THE CONDITIONAL VALUE OF COMPUTE: WHEN IS ADAPTIVE INFERENCE WORTH IT?

Anonymous authors
Paper under double-blind review

ABSTRACT

More test-time compute can improve LLM performance, but a fixed budget makes this benefit selective: which inputs should receive the extra computation? Average quality–cost curves do not answer this, and difficulty or confidence can be misleading because inputs that are hard for a cheap action may also be hard for a more expensive one. We therefore study the expected improvement from additional computation conditioned on information available before the decision—the *conditional value of compute*. We derive the optimal allocation rule across budgets, characterize what routing information must preserve to support these decisions, and quantify losses from estimation error and from paying to acquire the routing signal. On four SWE-Gym judge pairs, learned routing improves held-out trajectory-judgment accuracy by 3.92–8.35 percentage points over random allocation when 30% of inputs receive the more expensive judge, with positive gains in all 44 leave-one-repository-out evaluations. On RouterBench, question-text routing improves actual answer correctness by 3.32–6.02 points on three representative model pairs; 1.37–2.64-point gains remain when the upgrade budget is fixed separately within each task. Complementary experiments reinforce a second distinction: a signal can improve targeting yet may not be worth acquiring once its own cost and competing cheap actions are counted. Together, these results support a simple principle: allocate compute according to predicted incremental return, not difficulty alone.

1 INTRODUCTION

Test-time computation can improve an answer by sampling another solution, retrieving evidence, running a verifier or tool, or consulting a larger model (Wang et al., 2023; Cobbe et al., 2021; Lightman et al., 2024; Snell et al., 2025). When additional computation is available for every input, its average benefit is a natural quantity to measure. Under a fixed budget, however, the question changes: **which inputs should receive the extra computation?**

Average quality–cost curves do not answer this allocation question, and input difficulty alone is not enough. Consider a cheap action A and a more expensive action B. An input on which A is uncertain may still be a poor target for additional computation if B is likely to fail as well. Conversely, a confident prediction from A can be worth revisiting when B has complementary strengths. This distinction is familiar in model cascades (Jitkrittum et al., 2023), but it also applies to retrieval, verification, tool use, and additional sampling. The relevant quantity is therefore not simply whether the current computation is likely to be wrong, but **how much the available next computation is expected to improve it**.

We formalize this quantity as the *conditional incremental advantage*

$$g(F) = \mathbb{E}[U_B - U_A \mid F],$$

where U_A and U_B are the utilities of two inference actions and F is the information available before choosing B. If a budget allows the expensive action on only a fraction r of inputs, the optimal policy prioritizes those with the largest conditional advantages. We compare this targeted allocation with random upgrading at the same rate; the resulting curve $\mathcal{R}_F(r)$ measures how much value the available information contributes beyond simply buying more computation.

This view separates three questions. **How much value can selective allocation create?** This depends on predictable variation in $g(F)$, not only on its mean. **What information must a router**

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.

preserve? Choosing the better action requires less information than ranking inputs under a fixed budget, while supporting all budgets requires preserving the relevant ordering. **Is that information worth obtaining?** A signal can improve allocation after it is observed yet still cost more to acquire than the value it creates. These distinctions connect the population allocation problem to learned routing and deployment cost.

We first test whether incremental returns are predictable enough to improve allocation. On SWE-Gym, where Qwen2.5-Coder models judge software-repair trajectories, learned routing improves trajectory-judgment accuracy by **3.92–8.35 percentage points** over random allocation at a 30% upgrade budget across four model pairs. The signal also transfers: direct-advantage estimates remain positive in all **44 leave-one-repository-out evaluations**. Richer lexical and pretrained representations recover the same qualitative structure, while no single estimator universally dominates; once the cheap judgment is available, error prediction can recover nearly the same useful ordering as direct advantage estimation. The empirical claim is therefore about **predictable incremental returns**, not a particular regression method.

We next ask whether the same principle improves actual model answers. On RouterBench, routing from question text improves held-out answer correctness by **3.32–6.02 points** for three representative model pairs. To control for differences between benchmark categories, we also fix the 30% upgrade budget separately within every task; **1.37–2.64-point** gains remain. Thus, the benefit is not explained only by moving calls between task categories: useful variation also exists among questions within the same task.

Finally, better allocation does not necessarily imply worthwhile spending. If obtaining a routing signal itself consumes computation, its targeting benefit must repay the calls that acquisition displaces. The available action set matters as well: on SWE-Gym, an inexpensive text-only classifier is already a strong alternative, and our residual-routing and output-fusion controls do not establish an additional balanced-accuracy gain beyond it. Sequential sampling follows the same principle—continuing is valuable only relative to stopping now and to the future actions that continuation enables. Adaptive inference therefore depends jointly on **the actions being compared, the information available before the decision, and the cost of obtaining that information**.

Our contributions. We address three questions:

- **What quantity should guide adaptive inference?** We frame compute allocation around the conditional incremental return $g(F)$, separating input difficulty from upgrade value and characterizing allocation and information requirements across budgets.
- **Is this return predictable enough to improve allocation?** Learned routing yields 3.92–8.35-point gains on SWE-Gym and 3.32–6.02-point gains in actual RouterBench answer correctness, including repository-transfer and within-task controls.
- **When does better allocation justify additional spending?** We account for routing-information cost and inexpensive alternatives, showing why matched-budget allocation value need not translate directly into deployment value.

2 Related Work

Model routing and cascades. Routing selects a model from the input; cascading can also use an existing response. Cost-aware selection uses quality predictions (Šakota et al., 2024; Ding et al., 2024), preference data (Ong et al., 2025), and routing benchmarks (Hu et al., 2024). Cascades exploit confidence, reasoning traces, and token uncertainty (Chen et al., 2024; Aggarwal et al., 2024; Yue et al., 2024; Gupta et al., 2024); online learning uses expert demonstrations (Nie et al., 2024). Jitkrittum et al. (2023) characterize relative-correctness deferral and introduce the Diff-01 target used by accuracy-based advantage regression. Dekoninck et al. (2025) unify routing and cascading. We connect these rules to allocation value across budgets and information-acquisition cost.

Selective prediction and learning to defer. Selective prediction trades coverage against accepted-decision risk (Chow, 1970; El-Yaniv & Wiener, 2010; Geifman & El-Yaniv, 2017; 2019). Learning to defer accounts for the downstream decision-maker (Madras et al., 2018), with consistent surrogates (Mozannar & Sontag, 2020), calibrated expert probabilities (Verma & Nalisnick, 2022), and

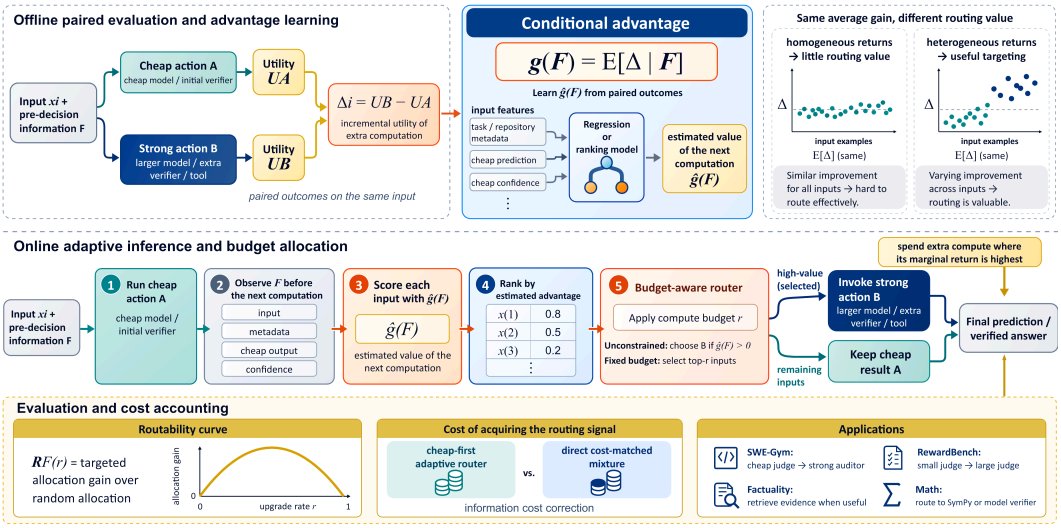

Figure 1: **The conditional value of compute.** Paired outcomes and pre-decision information support conditional incremental advantage estimation and budget allocation. The upper-right schematic illustrates predictable heterogeneity. Difficulty need not imply upgrade value; better targeting need not justify spending once signal cost and available alternatives are counted. Features and applications are illustrative; each experiment specifies its information set.

multi-expert extensions (Mao et al., 2023). Our expanded-action controls likewise compare extra calls with inexpensive final decisions.

