaves every reported gate decision unchanged.

Let a question have R recorded independent low-effort reference responses, with c_j occurrences of valid final-answer identity j. The probability of unanimity for k distinct sampled responses is

$$u_q(k) = \frac{\sum_{j \in \text{valid}} \binom{c_j}{k}}{\binom{R}{k}}. \tag{6}$$

Invalid or missing answers remain in R but contribute to no valid identity. We evaluate the expected outcome of the actual k-call rule: $u_q p_{qL} + (1 - u_q) p_{qH}$. Its expected complete dollar cost is

$$k\, C_{qL}^{\text{ref}} + u_q C_{qL}^{\text{target}} + (1 - u_q) C_{qH}^{\text{target}}. \tag{7}$$

Linearity of expectation permits charging the mean of all purchased reference calls even when their length and agreement are associated. Averaging over subsets estimates the expected outcome across possible runs of the two- or four-call rule. Each run observes and pays for only k answers. Target means use independent model responses. An opposite-half sensitivity uses one reference response half and the opposite target half, then reverses and averages. It gives 87.04% at \$10.69 for two probes and 87.60% at \$12.97 for four, close to the principal 87.02%/\$10.70 and 87.57%/\$12.96.

All input and generated tokens are charged using the frozen per-model provider rates, including unsuccessful and length-terminated calls. The listed scenario uses the least expensive quoted provider. The full-allowance scenario also requires an output allowance of at least 81,920 tokens. Provider definitions and quotes appear in Table 55. The priced replay does not measure provider-specific accuracy or latency.

H.2 Paired comparisons and stronger direct-calibration controls

Questions receive equal weight within each benchmark, and benchmarks receive equal weight in the aggregate. Paired 95% intervals use 1,000 bootstrap draws over question groups within benchmarks. Related questions share weights, and every compared rule uses the same weights. We reaggregate the saved task-default decisions using these same weights so that differences remain paired. The intervals condition on the evaluated model roster and fitted procedures.

The free mixture selects low/high independently of the question, with a fixed probability shared across benchmarks. The original direct-target control buys low/high pairs within a \$0.222 average allowance, while the complete-pair sensitivity finishes and charges its final pair. Each control chooses a task default. Varying a fixed probability of replacing that choice with high traces a comparison curve that retains the setup bill. Agreement has no setup bill, and all recurring probes remain charged. Table 47 compares accuracy at each agreement rule's evaluated total cost. Every primary comparison has all 1,000 draws within interpolation support.

We also retain the initial sweep's 32- and 128-question target-calibration curves. Each uses five group folds, three outcome-blind calibration orders and two response views, with one target response per setting on separate labeled questions. For each of twelve fixed dollar penalties, the selected setting maximizes sample accuracy minus the penalty times sample cost. Ties favor lower cost. These direct controls cost \$0.641 and \$2.574 per task, respectively. To give this baseline its strongest comparison, we use the upper envelope of all tested penalty points, their mixtures, and always high with the setup bill retained. This describes the best tradeoff among fixed policies after evaluation. It is not an additional policy trained on test outcomes. Each paired bootstrap draw recomputes the envelope and interpolates accuracy at the agreement rule's cost.

Table 47: **Agreement accuracy above the tested alternatives at its evaluated total cost.** Entries give percentage points with paired 95% intervals. Acquisition costs are spread over 1,000 final answers per task. "Direct allowance" uses the original \$0.222 task allowance under listed prices. The larger samples use their full acquisition bills. The last two columns compare the upper envelopes of all fixed calibration policies and their mixtures. The direct-calibration columns use the original twelve-penalty defaults, not the exhaustive defaults used in the main-text comparison.

Quotes	Probes	Free mixture	Direct allowance	Direct 32	Direct 128
Listed	Two	0.77 [0.58, 0.97]	0.47 [0.28, 0.65]	0.53 [0.32, 0.72]	1.07 [0.78, 1.32]
Listed	Four	0.61 [0.41, 0.82]	0.46 [0.26, 0.65]	0.43 [0.22, 0.63]	0.75 [0.43, 1.02]
Full allowance	Two	0.84 [0.65, 1.04]	0.57 [0.40, 0.73]	0.56 [0.36, 0.76]	1.17 [0.90, 1.41]
Full allowance	Four	0.75 [0.55, 0.97]	0.59 [0.40, 0.78]	0.52 [0.30, 0.74]	0.86 [0.53, 1.16]

Larger future workloads reduce the relative price of calibration. Table 48 retains all three horizons and both scenarios. At 10,000 later answers per task, the four-probe advantage over the 128-question direct curve is 0.01 [−0.25, 0.28] points under listed prices. The interval includes zero at this horizon, where calibration is competitive with recurring probes. At short horizons, some calibration curves cost more than agreement at every point. Those comparisons remain undefined because interpolation has no support.

Table 48: **The acquisition horizon changes the comparison.** Agreement-minus-direct accuracy [paired 95%], at the agreement rule's evaluated total cost. The last column gives defined draws out of 1,000. Direct controls use the original twelve-penalty defaults.

Quotes	Target sample	Later answers	Probes	Advantage (points)	Defined
Listed	32	100	Two	2.72 [2.35, 3.09]	975
Listed	32	100	Four	2.37 [2.00, 2.75]	1000
Listed	32	1,000	Two	0.53 [0.32, 0.72]	1000
Listed	32	1,000	Four	0.43 [0.22, 0.63]	1000
Listed	32	10,000	Two	0.34 [0.14, 0.52]	1000
Listed	32	10,000	Four	0.30 [0.11, 0.51]	1000
Full allowance	32	100	Two	2.80 [2.42, 3.17]	860
Full allowance	32	100	Four	2.54 [2.16, 2.94]	1000
Full allowance	32	1,000	Two	0.56 [0.36, 0.76]	1000
Full allowance	32	1,000	Four	0.52 [0.30, 0.74]	1000
Full allowance	32	10,000	Two	0.39 [0.19, 0.57]	1000
Full allowance	32	10,000	Four	0.40 [0.19, 0.60]	1000
Listed	128	100	Two	–	0
Listed	128	100	Four	–	0
Listed	128	1,000	Two	1.07 [0.78, 1.32]	1000
Listed	128	1,000	Four	0.75 [0.43, 1.02]	1000
Listed	128	10,000	Two	0.17 [−0.09, 0.40]	1000
Listed	128	10,000	Four	0.01 [−0.25, 0.28]	1000
Full allowance	128	100	Two	–	0
Full allowance	128	100	Four	–	0
Full allowance	128	1,000	Two	1.17 [0.90, 1.41]	1000
Full allowance	128	1,000	Four	0.86 [0.53, 1.16]	1000
Full allowance	128	10,000	Two	0.20 [−0.07, 0.44]	1000
Full allowance	128	10,000	Four	0.12 [−0.16, 0.40]	1000

A setting learned from the Qwen reference task profile offers another comparison. Relative to this fully charged control, four-probe agreement changes accuracy by -0.07 [-0.28, 0.16] points and saves approximately $1.64 per 1,000 answers. The accuracy interval includes zero. Appendix I describes the procedure and its variation over 200 fresh calibration samples.

H.3 Question-specific information and benchmark variation

To estimate target gains conditional on agreement, we weight each question by its probability of unanimous or non-unanimous reference answers. We divide the weighted gain sum by the corresponding question weight. The resulting 1.33 versus 10.87 points compare Qwen 27B low-to-high gains conditional on two Qwen 9B low-effort answers, using the same subset probabilities as the operational rule. Because the two groups contain different proportions of tasks, task composition contributes to this separation.

The within-task frequency control instead chooses high independently of question identity with probability equal to that task's expected agreement-rule high frequency. This rate is recomputed in each bootstrap draw. The control uses the evaluated task frequency as a diagnostic, rather than learning it for free in deployment. It matches the amount of high effort in call counts, not target tokens or the additional probe bill. Agreement improves accuracy by 0.63 [0.45, 0.81] points with two probes and 0.81 [0.55, 1.09] with four. Table 47 separately tests whether that information is worth its monetary cost.