Inference scaling and the value of computation. Sampling, verification, and search change the return to inference effort (Wang et al., 2023; Brown et al., 2024; Snell et al., 2025; Wu et al., 2024; Chen et al., 2023; Lightman et al., 2024; Rohatgi et al., 2026). Their allocation connects to classical information value (Howard, 1966) and metareasoning (Russell & Wefald, 1991): an observation or computation is valuable through the decisions it improves. Optimal observation selection (Krause & Guestrin, 2009) and adaptive submodularity (Golovin & Krause, 2011) study sequential acquisition under structural assumptions. We focus on the conditional utility difference between declared compute actions, separating its budget allocation value from the expenditure needed to observe a routing signal.

Prioritization and policy evaluation. Conditional advantage is the treatment effect of a compute action. Capacity-constrained effect ranking and multi-arm allocation are established in prioritization (Yadlowsky et al., 2025; Sverdrup et al., 2025). The connection is exact. If $S_{s,r}$ selects the top-r fraction of score s, with uniform randomization at ties, then

$$\mathcal{R}_s(r) = r\,\mathrm{TOC}(r; s), \qquad \int_0^1 \mathcal{R}_s(r)\,dr = \mathrm{Qini}(s). \tag{1}$$

We study their optimal envelope over F-measurable policies, the information needed to preserve it, and estimation and acquisition costs. Fully paired outcomes enable direct evaluation. Appendix A relates the derivations to these precedents; our focus is compute value and its realization, not a new routing architecture.

3 The Value of Allocating Compute

We begin with the first question: given a fixed pair of inference actions and information available at decision time, how much value can selective allocation create?

3.1 A matched-budget decision problem

Let U_A and U_B denote the utilities of two inference actions on the same input. Utility may be correctness, or a task-specific reward that prices mistakes and calls. Define the conditional incremental

advantage $g(F)$ and its mean μ by

$$\Delta = U_B - U_A, \qquad g(F) = \mathbb{E}[\Delta \mid F], \qquad \mu = \mathbb{E}\Delta. \tag{2}$$

Information F must be observable before choosing B: it may contain the input, metadata, or an existing cheap answer. The expensive output is an outcome of this choice, not a routing feature.

A randomized router $S(F) \in [0,1]$ selects B with probability $S(F)$. Its expected utility is $\mathbb{E}U_A + \mathbb{E}[\Delta S(F)]$. We first study a common-prefix protocol: every policy has acquired F, and each upgrade has the same additional cost. A budget r then imposes $\mathbb{E}S = r$. A feature-free allocation upgrades independently of F and gains $r\mu$ over A. We define the allocation value

$$\mathcal{R}_F(r) = \sup_{0 \le S \le 1:\, \mathbb{E}S = r} \mathbb{E}[\Delta S(F)] - r\mu. \tag{3}$$

For these two actions and this exact budget, random allocation is the feature-free reference. Equation 3 measures how much the representation contributes beyond simply buying more calls.

Since $\mathbb{E}S = r$ and $\mathbb{E}g = \mu$, the residual of a feasible router is $\mathrm{Cov}(g(F), S(F))$: allocation creates value by aligning calls with heterogeneous returns.

3.2 A CURVE FOR EVERY BUDGET

Theorem 3.1 (Budget-resolved allocation value)**.** *Assume $\mathbb{E}|\Delta| < \infty$, and let Q_g be a quantile function of $g(F)$. An optimal policy upgrades the top-r fraction of $g(F)$, randomizing at threshold ties, and*

$$\mathcal{R}_F(r) = \int_{1-r}^{1} Q_g(u)\, du - r\mu. \tag{4}$$

The curve is nonnegative and concave, with $\mathcal{R}_F(0) = \mathcal{R}_F(1) = 0$. At continuity points,

$$\mathcal{R}'_F(r) = Q_g(1-r) - \mu. \tag{5}$$

For an independent copy F', its area satisfies

$$\int_0^1 \mathcal{R}_F(r)\, dr = \frac{1}{4}\mathbb{E}|g(F) - g(F')|. \tag{6}$$

Moving selection probability toward higher advantage preserves the budget and increases utility. Equation 5 gives marginal advantage relative to random upgrading; Appendix A proves the claims. These properties concern the optimal curve; fitted-score curves need not be nonnegative or concave.

A constant shift in advantage changes average compute value but not $\mathcal{R}_F$. Predictable variation can create allocation value even when all advantages share a sign.

Targeting and spending have different stopping points. At price λ per upgrade, net value is $\mathbb{E}U_A + r\mu + \mathcal{R}_F(r) - \lambda r$; an interior spending optimum satisfies $Q_g(1-r) = \lambda$, whereas an interior maximum of targeting value satisfies $Q_g(1-r) = \mu$.

3.3 WHICH INFORMATION DOES A ROUTER NEED?

Let $Z = \phi(F)$ be a compressed routing representation. Since every Z-based policy is also an F-based policy, $\mathcal{R}_Z(r) \le \mathcal{R}_F(r)$ for every r. The practical question is when compression preserves the decisions that matter.

Proposition 3.2 (Information requirements)**.** *Without a budget constraint, Z preserves Bayes value if and only if a common optimal action exists almost surely within each Z-cell. At a fixed rate r, it preserves value if and only if a Z-measurable router can realize an optimal upper-r selection of $g(F)$. In particular, a score preserving the ordering of $g(F)$, including distinctions needed to resolve its level sets, preserves the complete curve.*

Unconstrained decisions need the sign of g; a fixed budget needs its selection boundary; all budgets require the ordering of distinct advantages.

Cheap confidence is useful when it preserves this ordering. Under accuracy utility, $g(F) = p_B(F) - p_A(F)$, where p_A, p_B are conditional correctness probabilities. Cheap-error ranking, $1 - p_A(F)$, uses only one term. A class with $(p_A, p_B) = (0.50, 0.52)$ is harder for A but gains only 0.02, versus 0.15 for $(0.80, 0.95)$. Constant p_B suffices for the rankings to agree; specialization makes their difference essential.

4 From Conditional Value to a Routing Policy

We now consider learning this ordering from finite data and paying for the information it requires.

4.1 Learning the quantity used by the decision

Paired evaluations give targets $\Delta_i = U_{B,i} - U_{A,i}$. Under accuracy utility, these equal $+1$ for useful correction, -1 for harmful replacement, and zero for equal correctness. Squared-loss regression estimates $g(F)$ because its population minimizer is the conditional mean, in the same utility units as the decision.

A batch policy selects the top-r fraction of $\widehat{g}$, randomizing at boundary ties to meet the expected quota. Unconstrained decisions select B when $\widehat{g} > 0$. A streaming policy freezes a threshold and tie probability on a reference sample; its deployment rate depends on the score distribution.