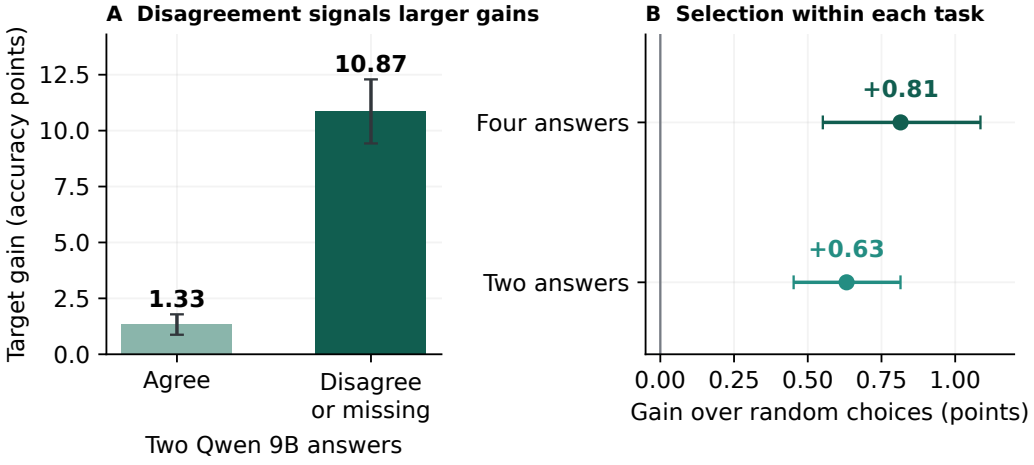


Figure 13: **Disagreement identifies larger target gains and useful choices within tasks.** Panel A shows Qwen 27B low-to-high gains conditional on two Qwen 9B low-effort answers. Task composition contributes to this pooled separation. Panel B compares agreement with random high-effort choices at the same frequency within each task, recomputed in every bootstrap. This diagnostic matches frequency, not token expenditure.

Every single-benchmark omission retains a positive within-task targeting point estimate and paired interval. For four probes, point estimates range from 0.67 to 0.91 points. Final-answer identity uses the bank's fixed, gold-blind normalization. Exact agreement can miss mathematically equivalent expressions or equivalent programs. On LiveCodeBench, the rule almost always selects high and pays the extra probe bill. It therefore does not save money on every task.

H.4 All reference–target pairs and mixed references

Figure 14 and Table 50 retain all nine pairs, both prices, and the medium/high variant for GPT-OSS. Native-setting controls include medium: the two-probe GPT-OSS sibling medium/high rule is below the native mixture by 0.52 [0.26, 0.77] accuracy points at comparable listed-price cost. Several probes cost more than an inexpensive target's entire high-effort answer. Undefined same-cost entries indicate

Table 49: **Every benchmark contributes to evaluating the Qwen rule.** Listed prices. Each group reports target accuracy, complete dollars per 1,000 final answers, target high frequency, and accuracy above the within-task frequency control. The table gives point estimates. Paired intervals and task-omission results are in the artifact.

	Two low-effort probes				Four low-effort probes			
Task	Acc.	$	High (%)	Gain (pp)	Acc.	$	High (%)	Gain (pp)
GSM8K	96.56	1.32	4.4	0.31	96.67	1.68	7.4	0.41
MATH500	97.91	4.75	6.7	0.60	98.27	5.93	11.4	0.89
GPQA	85.62	12.64	17.7	0.52	86.31	17.74	29.8	1.00
MMLU-Pro	84.09	4.97	16.1	0.31	84.28	6.71	25.6	0.48
MuSR	69.77	3.08	23.5	0.02	70.06	4.30	40.9	-0.03
ReClor	97.80	1.74	5.7	0.14	97.96	2.29	10.2	0.25
ProofWriter	78.40	2.15	11.4	0.87	79.43	2.96	20.7	1.57
ZebraLogic	93.96	12.78	33.2	2.06	94.70	15.53	48.4	2.10
CogniLoad	90.22	22.93	49.8	1.47	92.16	30.35	73.7	1.47
LiveCodeBench	75.88	40.63	99.8	0.03	75.88	42.13	100.0	0.01

a lack of interpolation support, often because the probe bill exceeds always high. We still report their absolute accuracy and cost.

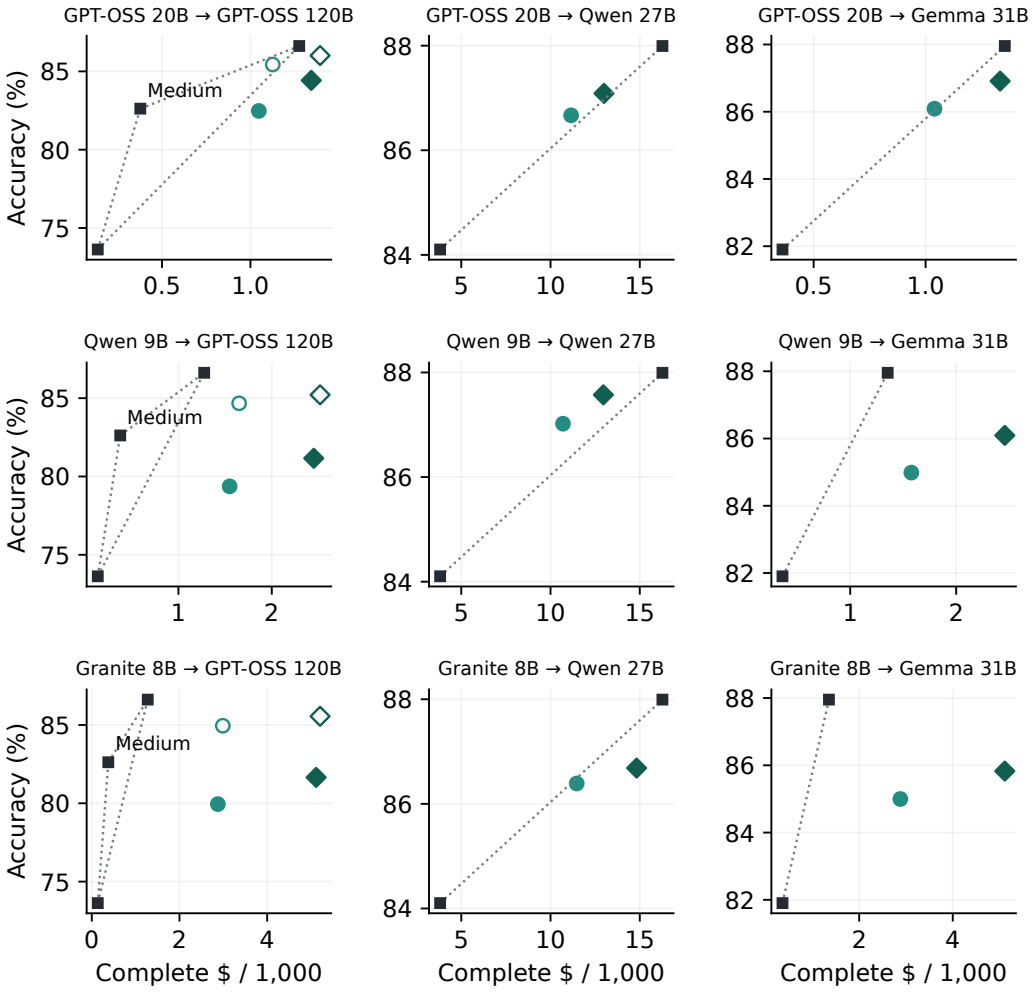


Figure 14: **Question-specific agreement information does not always pay for its calls.** The figure uses listed prices and shows all nine pairs. Squares and dotted lines mark the target's native settings and random mixtures. Circles use two low-effort reference answers, and diamonds use four. Filled markers select target low/high. Hollow GPT-OSS markers select medium/high. Medium remains an explicit alternative. Different panels have different cost scales.

Table 50: **Complete single-reference agreement comparison.** L/H selects low on unanimity and high otherwise. M/H selects medium on unanimity. Premium is accuracy above the full native-setting mixture envelope at the evaluated complete cost, with paired 95% intervals.

Quotes	Reference	Target	Choice	Probes	Acc.	Total $	Premium (pp)
Listed	GPT-OSS 20B	GPT-OSS 120B	L/H	2	82.47	1.05	−3.13 [−3.60, −2.69]
Listed	GPT-OSS 20B	GPT-OSS 120B	L/H	4	84.42	1.34	–
Listed	GPT-OSS 20B	GPT-OSS 120B	M/H	2	85.44	1.13	−0.52 [−0.77, −0.26]
Listed	GPT-OSS 20B	GPT-OSS 120B	M/H	4	86.01	1.39	–
Listed	GPT-OSS 20B	Qwen 27B	L/H	2	86.67	11.15	0.27 [0.05, 0.50]
Listed	GPT-OSS 20B	Qwen 27B	L/H	4	87.09	12.99	0.12 [−0.12, 0.35]
Listed	GPT-OSS 20B	Gemma 31B	L/H	2	86.09	1.04	0.05 [−0.25, 0.31]
Listed	GPT-OSS 20B	Gemma 31B	L/H	4	86.91	1.33	−0.92 [−1.27, −0.58]
Listed	Qwen 9B	GPT-OSS 120B	L/H	2	79.37	1.55	–
Listed	Qwen 9B	GPT-OSS 120B	L/H	4	81.16	2.45	–
Listed	Qwen 9B	GPT-OSS 120B	M/H	2	84.67	1.65	–

Quotes	Reference	Target	Choice	Probes	Acc.	Total $	Premium (pp)
Listed	Qwen 9B	GPT-OSS 120B	M/H	4	85.21	2.52	–
Listed	Qwen 9B	Qwen 27B	L/H	2	87.02	10.70	0.77 [0.58, 0.97]
Listed	Qwen 9B	Qwen 27B	L/H	4	87.57	12.96	0.61 [0.41, 0.82]
Listed	Qwen 9B	Gemma 31B	L/H	2	84.99	1.58	–
Listed	Qwen 9B	Gemma 31B	L/H	4	86.09	2.46	–
Listed	Granite 8B	GPT-OSS 120B	L/H	2	79.95	2.87	–
Listed	Granite 8B	GPT-OSS 120B	L/H	4	81.65	5.12	–
Listed	Granite 8B	GPT-OSS 120B	M/H	2	84.95	2.99	–
Listed	Granite 8B	GPT-OSS 120B	M/H	4	85.55	5.21	–
Listed	Granite 8B	Qwen 27B	L/H	2	86.39	11.47	−0.10 [−0.39, 0.16]
Listed	Granite 8B	Qwen 27B	L/H	4	86.68	14.82	−0.86 [−1.23, −0.52]
Listed	Granite 8B	Gemma 31B	L/H	2	85.00	2.88	–
Listed	Granite 8B	Gemma 31B	L/H	4	85.83	5.11	–
Full allowance	GPT-OSS 20B	GPT-OSS 120B	L/H	2	82.47	1.05	−3.13 [−3.60, −2.69]
Full allowance	GPT-OSS 20B	GPT-OSS 120B	L/H	4	84.42	1.34	–
Full allowance	GPT-OSS 20B	GPT-OSS 120B	M/H	2	85.44	1.13	−0.52 [−0.77, −0.26]
Full allowance	GPT-OSS 20B	GPT-OSS 120B	M/H	4	86.01	1.39	–
Full allowance	GPT-OSS 20B	Qwen 27B	L/H	2	86.67	18.45	0.29 [0.06, 0.52]
Full allowance	GPT-OSS 20B	Qwen 27B	L/H	4	87.09	21.43	0.15 [−0.09, 0.39]
Full allowance	GPT-OSS 20B	Gemma 31B	L/H	2	86.09	1.05	0.05 [−0.25, 0.31]
Full allowance	GPT-OSS 20B	Gemma 31B	L/H	4	86.91	1.34	−0.92 [−1.27, −0.58]
Full allowance	Qwen 9B	GPT-OSS 120B	L/H	2	79.37	1.67	–
Full allowance	Qwen 9B	GPT-OSS 120B	L/H	4	81.16	2.70	–
Full allowance	Qwen 9B	GPT-OSS 120B	M/H	2	84.67	1.77	–
Full allowance	Qwen 9B	GPT-OSS 120B	M/H	4	85.21	2.77	–
Full allowance	Qwen 9B	Qwen 27B	L/H	2	87.02	17.42	0.84 [0.65, 1.04]
Full allowance	Qwen 9B	Qwen 27B	L/H	4	87.57	20.82	0.75 [0.55, 0.97]
Full allowance	Qwen 9B	Gemma 31B	L/H	2	84.99	1.71	–
Full allowance	Qwen 9B	Gemma 31B	L/H	4	86.09	2.71	–
Full allowance	Granite 8B	GPT-OSS 120B	L/H	2	79.95	2.87	–
Full allowance	Granite 8B	GPT-OSS 120B	L/H	4	81.65	5.12	–
Full allowance	Granite 8B	GPT-OSS 120B	M/H	2	84.95	2.99	–
Full allowance	Granite 8B	GPT-OSS 120B	M/H	4	85.55	5.21	–
Full allowance	Granite 8B	Qwen 27B	L/H	2	86.39	17.66	0.16 [−0.11, 0.40]
Full allowance	Granite 8B	Qwen 27B	L/H	4	86.68	21.84	−0.33 [−0.65, −0.04]
Full allowance	Granite 8B	Gemma 31B	L/H	2	85.00	2.89	–
Full allowance	Granite 8B	Gemma 31B	L/H	4	85.83	5.12	–

The mixed-reference comparison purchases one low answer from each of two distinct reference models, then applies the same unanimity rule. We evaluate all three unordered reference pairs on all three targets, with the same GPT-OSS alternatives. Each run buys exactly two reference answers. Expected agreement averages the Cartesian product of their recorded answer identities, and both calls are charged. Each reference pair is used on every benchmark, without choosing a pair separately for each task. This comparison tests whether answers from different references also provide useful guidance.

Table 51: **One answer each from two reference models.** Costs are complete dollars per 1,000 final answers. Premium is the accuracy advantage over the target's full native-setting mixture at evaluated cost. Every reference combination is retained.

Quotes	References	Target	Choice	Acc.	Total $	Premium (pp)
Listed	GPT-OSS 20B + Qwen 9B	GPT-OSS 120B	L/H	83.47	1.45	–
Listed	GPT-OSS 20B + Qwen 9B	GPT-OSS 120B	M/H	85.75	1.50	–
Listed	GPT-OSS 20B + Qwen 9B	Qwen 27B	L/H	87.22	12.22	0.49 [0.33, 0.67]
Listed	GPT-OSS 20B + Qwen 9B	Gemma 31B	L/H	86.68	1.43	–
Listed	GPT-OSS 20B + Granite 8B	GPT-OSS 120B	L/H	82.70	2.04	–
Listed	GPT-OSS 20B + Granite 8B	GPT-OSS 120B	M/H	85.59	2.12	–
Listed	GPT-OSS 20B + Granite 8B	Qwen 27B	L/H	86.74	12.26	0.00 [−0.22, 0.24]
Listed	GPT-OSS 20B + Granite 8B	Gemma 31B	L/H	86.30	2.04	–

Quotes	References	Target	Choice	Acc.	Total $	Premium (pp)
Listed	Qwen 9B + Granite 8B	GPT-OSS 120B	L/H	81.83	2.37	–
Listed	Qwen 9B + Granite 8B	GPT-OSS 120B	M/H	85.53	2.44	–
Listed	Qwen 9B + Granite 8B	Qwen 27B	L/H	87.10	12.49	0.29 [0.07, 0.50]
Listed	Qwen 9B + Granite 8B	Gemma 31B	L/H	85.97	2.36	–
Full allowance	GPT-OSS 20B + Qwen 9B	GPT-OSS 120B	L/H	83.47	1.51	–
Full allowance	GPT-OSS 20B + Qwen 9B	GPT-OSS 120B	M/H	85.75	1.56	–
Full allowance	GPT-OSS 20B + Qwen 9B	Qwen 27B	L/H	87.22	20.09	0.54 [0.38, 0.71]
Full allowance	GPT-OSS 20B + Qwen 9B	Gemma 31B	L/H	86.68	1.50	–
Full allowance	GPT-OSS 20B + Granite 8B	GPT-OSS 120B	L/H	82.70	2.04	–
Full allowance	GPT-OSS 20B + Granite 8B	GPT-OSS 120B	M/H	85.59	2.12	–
Full allowance	GPT-OSS 20B + Granite 8B	Qwen 27B	L/H	86.74	19.64	0.14 [−0.07, 0.38]
Full allowance	GPT-OSS 20B + Granite 8B	Gemma 31B	L/H	86.30	2.05	–
Full allowance	Qwen 9B + Granite 8B	GPT-OSS 120B	L/H	81.83	2.43	–
Full allowance	Qwen 9B + Granite 8B	GPT-OSS 120B	M/H	85.53	2.50	–
Full allowance	Qwen 9B + Granite 8B	Qwen 27B	L/H	87.10	19.88	0.45 [0.24, 0.67]
Full allowance	Qwen 9B + Granite 8B	Gemma 31B	L/H	85.97	2.43	–

H.5 Answer identity audit

The archived shared string normalizer removed periods as well as whitespace, dollar signs and commas. This could conflate decimal numbers. We reconstructed all 244,773 valid identities that passed through that adapter and inspected many-to-one mappings within each question. Among 590,034 valid responses, 839 question–identity groups contain distinct unmodified extracted values, mostly formatting or structured-grid representations. Two merge different numerical values: `1.00`/`100` on one GSM8K question and `1.2`/`12` on one MATH500 question. These are also the only merged groups with differing recorded correctness. Correctness is used here to audit the signal, never to construct a runtime identity.