Proposition 4.1 (Advantage-estimation regret)**.** *Let $V(S) = \mathbb{E}U_A + \mathbb{E}[g(F)S(F)]$. For the unconstrained rules $S^* = \mathbf{1}\{g > 0\}$ and $\widehat{S} = \mathbf{1}\{\widehat{g} > 0\}$,*

$$V(S^*) - V(\widehat{S}) = \mathbb{E}[|g|\mathbf{1}\{S^* \neq \widehat{S}\}] \leq \mathbb{E}|g - \widehat{g}|. \tag{7}$$

If S_r^* and $\widehat{S}_r$ maximize g and $\widehat{g}$, respectively, over the same rate-r feasible set, then

$$V(S_r^*) - V(\widehat{S}_r) \leq \mathbb{E}|g - \widehat{g}|. \tag{8}$$

Estimated-policy optimality gives $\mathbb{E}[\widehat{g}(S_r^* - \widehat{S}_r)] \leq 0$, leaving the absolute-error bound. The bound applies to any estimator; preserving the ordering suffices without calibration. We compare how learning targets recover that ordering from available information.

4.2 When routing information has a price

The common-prefix comparison asks how to allocate upgrades after F has been acquired. A system may also have the option to skip that prefix. Suppose the cheap action costs one unit and a direct strong call costs $c_s > 1$. A cheap-first router pays $1 + rc_s$, since it executes both calls on upgraded inputs. A direct random mixture chooses either action and costs $1 + q(c_s - 1)$. Equal expected cost therefore requires

$$q = \frac{rc_s}{c_s - 1}. \tag{9}$$

The direct mixture can afford more strong calls because it does not purchase the cheap response on those inputs.

Proposition 4.2 (The cost of the routing signal)**.** *If $q \leq 1$, the residual of an optimal cheap-first rate-r router against the direct cost-matched mixture is*

$$\mathcal{R}_F^{\text{direct}}(r) = \mathcal{R}_F(r) - \frac{r}{c_s - 1}\mu. \tag{10}$$

When $\mu \geq 0$, this mixture is the best feature-free policy over A, B with cost at most the router's budget.

The first term rewards targeting; the second charges strong calls forgone to obtain the signal. For $\mu > 0$, acquisition pays when the former exceeds the latter. Feasibility requires $r \leq 1 - 1/c_s$.

Before any call, an informed router observes free input features F_0. It can select an action directly or purchase the cheap output, obtaining $F_1 = (F_0, O_A)$ before deciding whether to call B. At equal cost, with $q = rc_s/(c_s - 1) \leq 1$, the respective optimal policies satisfy

$$V_{F_1}^{\text{cascade}}(r) - V_{F_0}^{\text{direct}}(q) = \mathcal{R}_{F_1}(r) - \mathcal{R}_{F_0}(q) - (q - r)\mu. \tag{11}$$

Equation 10 has $\mathcal{R}_{F_0} = 0$. Buying information must compensate for fewer strong calls and for targeting already possible from free features.

For heterogeneous costs, let $m(F) = \mathbb{E}[\Delta R \mid F]$ and $d(F) = \mathbb{E}[\Delta C \mid F]$. Under an expected incremental-cost budget b, a Lagrange multiplier $\lambda \geq 0$ gives the selection rule

$$S_\lambda(F) = \mathbf{1}\{m(F) - \lambda d(F) > 0\}, \qquad \mathbb{E}[d(F)S_\lambda(F)] \leq b, \tag{12}$$

with boundary randomization when needed. For positive $d(F)$, this thresholds benefit per marginal cost; if U already prices calls, it thresholds net advantage.

5 EXPERIMENTAL DESIGN

We test whether incremental returns predict useful fixed-pair allocation on SWE-Gym and RouterBench, then compare signal-acquisition costs, inexpensive alternatives, and sequential continuation. Complementary verification tasks examine other actions. Expanded-action, richer-representation, and within-task controls were designed after inspecting primary results and reuse their holdouts; we treat them as exploratory.

Actions, data, and splits. SWE-Gym (Pan et al., 2025) supplies 5,101 repair trajectories from 2,251 problem instances across 11 repositories. Given an existing trajectory, A and B predict "resolved" or "unresolved"; utility is agreement with repository-test outcomes. Four Qwen2.5-Coder configurations are evaluated: 1.5B→7B, 3B→14B, 7B→32B, and 14B→32B. The instance-disjoint split contains 2,552 fit, 1,292 calibration, and 1,257 test trajectories, with 553 test instances. Predictors train on fit only; routing methods share paired outcomes. The main RouterBench (Hu et al., 2024) analysis uses 27,241 deduplicated questions with binary correctness: 13,644 fit, 6,825 calibration, and 6,772 test. Its 11 models yield all 55 unordered pairs. A supplementary graded-utility analysis uses 36,497 questions, with 9,126 held out (Appendix B).

Routing scores. We compare five learned scores: cheap-error prediction $1-\widehat{p}_A(F)$, two-head prediction $\widehat{p}_B(F) - \widehat{p}_A(F)$, and direct advantage regression $\widehat{g}(F)$ using ridge regression or gradient-boosted trees (GBDT). Positive-gain GBDT predicts $\Pr(\Delta > 0 \mid F)$ rather than net mean improvement. Simple references are verbal confidence and a one-bit rule grouping cheap "resolved" predictions, with groups ranked by fit mean advantage. Learned SWE scores share features, including the cheap prediction, confidence, trajectory length, and patch/test/error counts; the four GBDT constructions share a base learner. Repository identity is excluded. RouterBench uses question-text TF–IDF and linear predictors fitted on fit; its category reference estimates mean advantage by task category from fit outcomes.

Budgets and evaluation. All SWE policies share action A and allocate the same expected fraction r of calls to B, randomizing uniformly at score ties. Their empirical allocation value is

$$\widehat{\mathcal{R}}(\widehat{g}, r) = \frac{1}{n}\sum_{i=1}^{n} \Delta_i (S_i - r), \qquad \frac{1}{n}\sum_i S_i = r. \tag{13}$$

We report $R_{30} = \widehat{\mathcal{R}}(\widehat{g}, 0.3)$ and the integral over budgets, AURC. Tables multiply both by 100; R_{30} is then an accuracy difference in percentage points. These are gains in trajectory judgment accuracy. Uncertainty uses 5,000 paired bootstrap draws, resampling SWE problem instances or RouterBench questions. Appendix B details features, fitting, ties, and uncertainty.

6 CONDITIONAL VALUE IN PRACTICE

6.1 PAIRED OUTCOMES SUPPORT USEFUL BUDGET ALLOCATION

All five learned scores yield positive allocation value in every configuration: 3.92–8.35 points at 30% upgrading (Table 1). On 7B→32B, ridge advantage raises accuracy from 69.93% under random allocation to 78.28%.

Aligning utility with the verification decision. Unresolved trajectories constitute 94.27% of the test set. A constant unresolved decision attains this accuracy, with zero resolved recall and 50% balanced accuracy (BA). For class-balanced utility, fit-weighted advantage routing yields BA of 76.28, 77.59, 78.90, and 77.73%. Appendix E reports inexpensive final decisions, both recalls, and ordinary accuracy alongside these fixed-pair gains.

Label-probability uncertainty yields positive R_{30} in every pair, while verbal confidence yields negative value (Table 1). The one-bit rule recovers 2.13–6.48 points; richer features refine priorities within these groups. All numerical comparisons are in Table 1.

On 7B→32B, ridge advantage yields 8.35 points with a 95% interval of $[6.96, 9.72]$, and cheap-error prediction yields 8.19 points with $[6.59, 9.28]$. Ridge advantage exceeds error prediction by 0.32,

Table 1: **Allocation value at matched upgrade budgets.** SWE-Gym test set. $R_{30} = R(0.3)$ and AURC is the area under the budget curve; both are multiplied by 100. Learned scores share features; GBDT-based scores share hyperparameters. Label probability uses an additional two-label likelihood probe; budgets here count upgrades. Bold marks the largest displayed point estimate, including rounded ties.

	1.5B→7B		3B→14B		7B→32B		14B→32B	
Routing score	R_{30}	AURC	R_{30}	AURC	R_{30}	AURC	R_{30}	AURC
Verbal confidence	-1.65	-1.34	-1.31	-0.70	-4.06	-3.68	-3.04	-2.72
Label probability	0.82	0.28	2.08	1.52	0.56	0.24	4.50	2.66
One-bit	2.13	1.37	2.42	1.64	6.48	4.16	6.23	3.81
Error (GBDT)	3.92	2.18	5.26	2.99	8.19	4.58	**7.53**	4.29
Two-head (GBDT)	3.92	2.64	5.50	3.46	7.65	4.30	7.05	3.97
Positive-gain (GBDT)	4.08	2.48	**5.82**	**3.48**	7.96	**4.92**	7.30	4.49
Advantage (Ridge)	**4.24**	**2.68**	5.26	3.39	**8.35**	4.90	**7.53**	**4.52**
Advantage (GBDT)	3.92	2.54	5.26	3.46	7.88	4.41	7.05	4.02



Figure 2: **Allocation value across upgrade budgets.** Label-probability uncertainty, one-bit, error GBDT, and advantage Ridge use the same held-out outcomes and upgrade fractions. The likelihood probe adds signal-acquisition cost; the two trained predictors share the full feature vector. Curves connect 41 fractions. The zero line is random allocation; allocation choices disappear at both endpoints.

0.00, 0.16, and 0.00 points across configurations; all four paired intervals include zero. Several estimators recover the useful ordering, with no established universal advantage of direct regression.

Available information changes the target comparison. We replace structural features with full-trajectory word/character TF–IDF or a frozen CodeBERT encoder (Feng et al., 2020). Each is evaluated before (F_0) and after (F_1) observing the cheap judgment, using the same ridge learner and fit-only preprocessing (Appendix D).

All 16 advantage estimates remain positive (Table 2). Before the cheap judgment, advantage exceeds matched error prediction by 2.07 and 3.74 points for text and 1.59 and 3.19 for CodeBERT on the first two pairs. These differences survive Holm correction across all 16 comparisons; after adding the cheap output, none does. Thus the distinction between failure and incremental return depends on the information state (Appendix D).

From batch budgets to fixed decision thresholds. Freezing text-F_1 thresholds on calibration at a 30% target yields test call fractions of 30.95, 31.11, 32.86, and 32.78%. Gains over random allocation at those fractions remain 4.43, 5.57, 8.21, and 7.20 points. Unlike exact-quota batch evaluation, this policy never observes the test score distribution (Appendix D).

6.2 Learned allocation transfers across repositories

For repository transfer, direct GBDT advantage trains on fit instances from ten repositories and evaluates every trajectory in the eleventh. Repository identity is excluded.

Table 2: **Allocation value persists with richer input representations.** SWE-Gym R_{30} in percentage points, using advantage Ridge in all cells. F_0 contains trajectory information; F_1 also contains the cheap judgment. Every displayed gain has a positive paired 95% interval. Upgrade fractions are matched; feature-extraction cost is excluded.

Information	1.5B→7B	3B→14B	7B→32B	14B→32B
Text, F_0	2.73	3.03	5.65	5.30
Text, F_1	4.48	5.66	8.51	7.37
CodeBERT, F_0	2.49	2.64	5.10	4.98
CodeBERT, F_1	4.24	5.51	8.27	7.68

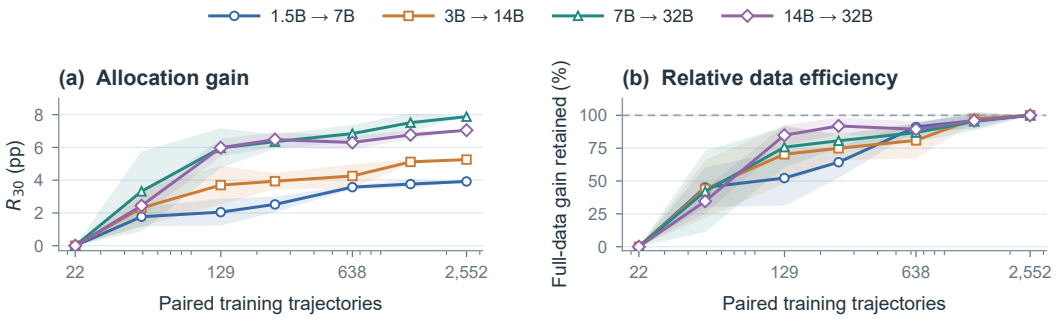

Figure 3: **Data requirements for learning conditional incremental advantage.** Direct GBDT advantage with nested, instance-grouped training subsets. Left: allocation gain at a 30% upgrade budget. Right: gain relative to training on all 2,552 fit trajectories. Lines show means over five subset seeds and bands show one standard deviation across these seeds. The horizontal axis is logarithmic and uses actual mean trajectory counts.

All 44 configuration–repository combinations have positive R_{30} and AURC. Mean gains range from 4.10 to 6.52 points at the 30% budget, and the smallest repository-level AURC remains positive in each configuration (Table 8).

Repeating the full fitting and calibration procedure with the four richer representations gives positive R_{30} and AURC in all 176 configuration–representation–repository conditions. Text-F_1 mean gains are 4.88, 5.84, 7.91, and 7.41 points. Thus repository transfer persists with both lexical and pretrained information. These are descriptive fold results, not independent significance tests (Appendix D).

6.3 A MODEST PAIRED SAMPLE CAPTURES MUCH OF THE VALUE

Across seven nested fit-instance fractions and five seeds, the test set stays fixed (Figure 3). On 14B→32B, about 129 paired trajectories recover 5.98 points, or 84.9% of full-data gain; about 249 yield 6.48 points. About 1,293 trajectories retain 95.4–97.3% across all four pairs. Cheap-error prediction is also competitive on small subsets for both 32B configurations. The return to acquiring paired supervision depends on the pair (Appendix C).

6.4 ALLOCATION IMPROVES ACTUAL ANSWER CORRECTNESS

Actual answer correctness. The main RouterBench subset retains MMLU, HellaSwag, ARC-Challenge, Winogrande, and MBPP tasks with binary correctness for every model. Using the question-text protocol in Section 5, three prespecified representative pairs improve answer correctness by 3.32–6.02 points (Table 3). Across all 55 model pairs, text advantage has positive R_{30} and AURC point estimates, with medians 4.21 and 3.17.

Controlling task-mixture effects. Fixing 30% upgrading within every category, including each MMLU subject, prevents moving calls between tasks. Gains remain 1.37–2.64 points, with positive paired intervals (Table 3) and Holm-adjusted $p < 0.001$ across the three pairs. Each stratified bootstrap draw re-ranks within categories while preserving their sizes. For GPT-3.5→GPT-4, text

Table 3: **Allocation improves actual answer correctness within and across tasks.** RouterBench, 6,772 held-out questions. Accuracy columns use a global 30% upgrade budget. The final column fixes 30% separately within each task and reports gain over within-task random allocation, with stratified paired 95% intervals. Gains are percentage points.

Model pair	Random	Category	Text	Gain	Within-task gain [CI]
Mistral→Mixtral	35.93	41.77	41.95	6.02	2.64 [2.00, 3.21]
Mistral→GPT-3.5	38.41	41.48	41.72	3.32	1.37 [0.72, 2.16]
GPT-3.5→GPT-4	69.60	71.46	73.05	3.45	1.47 [0.83, 2.01]

also exceeds the unrestricted category router by 1.59 $[1.05, 2.10]$ points. Appendix G reports task-family results and transfer.