Retaining periods resolves both numerical collisions. Checking all 52,809 model–question–setting cells containing valid answers finds no change in their agreement partitions: the colliding values came from different models. The tested mixed-reference gates are also unchanged. Thus every reported single- and mixed-reference gate outcome is identical with decimal points preserved. The artifact retains the original identities and a separate decimal-preserving implementation and audit.

Coding bypasses this normalizer entirely. Its 82,626 valid responses use hashes of extracted programs that preserve internal whitespace, case, and punctuation. We find no within-question inconsistencies between these identities and the independent code-scoring keys. As a more conservative sensitivity, preserving all mathematical string case, punctuation and whitespace, together with exact structured values, gives Qwen four-probe accuracy of 87.54% at $13.31 rather than 87.57% at $12.96. The corresponding saving is 18.22%. This stricter test also splits benign representations, so it measures sensitivity to exact form rather than establishing semantic equivalence.

H.6 The strongest tested paid length-only comparison

The earlier learned controls use question length and the mean and standard deviation of reference output lengths, plus input lengths. We retain their four probe designs (one low, two low, low plus high, four low), 32/128 calibration questions, five held-out question-group folds, three calibration orders, two calibration response views, and twelve fixed dollar penalties. Ridge penalties come from the existing grid $\{0.01, 0.1, 1, 10, 100\}$ by three-fold grouped calibration-only validation. Each inner fit learns its own standardization and target-cost scale. Predicted gains are clipped to $[-1, 1]$. No reference correctness or target test outcomes enter the predictors.

We evaluate these same fitted regressions on *every* possible held-out probe subset so that their comparison with agreement uses identical averaging. All 2,400 original fits and their original-view choices are reconstructed before this extension. Calibration still uses only its specified paid responses. Evaluation charges each policy's probes, selected target answer, and reference/target calibration bill. The best tested curve is the descriptive upper envelope of all fixed penalty points and their mixtures, including both calibration sizes. This is a comparison of evaluated tradeoffs, not a new deployment procedure selected using test outcomes. Paired question resampling recomputes the

envelope. Mixtures across different calibration designs randomize the whole procedure before buying its sample, so their expected setup bill is also mixed.

Table 52: **Agreement exceeds the tested paid length-only curves.** Qwen 9B to 27B, listed prices, 1,000 later answers per task. Entries are agreement-minus-length accuracy points [paired 95%] at the agreement rule's complete cost. Each row includes all twelve fixed penalties and their mixtures.

Calibration questions	Paid length inputs	Two-probe advantage	Four-probe advantage
32	One low	0.61 [0.40, 0.81]	0.46 [0.22, 0.67]
32	Two low	0.75 [0.54, 0.96]	0.56 [0.32, 0.78]
32	Low + high	1.16 [0.92, 1.37]	0.88 [0.63, 1.12]
32	Four low	0.98 [0.75, 1.21]	0.72 [0.48, 0.97]
128	One low	1.19 [0.92, 1.47]	0.90 [0.60, 1.20]
128	Two low	1.36 [1.09, 1.66]	1.05 [0.76, 1.35]
128	Low + high	1.82 [1.52, 2.14]	1.41 [1.09, 1.75]
128	Four low	1.64 [1.36, 1.95]	1.32 [1.01, 1.64]

At 1,000 later answers, the full upper envelope gives advantages of 0.61 [0.40, 0.81] and 0.46 [0.22, 0.67] points for two and four probes. Under full-output-allowance quotes these become 0.64 [0.43, 0.84] and 0.54 [0.30, 0.77]. At 10,000 later answers under listed quotes they shrink to 0.32 [0.07, 0.54] and 0.22 [-0.03, 0.45]. Thus the four-probe advantage over the learned length controls, like its comparison with larger task-default calibration, is unresolved at the longer horizon.

The original nested regressions that add answer features to the same length features remain in the artifact. They did not consistently improve over their matched length-only predictors. Those regressions test the value of adding features to a learner. Table 52 asks a different question: whether the fixed agreement rule improves on the strongest tested length curves.

H.7 Other uses of the purchased answers

The main fixed-target rule buys reference answers only to choose one fresh target generation. We also evaluate two natural alternatives with the same fixed unanimity condition. A reference-return cascade returns its first cheap answer on valid unanimity and otherwise calls target high. A target-reuse rule buys two or four low-effort target answers, returns the first on unanimity, and otherwise generates one high-effort target answer. Neither rule uses labels, and every purchased response is charged. Target high responses are independent of the low responses used for the decision.

For an identity with c_j recorded occurrences, its unanimity probability is $\binom{c_j}{k}/\binom{R}{k}$. Expected returned correctness multiplies this probability by the average correctness of responses with that identity: the first response is uniformly distributed within the unanimous group. This evaluates returning an actual purchased answer rather than choosing a correct one with access to labels.

Table 53: **Answer reuse and early stopping change the operating point.** Qwen 9B probes Qwen 27B unless "reuse target" specifies target probes instead. Costs use listed prices and include all purchased calls. Sequential variants preserve the corresponding fixed-call decisions and accuracy.

Maximum probes	Rule	Accuracy (%)	$/1,000	Mean probes
2	Fresh target	87.02	10.70	2.00
2	Return reference	85.98	8.83	2.00
2	Reuse target	87.84	14.18	2.00
4	Fresh target	87.57	12.96	4.00
4	Return reference	87.42	11.61	4.00
4	Reuse target	88.21	23.33	4.00
4	Sequential reference	87.57	12.55	3.40
4	Sequential return reference	87.42	11.20	3.40
4	Sequential reuse target	88.21	19.94	3.59

The reference-return curve, including both probe counts and mixtures with native target settings, is competitive with fresh-target selection. The latter's same-cost advantages are -0.01 [-0.31, 0.29]

and -0.01 [-0.23, 0.22] points for two and four probes. Against the curve reusing target answers instead, the differences are 0.44 [0.21, 0.65] and 0.17 [-0.14, 0.47]. Allowing both reuse types yields -0.01 [-0.31, 0.27] and -0.07 [-0.28, 0.12]. The results therefore support useful agreement information in several inference systems, without establishing unique superiority of returning a fresh target answer.

Sequential acquisition. Invalidity or any disagreement makes unanimity impossible, so further probes can be omitted without changing the effort decision. Let S be the sum of the R recorded reference-call costs and $\bar{C}_j$ their mean within identity j. The expected bill for a policy allowing at most k probes is

$$\frac{S}{R} + \sum_{h=1}^{k-1} \sum_{j \in \text{valid}} \frac{\binom{c_j}{h}}{\binom{R}{h}} \frac{S - h\bar{C}_j}{R - h}. \tag{8}$$

The first call is always purchased. Call $h+1$ is purchased only when the first h answers are valid and unanimous. The final factor accounts for dependence between continuation and response cost. Direct enumeration of ordered response subsets checks this expression on every task. Four Qwen 9B probes require 3.40 calls on average, reducing the fresh-target bill to \$12.55 and increasing its saving to 22.86%, with exactly the original 87.57% accuracy. The fixed-call design permits parallel probes, while early stopping requires sequential decisions. Their latency has not been measured.

H.8 Relative-price sensitivity

We scale the reference's input and output prices by a common multiplier s, leaving target rates and every policy decision fixed. If C_T is the selected target cost and C_R the original complete probe cost, the new bill is $C_T + sC_R$. For a fixed comparison curve, let $C_B(A)$ be the least interpolated cost attaining the agreement rule's accuracy A. The break-even multiplier is $(C_B(A) - C_T)/C_R$. A nonpositive value means that even free probes do not improve that curve. No accuracy or cost is extrapolated outside supported ranges.

Table 54: **How much reference prices can rise before the comparison reverses.** The comparison uses Qwen 9B to guide Qwen 27B. Listed quotes correspond to a multiplier of one. Both reference rates scale together. Paired ranges resample questions and recompute interpolation. The final column reports defined draws out of 1,000.

Probes	Comparison curve	Later answers	Multiplier [95%]	Defined
2	Random native settings	1,000	4.29 [3.39, 5.34]	1000
2	32 target questions	1,000	3.29 [2.35, 4.39]	1000
2	32 target questions	10,000	2.52 [1.59, 3.62]	1000
2	128 target questions	1,000	4.71 [3.76, 5.62]	1000
2	128 target questions	10,000	1.61 [0.67, 2.49]	1000
4	Random native settings	1,000	2.31 [1.87, 2.83]	999
4	32 target questions	1,000	2.33 [1.75, 3.01]	999
4	32 target questions	10,000	1.95 [1.36, 2.62]	999
4	128 target questions	1,000	2.60 [1.78, 3.76]	999
4	128 target questions	10,000	1.05 [0.25, 2.21]	999

The four-probe threshold is 2.31 [1.87, 2.83] against random native-setting mixtures and 2.33 [1.75, 3.01] against 32-question task defaults at the 1,000-answer horizon. One resample puts gate accuracy beyond the comparison curve's range, so its threshold is undefined. At 10,000 answers, the 128-question control's threshold is only 1.05 [0.25, 2.21]. These calculations identify the value of a particular signal under fixed target prices, not a universal price ratio or a new measurement of provider behavior.