6.5 When Does Allocation Value Justify Additional Compute?

Paying for the signal. Using matched text features and ridge learners, we compare a cheap-first SWE-Gym cascade with input-only direct routing. At $c_s = 10$, a 30% cascade and 33.3% direct router both cost four cheap-call units; cascade gains are 1.67, 2.55, 2.47, and 1.75 points, with positive paired 95% intervals. At ratio two, both 32B configurations instead have negative point estimates with intervals crossing zero. Appendix F reports all prices from 2 to 20; feature extraction and hardware latency are excluded.

Available alternatives. A fit-trained, calibration-selected text classifier reaches 79.35% BA without a judge call. Relative to this classifier, residual-routing and output-fusion controls do not establish a BA improvement: all nonzero paired difference intervals include zero (Table 14). Gains within the original model pair therefore do not establish the value of additional calls once this inexpensive decision is available. Appendix E also evaluates richer text representations and joint deferral.

Allocating answer samples through continuation. Using independently graded 1/4/8-sample caches, we fit stopping and continuation values from question text and the observed prefix. The continuation target includes the fitted value of subsequent sampling and stopping decisions. At zero token penalty, the policy reaches 87.29% accuracy on MATH500 and 96.02% on GSM8K, using 62.4% and 40.5% fewer generated tokens than fixed eight, whose accuracies are 86.44% and 96.02%. On these 118 and 327 test questions, zero-penalty myopic stopping attains the same accuracies with higher token use. Appendix G.4 reports fixed budgets, myopic stopping, and all five penalties.

Evidence retrieval and tool selection. Appendix H examines retrieval and symbolic/model verification, specifying the available signals, action pairs, and utilities for each task.

7 Implications and Conclusion

Adaptive inference depends on predictable variation in incremental return, not only on average improvement. We characterize this return through $g(F)$ and its budget-resolved allocation value; learned scores recover useful ordering across SWE-Gym repositories and actual RouterBench answers.

Deployment value also depends on signal-acquisition costs and inexpensive alternatives. Our cost, expanded-action, and continuation experiments show why predicted incremental return must be evaluated against the full information and action set available at deployment.

These experiments do not establish a universally superior estimator or a population noninferiority guarantee for continuation. They support conditional incremental return as an allocation target whose practical benefit must be checked under the deployment utility, budget, and available alternatives.

Reproducibility Statement

The accompanying materials contain paired predictions, instance-level splits, input hashes, complete comparisons, and figure-generation scripts. Appendix B specifies the main protocol, repository transfer, and training-subset evaluations. Appendix H documents the fixed policies, utilities, and sources for complementary verification tasks.

Ethics Statement

The experiments use public benchmarks and cached model outputs. Their utilities make error penalties and computation costs explicit. Application-specific deployment should choose these quantities to reflect the consequences of the final decisions and the populations receiving additional verification.

AI Use Statement

Generative AI tools were used to polish the writing and to assist with software code. OpenAI Codex also assisted with manuscript editing, bibliographic corrections, and preparation of the supplementary code package for this submission. The accompanying code includes executable checks and records of the validations performed. The authors take responsibility for the final text, code, analyses, and claims.

References

Pranjal Aggarwal, Aman Madaan, Ankit Anand, Srividya Pranavi Potharaju, Swaroop Mishra, Pei Zhou, Aditya Gupta, Dheeraj Rajagopal, Karthik Kappaganthu, Yiming Yang, Shyam Upadhyay, Manaal Faruqui, and Mausam. AutoMix: Automatically mixing language models. In *Advances in Neural Information Processing Systems*, volume 37, 2024. doi: 10.52202/079017-4164.

Bradley Brown, Jordan Juravsky, Ryan Ehrlich, Ronald Clark, Quoc V. Le, Christopher Ré, and Azalia Mirhoseini. Large language monkeys: Scaling inference compute with repeated sampling. *arXiv preprint arXiv:2407.21787*, 2024.

Bei Chen, Fengji Zhang, Anh Nguyen, Daoguang Zan, Zeqi Lin, Jian-Guang Lou, and Weizhu Chen. CodeT: Code generation with generated tests. In *International Conference on Learning Representations (ICLR)*, 2023.

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://openreview.net/forum?id=cSimKw5p6R`.

C. K. Chow. On optimum recognition error and reject tradeoff. *IEEE Transactions on Information Theory*, 16(1):41–46, 1970.

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.

Jasper Dekoninck, Maximilian Baader, and Martin Vechev. A unified approach to routing and cascading for LLMs. In *Proceedings of the 42nd International Conference on Machine Learning (ICML)*, 2025.

Dujian Ding, Ankur Mallick, Chi Wang, Robert Sim, Subhabrata Mukherjee, Victor Rühle, Laks V.S. Lakshmanan, and Ahmed Hassan Awadallah. Hybrid LLM: Cost-efficient and quality-aware query routing. In *International Conference on Learning Representations (ICLR)*, 2024.

Ran El-Yaniv and Yair Wiener. On the foundations of noise-free selective classification. *Journal of Machine Learning Research*, 11:1605–1641, 2010.

Zhangyin Feng, Daya Guo, Duyu Tang, Nan Duan, Xiaocheng Feng, Ming Gong, Linjun Shou, Bing Qin, Ting Liu, Daxin Jiang, and Ming Zhou. CodeBERT: A pre-trained model for programming and natural languages. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pp. 1536–1547, 2020. doi: 10.18653/v1/2020.findings-emnlp.139. URL https://aclanthology.org/2020.findings-emnlp.139/.

Yonatan Geifman and Ran El-Yaniv. Selective classification for deep neural networks. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2017.

Yonatan Geifman and Ran El-Yaniv. SelectiveNet: A deep neural network with an integrated reject option. In *International Conference on Machine Learning (ICML)*, 2019.

Daniel Golovin and Andreas Krause. Adaptive submodularity: Theory and applications in active learning and stochastic optimization. *Journal of Artificial Intelligence Research*, 42:427–486, 2011.

Neha Gupta, Harikrishna Narasimhan, Wittawat Jitkrittum, Ankit Singh Rawat, Aditya Krishna Menon, and Sanjiv Kumar. Language model cascades: Token-level uncertainty and beyond. In *International Conference on Learning Representations (ICLR)*, 2024. URL https://proceedings.iclr.cc/paper_files/paper/2024/hash/11f5520daf9132775e8604e89f53925a-Abstract-Conference.html.

Ronald A. Howard. Information value theory. *IEEE Transactions on Systems Science and Cybernetics*, 2(1):22–26, 1966.

Qitian Jason Hu, Jacob Bieker, Xiuyu Li, Nan Jiang, Benjamin Keigwin, Gaurav Ranganath, Kurt Keutzer, and Shriyash Kaustubh Upadhyay. RouterBench: A benchmark for multi-LLM routing system. *arXiv preprint arXiv:2403.12031*, 2024.

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 (NeurIPS)*, 2023.

Andreas Krause and Carlos Guestrin. Optimal value of information in graphical models. *Journal of Artificial Intelligence Research*, 35:557–591, 2009.

Hunter Lightman, Vineet Kosaraju, Yura Burda, Harri Edwards, Bowen Baker, Teddy Lee, Jan Leike, John Schulman, Ilya Sutskever, and Karl Cobbe. Let's verify step by step. In *International Conference on Learning Representations (ICLR)*, 2024.

David Madras, Toniann Pitassi, and Richard Zemel. Predict responsibly: Improving fairness and accuracy by learning to defer. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2018.

Saumya Malik, Valentina Pyatkin, Sander Land, Jacob Morrison, Noah A. Smith, Hannaneh Hajishirzi, and Nathan Lambert. RewardBench 2: Advancing reward model evaluation. In *International Conference on Learning Representations (ICLR)*, 2026. URL https://openreview.net/forum?id=fb0G86Dewb.

Anqi Mao, Christopher Mohri, Mehryar Mohri, and Yutao Zhong. Two-stage learning to defer with multiple experts. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2023.

Hussein Mozannar and David Sontag. Consistent estimators for learning to defer to an expert. In *Proceedings of the 37th International Conference on Machine Learning (ICML)*, 2020.

Lunyiu Nie, Zhimin Ding, Erdong Hu, Christopher Jermaine, and Swarat Chaudhuri. Online cascade learning for efficient inference over streams. In *Proceedings of the 41st International Conference on Machine Learning (ICML)*, 2024.

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 (ICLR)*, 2025.

Jiayi Pan, Xingyao Wang, Graham Neubig, Navdeep Jaitly, Heng Ji, Alane Suhr, and Yizhe Zhang. Training software engineering agents and verifiers with SWE-Gym. In *Proceedings of the 42nd International Conference on Machine Learning*, 2025. URL https://arxiv.org/abs/2412.21139.

Dhruv Rohatgi, Abhishek Shetty, Donya Saless, Yuchen Li, Ankur Moitra, Andrej Risteski, and Dylan J. Foster. Taming imperfect process verifiers: A sampling perspective on backtracking. In *International Conference on Learning Representations (ICLR)*, 2026.

Stuart Russell and Eric Wefald. Principles of metareasoning. *Artificial Intelligence*, 49(1–3):361–395, 1991.

Marija Šakota, Maxime Peyrard, and Robert West. Fly-swat or cannon? cost-effective language model choice via meta-modeling. In *ACM International Conference on Web Search and Data Mining (WSDM)*, 2024.

Charlie Snell, Jaehoon Lee, Kelvin Xu, and Aviral Kumar. Scaling LLM test-time compute optimally can be more effective than scaling parameters for reasoning. In *International Conference on Learning Representations (ICLR)*, 2025.

Erik Sverdrup, Han Wu, Susan Athey, and Stefan Wager. Qini curves for multi-armed treatment rules. *Journal of Computational and Graphical Statistics*, 34(3):948–960, 2025. doi: 10.1080/10618600.2024.2418820.

Rajeev Verma and Eric Nalisnick. Calibrated learning to defer with one-vs-all classifiers. In *Proceedings of the 39th International Conference on Machine Learning (ICML)*, 2022.

Yingjia Wan, Haochen Tan, Xiao Zhu, Xinyu Zhou, Zhiwei Li, Qingsong Lv, Changxuan Sun, Jiaqi Zeng, Yi Xu, Jianqiao Lu, Yinhong Liu, and Zhijiang Guo. FaStFact: Faster, stronger long-form factuality evaluations in LLMs. In *Findings of the Association for Computational Linguistics: EMNLP 2025*, pp. 23814–23854, 2025. URL https://aclanthology.org/2025.findings-emnlp.1295/.

Xuezhi Wang, Jason Wei, Dale Schuurmans, Quoc V. Le, Ed H. Chi, Sharan Narang, Aakanksha Chowdhery, and Denny Zhou. Self-consistency improves chain of thought reasoning in language models. In *International Conference on Learning Representations (ICLR)*, 2023.

Ian Waudby-Smith and Aaditya Ramdas. Estimating means of bounded random variables by betting. *Journal of the Royal Statistical Society: Series B*, 86(1):1–27, 2024.

Yangzhen Wu, Zhiqing Sun, Shanda Li, Sean Welleck, and Yiming Yang. Inference scaling laws: An empirical analysis of compute-optimal inference for problem-solving with language models. *arXiv preprint arXiv:2408.00724*, 2024.

Steve Yadlowsky, Scott Fleming, Nigam Shah, Emma Brunskill, and Stefan Wager. Evaluating treatment prioritization rules via rank-weighted average treatment effects. *Journal of the American Statistical Association*, 120(549):38–51, 2025. doi: 10.1080/01621459.2024.2393466.

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 (ICLR)*, 2024.

GUIDE TO THE APPENDIX

The appendix follows the questions developed in the paper. Appendix A collects the mathematical arguments; Appendix B defines the shared evaluation protocol. Appendices C and D examine prediction targets, supervision, and repository transfer. Appendix E evaluates the final decision when inexpensive alternatives are available, and Appendix F accounts for acquiring routing information. Appendix G studies actual answers and continuation. Complementary action types and their task-specific protocols appear in Appendix H; Appendix I maps the results to reproducible artifacts.

A PROOFS AND DECISION-THEORETIC DETAILS

A.1 THE BUDGET CURVE

For any F-measurable S, iterated expectation gives $\mathbb{E}[\Delta S] = \mathbb{E}[gS]$. Fix $r \in (0,1)$, let $t = Q_g(1-r)$, and define

$$S_r^*(F) = \mathbf{1}\{g(F) > t\} + \rho \mathbf{1}\{g(F) = t\},$$

where $\rho \in [0,1]$ is chosen to make $\mathbb{E}S_r^* = r$. When the threshold has no mass, the second term is immaterial. For any other feasible S,

$$\mathbb{E}[g(S_r^* - S)] = \mathbb{E}[(g-t)(S_r^* - S)] \geq 0.$$

The equality uses equal selection rates. The inequality holds pointwise: above t, $S_r^* = 1$; below t, $S_r^* = 0$; and at t the product is zero. This proves optimality and the upper-quantile integral in Equation 4.

Since $Q_g(1-r)$ is nonincreasing in r, the upper-quantile integral is concave. Subtracting $r\mu$ preserves concavity, and its derivative at continuity points is $Q_g(1-r) - \mu$. The only feasible endpoint allocations are $S = 0$ and $S = 1$, giving zero residual at both ends. A constant randomized policy $S = r$ is feasible and has zero residual, establishing nonnegativity.

For the area identity, integrability of g permits Fubini's theorem:

$$\begin{aligned}\int_0^1 \mathcal{R}_F(r)\, dr &= \int_0^1 \int_{1-r}^1 Q_g(u)\, du\, dr - \frac{1}{2}\mu \\ &= \int_0^1 \left(u - \frac{1}{2}\right) Q_g(u)\, du.\end{aligned}$$

For independent uniform variables U, V on $[0,1]$, $Q_g(U)$ and $Q_g(V)$ are independent copies of g. Monotonicity of Q_g gives

$$\begin{aligned}\mathbb{E}|g - g'| &= 2\int_0^1 \int_0^u \big(Q_g(u) - Q_g(v)\big)\, dv\, du \\ &= 2\int_0^1 (2u-1) Q_g(u)\, du = 4\int_0^1 \mathcal{R}_F(r)\, dr.\end{aligned}$$

Thus the area is one quarter of the Gini mean difference of conditional advantages. In particular, it is zero exactly when g is almost surely constant. This identity describes the optimal representation-level curve; a learned score's finite-sample curve need not be concave.

A.2 REPRESENTATIONS AND INFORMATION REQUIREMENTS

For a compressed representation $Z = \phi(F)$, its conditional advantage is $g_Z(Z) = \mathbb{E}[g(F) \mid Z]$. Any Z-measurable router remains feasible with F, which proves $\mathcal{R}_Z(r) \leq \mathcal{R}_F(r)$. No estimation assumptions are needed for this information ordering.