I Priced task-default controls

The agreement rule is compared with task defaults learned from separate labeled questions. This appendix specifies those controls and the frozen prices used for every paid call. Each later question receives one target answer. Costs include profiling as well as those final answers.

I.1 Reference and direct target calibration

The Qwen reference profile uses four answers per setting on about 32 labeled questions from Qwen 9B. The target supplies no calibration answers to this transfer rule. The reference's most accurate setting becomes the Qwen 27B task default, with exact accuracy ties broken by lower mean tokens. At 1,000 later answers per task this gives 87.64% accuracy at $14.61, including its setup bill, versus 87.99% at $16.27 for always high.

Calibration excludes every group in the held-out fold. Five group folds, three outcome-independent question orders, and two response views give six choices per evaluated question. Complete groups are added until at least 32 questions are included (mean 32.17, maximum 37). Four-response profiles use attempts 0–3 or 4–7. One-response profiles use attempt 0 or 1. All held-out target attempts evaluate the chosen setting. The source and target controls share question and response orderings.

For N later answers, mean total cost is

$$C_{\text{total}}(N) = C_{\text{chosen answer}} + C_{\text{profile}}/N. \tag{9}$$

A deployment pays for one fitted calibration set, not the five folds and six views used to evaluate the procedure. The 1,000- and 10,000-answer horizons extrapolate the measured mean request costs over future workloads, without adding new datasets. Benchmark labels are assumed available and their creation is not priced.

Direct calibration controls. The main comparison uses 32 or 128 separate target questions, one response per setting, and every calibration-derived task-default switching point. Appendix G gives the strengthened curves. Earlier dollar-allowance controls bought target pairs within the reference profile's bill and included a complete-pair overshoot sensitivity. Their complete outcomes and 200 fresh-selection checks remain in `matched_profile_budget_20260917` and the original agreement artifact. They concern that earlier acquisition procedure, rather than refits of the current 32/128-question controls.

I.2 Frozen prices and serving assumptions

For each model, provider selection uses the lowest paid, available quote for a fixed request containing 1,000 input and 10,000 output tokens. A single endpoint supplies both rates. Actual bills use every saved request's prompt and generated-token counts, including failed responses. Selected endpoints have no fixed request fees. We exclude free and batch rates as well as caching discounts. Reasoning tokens are charged as output (OpenRouter, 2026).

The listed scenario can select an endpoint with a shorter output allowance than the local experiment's 81,920 tokens. The full-allowance scenario requires reasoning support, at least 81,920 output tokens, and enough context for that allowance plus the maximum recorded prompt. Table 55 gives both for every model used in the agreement experiment. Matching those allowances does not establish identical quantization, prompting, accuracy, or latency across serving systems: these are frozen price scenarios applied to local generations. Raw quotes, timestamps, URLs and hashes accompany the artifact.

Table 55: **Frozen paid-provider quotes for the agreement experiment, 17 September 2026.** USD per million tokens. "Full cap" requires the local experiment's output allowance and prompt context. Bills use actual input and output counts rather than the request length used to select a provider.

Model	Scenario	Provider	Input	Output	Max. output
GPT-OSS 20B	Listed	AkashML	0.02	0.10	117,964
GPT-OSS 20B	Full cap	AkashML	0.02	0.10	117,964
GPT-OSS 120B	Listed	AkashML	0.03	0.17	117,964
GPT-OSS 120B	Full cap	AkashML	0.03	0.17	117,964
Qwen 9B	Listed	Darkbloom	0.08	0.13	65,536
Qwen 9B	Full cap	DeepInfra	0.1	0.15	81,920
Qwen 27B	Listed	Alibaba	0.195	1.56	65,536
Qwen 27B	Full cap	DeepInfra	0.26	2.60	81,920
Granite 8B	Listed	CoreWeave	0.1	0.15	117,964
Granite 8B	Full cap	CoreWeave	0.1	0.15	117,964
Gemma 31B	Listed	DeepInfra	0.09	0.34	16,384
Gemma 31B	Full cap	CoreWeave	0.1	0.34	235,929

J PURCHASING MORE INFORMATION ABOUT EFFORT

Maximum worthwhile probe bill. For a rule with accuracy A, target-answer cost C_T, and probe bill P, let $C_B(A)$ be the least supported cost of a specified baseline reaching at least A. Then

$$V_B = C_B(A) - C_T \tag{10}$$

is the maximum probe bill compatible with a strict cost advantage when $P < V_B$. If $V_B \leq 0$, there is no strict cost advantage even with free probes; if $0 < V_B < P$, the target choices are useful relative to that baseline but the probes are overpriced. This accounting depends on the baseline and its interpolation support. When the accuracy is not supported, the comparison remains undefined rather than being extrapolated.

This appendix examines how much agreement information is worth buying and how acquisition changes its cost. The original two- and four-probe rules anchor the comparison. The expanded counts and references use the same response bank, prices, and question-group resampling.

J.1 FIXED RULES AND THE EXPANDED SCREEN

The screen uses all eight models with frozen quotes, their non-high reference settings, and each target's available lower-to-high choice. EXAONE remains unpriced. Reference and target models must differ. Single-reference rules use $K \in \{2, 3, 4, 6, 8\}$, with either parallel or sequential acquisition. Mixed rules purchase one or two answers from each of two distinct reference models, at their non-high settings, and require every purchased answer to be valid and identical. The target differs from both references. Every rule applies unchanged to all ten tasks, and every target's native settings enter its comparison curve. This gives 1,868 screened rules per price scenario. The additional focused comparisons use a two–two–four schedule for Qwen probes, alternating Qwen and Gemma probes, and expanded answer-reuse rules. Complete results, including failures and unsupported comparisons, are saved in the artifact.

All rules use the decimal-preserving identities described in Appendix H.5. For k_1 and k_2 answers from two references, the unanimity probability is

$$u = \sum_{j \in \text{valid}} \frac{\binom{c_{1j}}{k_1}}{\binom{R_1}{k_1}} \frac{\binom{c_{2j}}{k_2}}{\binom{R_2}{k_2}}.$$

Invalid answers remain in R_1 and R_2. They contribute to no valid identity. No reference correctness or target outcome is used to choose effort.

J.2 SEQUENTIAL AND BATCHED ACQUISITION

Let S be the sum of the R recorded reference-call costs and $\bar{C}_j$ their mean within valid identity j. A further call is purchased only when the first h answers are valid and unanimous. Its expected

contribution to the bill is

$$D_h = \sum_{j \in \text{valid}} \frac{\binom{c_j}{h}}{\binom{R}{h}} \frac{S - h\bar{C}_j}{R - h}.$$

Writing $u_h = \sum_j \binom{c_j}{h} / \binom{R}{h}$, sequential acquisition costs $S/R + \sum_{h=1}^{K-1} D_h$ and purchases $1 + \sum_{h=1}^{K-1} u_h$ calls in expectation. The two–two–four schedule costs $2S/R + 2D_2 + 4D_4$ and purchases $2 + 2u_2 + 4u_4$ calls. These expressions retain the dependence between continuation and response length. Multiplying average call count by unconditional average call cost would discard that dependence.

Both strategies preserve the corresponding fixed-K target choices and accuracy. At $K = 8$, fully sequential Qwen acquisition costs \$13.98 per 1,000 final answers, compared with \$15.42 for parallel acquisition and \$14.08 for the three-batch schedule. Their common accuracy is 87.84%. The batch saving is 13.48% [12.02, 14.98] relative to always high. Table 57 reports the full sequential continuum with paired uncertainty. Fewer calls require serial dependencies, and no provider latency has been measured.

Table 56: **Early stopping preserves the decision and reduces acquisition cost.** Qwen 9B guides Qwen 27B. Accuracy is identical for parallel and sequential acquisition at each maximum. Mean probes and saving refer to sequential acquisition. Costs include every call, in dollars per 1,000 final answers. Always high reaches 87.99% at \$16.27. The original counts are two and four.

Maximum probes	Mean probes	Accuracy (%)	Parallel cost	Sequential cost	Saving (%)
2	2.00	87.02	10.70	10.70	34.24
3	2.73	87.39	12.05	11.86	27.11
4	3.40	87.57	12.96	12.55	22.86
6	4.64	87.74	14.33	13.42	17.51
8	5.79	87.84	15.42	13.98	14.08

Table 57: **Accuracy and the amount worth paying for guidance change differently.** Qwen 9B guides Qwen 27B under listed prices. Loss and saving are relative to always high. V_B uses the native-setting mixture. Bills are dollars per 1,000 answers. Intervals are paired 95% ranges. The defined count concerns V_B, whose accuracy constraint can exceed the baseline's support in a resample. Saving refers to sequential acquisition; the final two columns report probe-only bills.

K	Accuracy loss (pp)	Saving (%)	V_B [95%]	Defined	Parallel probe bill	Sequential probe bill
2	0.97 [0.64, 1.30]	34.24 [32.60, 36.00]	3.20 [2.56, 4.01]	1000	0.75	0.75
3	0.60 [0.32, 0.86]	27.11 [25.52, 28.77]	3.42 [2.77, 4.23]	1000	1.12	0.93
4	0.42 [0.17, 0.66]	22.86 [21.30, 24.47]	3.45 [2.80, 4.22]	999	1.49	1.08
6	0.25 [0.04, 0.45]	17.51 [16.04, 19.04]	3.38 [2.74, 4.01]	990	2.24	1.33
8	0.16 [−0.04, 0.36]	14.08 [12.64, 15.56]	3.34 [2.67, 3.84]	935	2.99	1.55

J.3 Target tokens, probe tokens, and dollar costs

The dollar saving comes from substituting cheaper reference computation for more expensive target computation. Table 58 separates these components for the original four-probe rule and its sequential version. Both use about 30% fewer target output tokens than always high. The probe outputs raise the combined count above always high, even though the dollar bill falls. Different tokenizers and serving prices make the two measures distinct; summed tokens do not measure equal hardware work.

The calculation applies the conditional acquisition formula above separately to input and output tokens. An independent calculation instead weights each recorded response by its probability of being purchased. Ordered enumeration on real and synthetic questions verifies missing answers, disagreement, and variable response lengths. The artifact's `agreement_cost_components_20260918` module saves every task and paired draw, including both price scenarios. Its results reproduce the original accuracy and dollar draws exactly.

Table 58: **Cheaper probes replace target reasoning.** Output tokens per final answer and dollars per 1,000 final answers under listed prices. Dollars include input as well as output tokens for every purchased call. Four together and early stopping make the same target choices and reach 87.57% accuracy. Point estimates average the ten tasks equally.

	Output tokens / answer			Dollars / 1,000 answers		
Procedure	Target	Probes	Combined	Target	Probes	Total
Always high	10,356	0	10,356	16.27	0.00	16.27
Four together	7,278	10,029	17,307	11.47	1.49	12.96
Up to four, stop early	7,278	7,269	14,547	11.47	1.08	12.55

J.4 Economic comparisons and uncertainty

All averages weight benchmarks equally. Paired intervals use 1,000 question-group bootstrap draws with the original task order and seeds. Costs, accuracy, and interpolation are recomputed with common weights. The comparison curves include free native mixtures, every saved 32/128-question task-default policy, the strongest saved paid length envelope, and their specified mixtures. Setup bills remain charged at each stated horizon.

For each paired draw, $C_B(A)$ is the least supported cost of a fixed point or mixture attaining at least accuracy A. The three economic cases follow from $V_B = C_B(A) - C_T$: a nonpositive value, a positive value below the actual probe bill, or a value above that bill. No frontier is extrapolated. At eight Qwen probes, V_B is defined in 935 of 1,000 draws for the native mixture because some resamples put gate accuracy above every baseline setting. By contrast, all 1,000 accuracy comparisons at evaluated cost in Table 59 are defined. Their intervals describe the evaluated comparisons under the resampling convention in the study design.

The direct-calibration controls here and in the task-omission comparison below use the original twelve-penalty defaults. The main-text comparison uses the stronger exhaustive defaults.

Table 59: **Strong controls across acquisition horizons and prices.** Accuracy advantage in points [paired 95%] at the rule's evaluated total cost. "Full" uses quotes supporting the study's output allowance. Direct controls use 32 or 128 labeled target questions. The length column uses the full saved paid envelope. Every comparison has 1,000 defined draws.

Quotes	K	Acquisition	Horizon	Native mixture	Direct 32	Direct 128	Paid length
Listed	4	Parallel	1,000	0.61 [0.41, 0.82]	0.43 [0.22, 0.63]	0.75 [0.43, 1.02]	0.46 [0.22, 0.67]
Listed	4	Parallel	10,000	0.61 [0.41, 0.82]	0.30 [0.11, 0.51]	0.01 [−0.25, 0.28]	0.22 [−0.03, 0.45]
Listed	8	Sequential	1,000	0.56 [0.35, 0.76]	0.47 [0.27, 0.68]	0.62 [0.31, 0.92]	0.50 [0.28, 0.70]
Listed	8	Sequential	10,000	0.56 [0.35, 0.76]	0.35 [0.15, 0.54]	0.14 [−0.10, 0.37]	0.29 [0.07, 0.49]
Listed	8	2–2–4	1,000	0.53 [0.33, 0.73]	0.45 [0.25, 0.66]	0.59 [0.28, 0.89]	0.47 [0.26, 0.68]
Listed	8	2–2–4	10,000	0.53 [0.33, 0.73]	0.33 [0.13, 0.52]	0.13 [−0.11, 0.36]	0.27 [0.05, 0.47]
Full	4	Parallel	1,000	0.75 [0.55, 0.97]	0.52 [0.30, 0.74]	0.86 [0.53, 1.16]	0.54 [0.30, 0.77]
Full	4	Parallel	10,000	0.75 [0.55, 0.97]	0.40 [0.19, 0.60]	0.12 [−0.16, 0.40]	0.29 [0.04, 0.53]
Full	8	Sequential	1,000	0.70 [0.50, 0.91]	0.57 [0.35, 0.79]	0.79 [0.46, 1.11]	0.58 [0.35, 0.80]
Full	8	Sequential	10,000	0.70 [0.50, 0.91]	0.45 [0.25, 0.65]	0.19 [−0.06, 0.46]	0.35 [0.11, 0.57]
Full	8	2–2–4	1,000	0.68 [0.48, 0.89]	0.56 [0.33, 0.78]	0.76 [0.43, 1.09]	0.57 [0.33, 0.79]
Full	8	2–2–4	10,000	0.68 [0.48, 0.89]	0.43 [0.23, 0.64]	0.18 [−0.07, 0.44]	0.33 [0.10, 0.56]

At 1,000 answers, eight sequential probes improve over the direct-32 curve by 0.47 [0.27, 0.68] points and over direct-128 by 0.62 [0.31, 0.92]. At 10,000 answers, the latter difference is 0.14 [−0.10, 0.37], so the interval includes zero. Under full-output-allowance prices, eight sequential probes save 16.96% [15.50, 18.49] against always high. Their advantage over the paid length curve at 1,000 answers is 0.58 [0.35, 0.80] points. The signal's price limit is specific to the alternative and horizon.

J.5 Do additional probes improve on smaller-probe policies?

More probes increase the frequency of high effort as well as the information available for choosing it. We compare each expanded count with random mixtures of all smaller-probe policies and native low/high settings. Both parallel and sequential acquisition enter the comparison. Every policy retains its complete bill. At each evaluated total cost, we find the best accuracy among these mixtures and

subtract it from the larger-count policy's accuracy. This comparison is repeated in every paired question-group draw.

Table 60 shows that early stopping changes the return to purchasing more evidence. Under listed prices, six sequential probes improve on smaller-policy mixtures by 0.07 [0.01, 0.14] points. Eight improve by 0.05 [−0.01, 0.10]. Against four sequential probes mixed only with native settings, eight improve by 0.103 [0.004, 0.210] points. The smaller advantage against the complete smaller-count comparison reflects the useful six-probe option. Eight in two–two–four batches give 0.04 [−0.02, 0.10] points beyond the complete smaller-count comparison. All these accuracy comparisons have 1,000 defined draws. The artifact also reports acquisition-matched comparisons, preserving the distinction between parallel calls and serial dependencies.

Table 60: **Accuracy above mixtures of smaller-probe policies and native settings.** Differences are accuracy points [paired 95%] at each rule's total cost. Every row has 1,000 defined draws. Full allowance uses the frozen provider scenario compatible with the study's output allowance.