Without a budget, the gain over always-A is $\mathbb{E}\max(0, g)$. After compression it is $\mathbb{E}\max(0, \mathbb{E}[g \mid Z])$. Equality in the conditional convexity inequality holds precisely when, within almost every Z-cell, there is no positive probability on both strictly positive and strictly negative advantages. Equivalently, a common maximizing action is available throughout the cell, with arbitrary choices at zero. This proves the first part of Proposition 3.2 and makes its treatment of ties explicit.

At a fixed rate, the preceding threshold proof shows that an optimal policy must select the regions strictly above the threshold, omit those strictly below, and use any remaining mass on the threshold. A compressed representation preserves value if and only if it can implement such a policy. If a Z-measurable score preserves the ordering of distinct g values, these selections can be recovered at every r. Equal-advantage regions can be selected in any order, or randomized, because they contribute equal marginal value.

The full-curve statement concerns the score as an information source. If a proposed compression merges different advantages into one indistinguishable value, it can preserve a particular boundary while losing other boundaries. Preserving only the sign is insufficient when a positive budget does not cover every positive-advantage input: the policy must also distinguish their return magnitudes.

A.3 Regret of learned policies

Let $S^* = \mathbf{1}\{g > 0\}$ and $\widehat{S} = \mathbf{1}\{\widehat{g} > 0\}$. Whenever the decisions agree, the conditional regret is zero. When they disagree and $g \neq 0$, it equals $|g|$ and satisfies $|g| \leq |g - \widehat{g}|$. When $g = 0$, both quantities in the regret identity are zero. Taking expectations proves Equation 7, including zero-valued score ties.

For the constrained problem, both policies are feasible for the same rate. Add and subtract the estimated advantage:

$$\begin{aligned} V(S_r^*) - V(\widehat{S}_r) &= \mathbb{E}[(g - \widehat{g})(S_r^* - \widehat{S}_r)] + \mathbb{E}[\widehat{g}(S_r^* - \widehat{S}_r)] \\ &\leq \mathbb{E}[|g - \widehat{g}|\,|S_r^* - \widehat{S}_r|] \leq \mathbb{E}|g - \widehat{g}|. \end{aligned}$$

The second expectation is nonpositive because $\widehat{S}_r$ maximizes the estimated objective. The last inequality uses $S_r^*, \widehat{S}_r \in [0, 1]$. These statements condition on the learned function; one can additionally average over the fit sample. For a finite batch, the same algebra holds with empirical expectations and a common exact quota.

A.4 Direct calls and variable costs

A cheap-first allocation has utility $\mathbb{E}U_A + \mathbb{E}[\Delta S]$ and cost $1 + rc_s$. The direct mixture has utility $\mathbb{E}U_A + q\mu$ and cost $1 + q(c_s - 1)$. Equating costs gives Equation 9, and subtracting utilities gives

$$\mathbb{E}[\Delta S] - q\mu = \big(\mathbb{E}[\Delta S] - r\mu\big) - \frac{r}{c_s - 1}\mu.$$

The correction is independent of the selection rule at a fixed rate. Maximizing over S proves Proposition 4.2. If $\mu \geq 0$, expected utility among feature-free mixtures is nondecreasing in q, so the cost-matched mixture is also optimal under the corresponding upper budget. If cheap and strong costs are c_A, c_B instead of $1, c_s$, the correction becomes $rc_A\mu/(c_B - c_A)$, when the corresponding mixture is feasible.

For variable incremental cost, maximizing $\mathbb{E}[mS]$ subject to $\mathbb{E}[dS] \leq b$ gives the Lagrangian $\mathbb{E}[(m - \lambda d)S] + \lambda b$. Pointwise maximization over $S \in [0, 1]$ selects positive $m - \lambda d$, omits negative values, and allows randomization at zero. A multiplier satisfying feasibility and complementary slackness gives an optimum. For $d > 0$, the rule is equivalent to thresholding m/d; zero-cost actions are handled by the sign of m. This is an expected-budget statement, appropriate for randomized allocation or a population budget.

A.5 Relation to targeting measures

For $r > 0$, $\mathrm{TOC}(r; s) = \mathbb{E}[\Delta S_{s,r}]/r - \mu$ under uniform selection within ties. Multiplying by r gives Equation 1; integrating gives the Qini functional in Yadlowsky et al. (2025). Our empirical score curve is this same object. The population curve $\mathcal{R}_F$ maximizes it over all policies measurable from F. It is concave; an arbitrary fitted score's empirical curve need not be concave or nonnegative.

The optimal relative-correctness rule and conditions under which confidence recovers it are also central to Jitkrittum et al. (2023). Theorem 3.1 uses the standard capacity-constrained upper-tail rule, whose quantile representation yields the area identity. Proposition 3.2 organizes decision sufficiency by the budget constraint. Proposition 4.1 is a plug-in policy bound specialized to paired

compute utility. The compute accounting in Proposition 4.2 and Equation 11 compares actions with different information-acquisition requirements. These relationships position the contribution as a characterization and synthesis for compute decisions.

A.6 Acquiring information before routing

To prove Equation 11, write cascade value as $\mathbb{E}U_A + r\mu + \mathcal{R}_{F_1}(r)$ and direct-router value as $\mathbb{E}U_A + q\mu + \mathcal{R}_{F_0}(q)$. Subtraction gives the identity. It concerns the respective population optima; the empirical comparison evaluates two fitted policies. Here, free features are available before any model call; their feature-extraction overhead is accounted for separately.

A.7 Value remaining after inexpensive decisions

At an acquired information state F, let $\mathcal{C}$ be a finite set of available terminal decisions, including A and lightweight corrections. Write $m_c(F) = \mathbb{E}[U_c \mid F]$, $v_{\mathcal{C}}(F) = \max_{c\in\mathcal{C}} m_c(F)$, and $d(F) > 0$ for the additional cost of B. The cheap actions share the already acquired prefix; any unequal terminal costs are included in their utilities. The relevant remaining advantage is

$$h_{\mathcal{C}}(F) = m_B(F) - v_{\mathcal{C}}(F) = g(F) - \big(v_{\mathcal{C}}(F) - m_A(F)\big). \tag{14}$$

Thus an opportunity to improve the cheap decision reduces the return assigned to the expensive action, even if the original pair has a strongly positive targeting curve.

Proposition A.1 (Remaining value of a paid action)**.** *For a nonnegative price λ per cost unit, the optimal additional value over the F-informed cheap envelope is*

$$H_{\mathcal{C}}(\lambda) = \mathbb{E}[(h_{\mathcal{C}}(F) - \lambda d(F))_+]. \tag{15}$$

An optimal policy calls B exactly where $h_{\mathcal{C}} > \lambda d$, with arbitrary decisions at equality. If $\mathcal{C} \subseteq \mathcal{C}'$, then $H_{\mathcal{C}'}(\lambda) \le H_{\mathcal{C}}(\lambda)$.

This is a consequence of conditional utility maximization, applied to the choice of comparator. Its operational implication is to train against the best available cheap decision and allow zero calls, rather than force an exact upgrade quota. In estimation, the cheap policy must be chosen without test labels, and its predictions used in residual training should be out of fold. The population envelope optimizes conditional expected utility. Appendix E estimates a feasible finite-classifier comparison with calibration-selected policies. The following proof also distinguishes returning a strong label from using its output as evidence.

Proof of Proposition A.1. After acquiring F, the best cheap decision has conditional value $v_{\mathcal{C}}(F)$. Calling B has conditional net value $m_B(F) - \lambda d(F)$. Pointwise maximization gives

$$\max\{v_{\mathcal{C}}, m_B - \lambda d\} - v_{\mathcal{C}} = (m_B - v_{\mathcal{C}} - \lambda d)_+.$$

Taking expectations proves the value expression and its maximizing decision. Enlarging $\mathcal{C}$ increases $v_{\mathcal{C}}$ pointwise and decreases the positive part. It holds for the stated common prefix and cost convention; moving back before that prefix requires charging for its acquisition.