Prices	Maximum probes	Buy together	Stop at disagreement
Listed	3	0.14 [0.07, 0.21]	0.17 [0.10, 0.24]
Listed	4	0.03 [−0.03, 0.08]	0.08 [0.04, 0.13]
Listed	6	−0.03 [−0.13, 0.07]	0.07 [0.01, 0.14]
Listed	8	−0.08 [−0.23, 0.06]	0.05 [−0.01, 0.10]
Full allowance	3	0.17 [0.09, 0.24]	0.19 [0.11, 0.26]
Full allowance	4	0.06 [0.01, 0.11]	0.10 [0.05, 0.14]
Full allowance	6	0.02 [−0.06, 0.10]	0.09 [0.03, 0.15]
Full allowance	8	−0.02 [−0.12, 0.07]	0.06 [0.01, 0.11]

The maximum worthwhile bill V_B asks a different question: how much could be spent on probes before losing the advantage over a specified baseline? Table 61 reports paired changes against native mixtures. From four to eight, the listed-price estimate changes by −\$0.12 [−0.43, 0.20] per 1,000 answers. This interval does not establish saturation. The point estimates stay near \$3.4 while the actual probe bills continue to increase. Undefined inverse comparisons occur when the estimated policy accuracy exceeds the native mixture's support. We retain those counts and use only jointly defined draws for changes.

Table 61: **Paired changes in the maximum worthwhile probe bill.** The baseline is the native low/high mixture. Changes are dollars per 1,000 answers [paired 95%]. The number of defined draws is reported explicitly.

Prices	Count change	Change in V_B	Defined
Listed	2 to 4	0.25 [−0.11, 0.55]	999
Listed	4 to 6	−0.07 [−0.25, 0.11]	990
Listed	4 to 8	−0.12 [−0.43, 0.20]	935
Listed	6 to 8	−0.04 [−0.19, 0.11]	935
Full allowance	2 to 4	0.42 [−0.18, 0.92]	999
Full allowance	4 to 6	−0.12 [−0.42, 0.18]	990
Full allowance	4 to 8	−0.19 [−0.72, 0.33]	935
Full allowance	6 to 8	−0.07 [−0.31, 0.18]	935

J.6 Economic robustness to task composition

We omit each benchmark in turn, average the remaining nine equally, and recompute costs and comparison curves. Each calibration bill is averaged over the retained tasks before amortization. Task-specific fits remain fixed. The original two/four-probe rules and both acquisition strategies are evaluated under both frozen price scenarios and both acquisition horizons.

Table 62 gives the original four-parallel-probe result at 1,000 later answers per task. Every single-task omission retains positive conditional intervals against all four baselines. Savings against always high range from 16.21% to 29.18%, with accuracy losses from 0.32 to 0.50 points. Removing LiveCodeBench increases the saving because exact program agreement almost always selects high

while still charging for probes. At 10,000 answers, the difference from 128-question calibration ranges from -0.05 to 0.15 points and every omission interval includes zero. Workload robustness therefore coexists with the change in competitiveness over longer use. The complete outcomes, including all other rules, scenarios, and horizons, are saved in the artifact.

Table 62: **The shorter-horizon economic advantage survives every task omission.** Four Qwen 9B answers together guide Qwen 27B under listed prices. The first column reports saving relative to always high. Other columns report accuracy advantages at the same total cost [paired 95%], with setup spread over 1,000 answers per retained task. All comparisons have 1,000 defined draws.

Omitted task	Saving (%)	Random (pp)	Direct 32 (pp)	Direct 128 (pp)	Length (pp)
GSM8K	18.92 [17.26, 20.61]	0.58 [0.37, 0.79]	0.41 [0.20, 0.63]	0.76 [0.41, 1.06]	0.44 [0.20, 0.67]
MATH500	18.24 [16.57, 19.96]	0.56 [0.35, 0.80]	0.38 [0.17, 0.60]	0.66 [0.33, 0.95]	0.43 [0.19, 0.66]
GPQA	16.21 [14.74, 17.91]	0.45 [0.25, 0.66]	0.43 [0.22, 0.63]	0.98 [0.70, 1.25]	0.46 [0.24, 0.67]
MMLU-Pro	18.79 [17.11, 20.50]	0.56 [0.34, 0.79]	0.40 [0.18, 0.62]	0.81 [0.45, 1.10]	0.44 [0.20, 0.67]
MuSR	20.01 [18.32, 21.77]	0.73 [0.54, 0.93]	0.51 [0.28, 0.71]	0.84 [0.52, 1.14]	0.55 [0.29, 0.77]
ReClor	19.35 [17.69, 21.06]	0.67 [0.46, 0.90]	0.47 [0.23, 0.69]	0.77 [0.43, 1.08]	0.50 [0.24, 0.73]
ProofWriter	19.03 [17.37, 20.77]	0.66 [0.46, 0.89]	0.43 [0.22, 0.64]	0.75 [0.43, 1.03]	0.49 [0.25, 0.71]
ZebraLogic	21.09 [19.35, 23.00]	0.59 [0.37, 0.82]	0.37 [0.15, 0.58]	0.62 [0.29, 0.91]	0.35 [0.12, 0.56]
CogniLoad	24.35 [22.37, 26.44]	0.68 [0.47, 0.91]	0.43 [0.20, 0.64]	0.65 [0.31, 0.97]	0.43 [0.18, 0.66]
LiveCodeBench	29.18 [27.16, 31.13]	0.52 [0.31, 0.72]	0.46 [0.20, 0.66]	0.63 [0.28, 0.91]	0.46 [0.19, 0.69]

The separate `agreement_incremental_value_20260918` module reconstructs these comparisons from saved per-question outcomes. It recovers every original pooled curve, independently aggregates question-group totals, and checks interpolation using convex hulls and linear programming. These checks validate the reaggregation, while the resampling intervals remain conditional on the existing response bank and fixed policies.

J.7 Reference breadth and unsuccessful targets

Several references guide Qwen 27B usefully, including cheap settings of Gemma, GPT-OSS 120B, and Granite. Mixed references also provide useful signals, but their incremental benefit over the expanded Qwen-only curve has intervals including zero. The complete reference and mixed-reference tables remain in `agreement_economics_followup_20260918/results/`. The target-level screen below retains the more consequential distinction: most target alternatives leave no economic advantage for these purchases.

Table 63: **Economic success remains concentrated on the Qwen 27B target.** The complete listed-price screen includes all native target settings in the baseline. Counts classify point estimates by V_B and the actual bill P. The final column counts conditional accuracy intervals above zero at comparable cost. These are overlapping rules, not independent observations or multiplicity-adjusted discoveries.

Target	Rules	$V_B \leq 0$	$0 < V_B < P$	$V_B > P$	Undefined	Interval > 0
GPT-OSS 20B	316	215	101	0	0	0
GPT-OSS 120B	316	196	120	0	0	0
Qwen 9B	184	128	56	0	0	0
Qwen 27B	184	0	43	141	0	83
GLM	184	184	0	0	0	0
Nemotron	184	184	0	0	0	0
Granite 8B	316	208	108	0	0	0
Gemma 31B	184	2	180	2	0	0

For example, Qwen 9B probes selecting GPT-OSS 120B medium or high give $V_B = -\$0.07$ per 1,000 answers against the full native mixture. Free probes would not make that fixed choice competitive. The GPT-OSS sibling rule instead has a positive value of about \$0.03 but pays about \$0.14 for its probes. The negative cases distinguish an unsuitable selection rule from an overpriced useful signal. Fixing the target also limits the decision problem: always-high Gemma gives 87.95% at \$1.35, so the Qwen policies do not establish superiority over unrestricted model choice.

J.8 Expanded answer reuse

Return-reference and target-answer-reuse alternatives receive the same expanded probe limits and acquisition schedules. At eight sequential reference probes, the fresh-target rule is 0.06 [-0.10, 0.23] points above the expanded return-reference curve at its cost. The original four-probe reuse comparison is in Table 5. Complete operating points, including negative comparisons, remain in the artifact's saved policy outcomes and paired arrays. The fixed-target decision does not have a demonstrated advantage over using a purchased answer as the final response.

Independent checks enumerate ordered response sequences, including invalid answers and unequal costs, and mixed-reference Cartesian products. They recover the original two/four-probe and native-setting draws, reconstruct paid outcomes from raw records, and reaggregate all saved question values with separate group-bootstrap code. Linear programs independently check the interpolated accuracy and minimum-cost comparisons. These tests verify numerical accounting and information access.

K Additional diagnostic displays

These displays provide the full information-access and native-setting summaries supporting Sections 2–4. They also show setting profiles, the location of additional tokens, and the prices at which guidance improves a target's alternatives.

Table 64: **Information available before choosing effort.** Earlier answers measure information value; forecasts and agreement guide new questions. Task defaults average the responses available in each experiment. Answer keys score responses and are never passed to the answering model. Section 5 charges guidance acquisition.

Experiment	Information available before choosing	What the comparison measures
Earlier answers	Target responses to this exact question, scored for correctness	Value of measured gains and costs; acquisition excluded
Text forecasting	Question text; training and target calibration on other questions	Value of learned predictions
Answer agreement	Cheap-model responses to this question, without correctness labels	Value of purchased guidance
Task default	Benchmark averages of the available scored target responses	Value of one setting per benchmark

Table 65: **The accuracy bought by each native effort increase.** Return is accuracy points gained per 1,000 additional tokens [paired 95%], using equal-task averages. Granite's additional choice is included. Appendix B gives successive GPT-OSS increases, remaining-error comparisons, and reciprocal token prices.

Model	Change	Starting accuracy (%)	Gain (pp)	Added tokens	Return (pp per 1,000 added tokens)
Native low/off to high/on					
GPT-OSS 20B	low → high	65.0	16.43	12,088	1.36 [1.25, 1.47]
GPT-OSS 120B	low → high	73.6	12.99	6,704	1.94 [1.76, 2.12]
Qwen 9B	off → on	76.2	5.72	7,828	0.73 [0.63, 0.82]
Qwen 27B	off → on	84.1	3.89	7,984	0.49 [0.41, 0.57]
GLM	off → on	52.5	23.93	8,511	2.81 [2.60, 3.04]
EXAONE	off → on	69.9	10.33	6,532	1.58 [1.40, 1.75]
Nemotron	off → on	42.1	35.83	7,754	4.62 [4.25, 5.05]
Granite	off → on	71.3	4.30	4,248	1.01 [0.84, 1.20]
Gemma	off → on	81.9	6.05	2,920	2.07 [1.80, 2.34]
Granite's additional native choice					
Granite	brief → off	57.7	13.56	6,357	2.13 [1.93, 2.34]

How much can a selector afford to spend on probes? For agreement accuracy A, let $C_B(A)$ be the least per-answer cost at which a specified baseline reaches at least A. The margin available for probes above the target-generation bill C_T is

$$V_B = C_B(A) - C_T. \tag{11}$$

The complete procedure is cheaper at that accuracy exactly when $C_P < V_B$. The threshold changes as calibration becomes cheaper to reuse or a purchased answer can be returned; if $V_B \leq 0$, even free probes give no strict cost advantage.

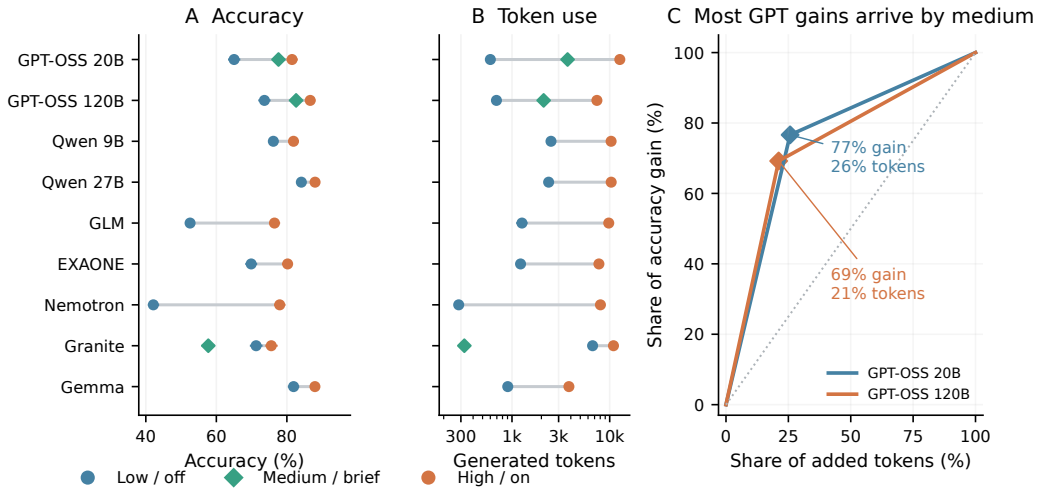


Figure 15: **Medium effort captures most GPT-OSS improvement at a minority of the added tokens.** Panels A and B show each model's mean accuracy and generated tokens across ten tasks. Intervals resample question groups. Green diamonds mark medium or brief where available. Granite brief uses fewer tokens and has lower accuracy than off. Panel C compares medium's share of the low-to-high accuracy gain with its share of added tokens. Shares are calculated from task-averaged changes, and the dotted diagonal marks equal shares.

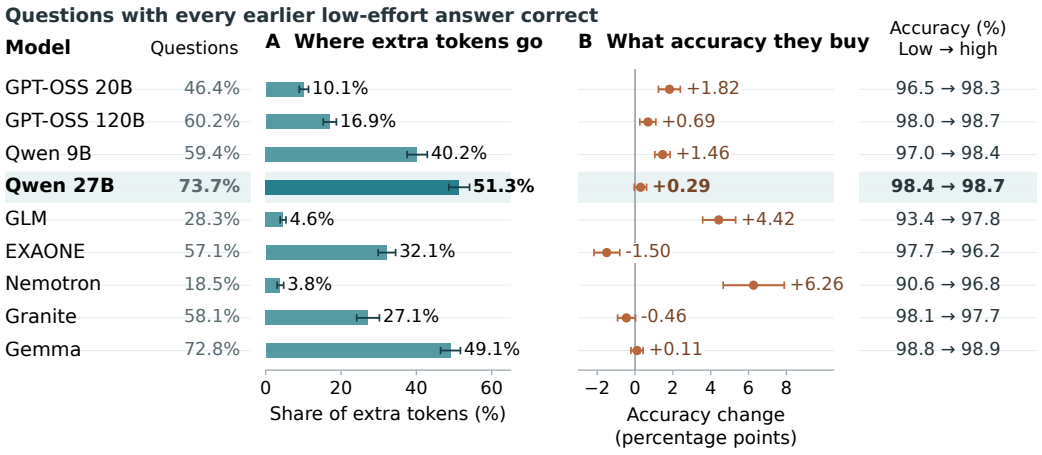


Figure 16: **Where extra tokens go, and what they buy.** Each row concerns questions whose earlier low/off answers were all correct. A: the share of the model's additional tokens spent on these questions. B: their accuracy change in percentage points, with low- and high-effort accuracy alongside. The Questions column gives the group's share of questions. Intervals are paired 95%. Both settings are evaluated on answers excluded from grouping. All nine models compare low/off with high/on, including Granite off-to-on. Negative gains are retained. Token shares concern the increase, not the complete bill.

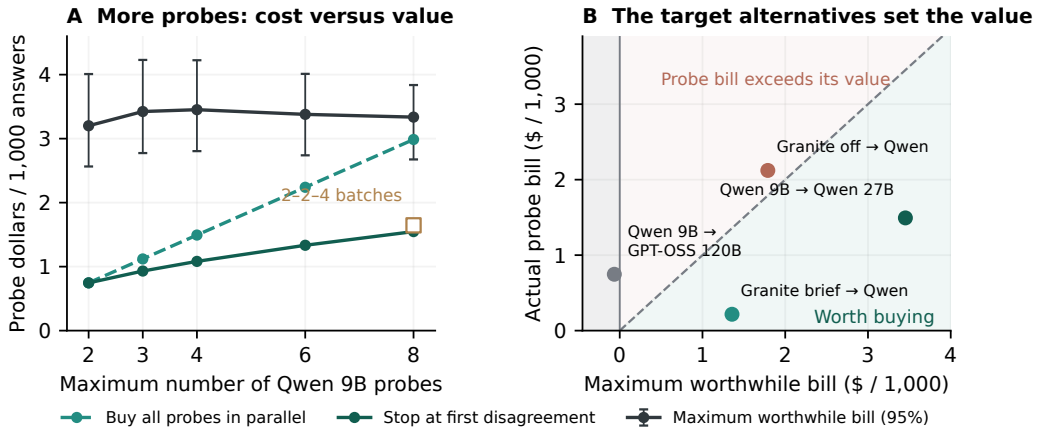


Figure 17: **Information has a price limit set by the target's alternatives.** A: maximum worthwhile Qwen probe bills against native mixtures, with paired 95% intervals, alongside actual acquisition bills. Intervals use supported draws: 935–1,000 depending on count. Table 61 reports paired changes. B: representative rules against complete native-setting mixtures. Below the diagonal, the purchase improves the tradeoff. Above it, the signal is overpriced. A nonpositive horizontal value means no strict cost advantage even with free probes. Qwen 9B to 27B uses four parallel answers, and the other cases use two. Panel B shows point estimates; Appendix J reports uncertainty and undefined comparisons.