An output can be evidence as well as a final decision. For binary accuracy, let $p = \Pr(Y = 1 \mid F)$ and $p' = \Pr(Y = 1 \mid F, O_B)$. If both terminal labels are admissible, the Bayes value of observing O_B before choosing a label is

$$\mathbb{E}[|p' - \tfrac{1}{2}| \mid F] - |p - \tfrac{1}{2}| \ge 0.$$

The inequality follows from $\mathbb{E}[p' \mid F] = p$ and convexity. The value is zero exactly when, conditionally on F, the posterior stays on one side of $1/2$, allowing equality. At $p = 1/2$, zero value requires $p' = 1/2$ almost surely. Thus an informative response need not change the optimal final decision. The fusion control in Appendix E evaluates a fitted post-call classifier, providing a finite-model counterpart to this Bayes calculation. Weighted classification replaces $1/2$ by its utility-specific threshold.

A.8 THE TWO-GROUP EXAMPLE

For a high-advantage group of probability p with value g_H and a low-advantage group with value $g_L < g_H$, the optimal curve is

$$\mathcal{R}_F(r) = \begin{cases} r(1-p)(g_H - g_L), & 0 \le r \le p, \\ p(1-r)(g_H - g_L), & p \le r \le 1. \end{cases}$$

The peak $p(1-p)(g_H - g_L)$ is attained at $r = p$. It depends on the difference in returns and group mass, rather than their common average level. The area is half the peak, agreeing with Equation 6, since two independent group draws differ with probability $2p(1-p)$.

This example connects the theory to an implementable class rule. If a cheap prediction identifies the two groups, selecting the high-return class uses exactly the information required at rate p. At a smaller budget, the rule randomizes within that class. Additional predictive features can improve the smaller-budget decision when returns also vary within the class. The empirical SWE slices illustrate this allocation logic; they are descriptive group averages, and the held-out score evaluations establish the performance of the actual policies.

B SHARED EVALUATION PROTOCOL

Throughout the SWE experiments, A is the current judge and B is the additional judge. The main pairs are 1.5B→7B, 3B→14B, 7B→32B, and 14B→32B. Utility is agreement with the trajectory's resolved status; answer-level correctness is evaluated separately in Appendix G. We write F_0 for input information and F_1 for input information together with the cheap response. These information sets determine what a policy can observe before calling B.

B.1 COMPACT FEATURES AND ESTIMATORS

All five learned main-table scores use a common 13-dimensional vector: the cheap prediction, verbal confidence, distance of confidence from $1/2$, normalized trajectory length, an invalid-output indicator, and the log-transformed counts of messages, tool messages, trajectory characters, patch characters, changed files, official-test mentions, error mentions, and success claims. Missing confidence is replaced by 0.5 and a missing prediction by -1. Features use the trajectory and action A only. Repository identity and action B's output are excluded.

Error GBDT regresses $1 - U_A$; two-head GBDT fits separate regressors for U_A and U_B and subtracts their predictions; direct advantage GBDT regresses $U_B - U_A$. All use histogram gradient boosting with 120 iterations, at most 15 leaves, minimum leaf size 20, ℓ_2 regularization 1, and no early stopping (seed 20260920). Ridge advantage standardizes the features and uses penalty 1. The fit/calibration/test trajectory counts are 2,552/1,292/1,257. No target outcomes enter feature computation or test-time ranking.

B.2 BUDGETS, TIES, AND UNCERTAINTY

Scores are rounded to 12 decimal places before tie grouping. At a budget r, all groups above the threshold are selected, the threshold group receives uniform selection probability sufficient to reach rn, and lower groups receive zero. Utilities average over this randomization. The area integrates the resulting piecewise-linear curve exactly, rather than approximating it from the 41 points drawn in Figure 2. Each bootstrap draw resamples entire test problem instances, retaining their trajectories, and re-evaluates the budget threshold for the fixed fitted score. The 95% intervals are percentile intervals for individual operating points. They capture test-sample uncertainty conditional on the fitted model. Figure 3 instead shows training-subset variability across five seeds.

All main operating-point intervals use 5,000 paired bootstrap draws. Multiplicity families are specified with their respective comparisons. Frozen-threshold deployment fixes the threshold and its boundary randomization on calibration; test call rates may then differ from the calibration target. In contrast, a batch budget curve re-ranks the observed scores at each quota. These evaluate deployable threshold rules and exact-budget allocation, respectively.

The expanded-action, richer-representation, and within-task controls were designed after inspecting the primary results and reuse the same holdout outcomes. We treat them as exploratory evidence conditional on these caches. Their candidate sets and parameter grids are reported in full, with no test-based inclusion of favorable configurations.

B.3 RouterBench graded utilities

All 11 models and all 55 unordered pairs share the 9,126-question test set. Fit contains 18,264 questions. For each pair, orientation is selected from fit mean utility, and task-category conditional advantages are estimated on fit only, shrinking each category mean toward the overall fit mean with 20 pseudo-observations. Each metric uses the same fractional tie rule. Question-level paired bootstrap resampling keeps the model outcomes together. Holm correction jointly covers R_{10}, R_{30}, R_{50}, and AURC across 55 pairs (220 tests). Pairwise results share test questions and are therefore dependent comparisons. This experiment evaluates transfer to held-out questions in the represented task categories.

C Routing Scores and Paired Supervision

C.1 Complete score comparison

Table 4 reports the full score comparison across the four configurations. Positive-gain GBDT predicts $\Pr(\Delta > 0 \mid F)$. kNN uses standardized features and 25 neighbors. Cost-aware GBDT divides estimated advantage by a fitted positive estimate of the additional call cost. ANLL uses the cheap answer's average negative log likelihood. Label-probe uncertainty uses the distance of the normalized two-label probability from $1/2$. SCK uses the first K sampled cheap judgments: its uncertainty score is one minus the largest valid vote share, while its learned scores append validity, vote, disagreement, and confidence features. The extra probe and sampling calls are separate acquisition costs. Accordingly, this table compares equal upgrade fractions, with measured deployment costs archived separately. Each cheap model supplies eight independent samples per trajectory, using temperature 0.7, top-p 0.95, top-k 20, and a cap of 128 generated tokens.

Table 4: Complete score comparison: R_{30}/AURC, both multiplied by 100.

Score	1.5B→7B	3B→14B	7B→32B	14B→32B
Random	0.00 / 0.00	0.00 / 0.00	0.00 / 0.00	0.00 / 0.00
Verbal confidence	-1.65 / -1.34	-1.31 / -0.70	-4.06 / -3.68	-3.04 / -2.72
One-bit	2.13 / 1.37	2.42 / 1.64	6.48 / 4.16	6.23 / 3.81
Error GBDT	3.92 / 2.18	5.26 / 2.99	8.19 / 4.58	7.53 / 4.29
Two-head GBDT	3.92 / 2.64	5.50 / 3.46	7.65 / 4.30	7.05 / 3.97
Advantage Ridge	4.24 / 2.68	5.26 / 3.39	8.35 / 4.90	7.53 / 4.52
Advantage GBDT	3.92 / 2.54	5.26 / 3.46	7.88 / 4.41	7.05 / 4.02
Advantage kNN	4.14 / 2.62	5.48 / 3.49	7.35 / 4.63	6.99 / 4.19
Positive-gain GBDT	4.08 / 2.48	5.82 / 3.48	7.96 / 4.92	7.30 / 4.49
Cost-aware GBDT	3.92 / 2.55	5.26 / 3.46	7.88 / 4.41	7.06 / 4.02
ANLL	0.90 / -0.01	0.01 / 0.32	-1.59 / -0.79	2.35 / 1.74
Label-probe uncertainty	0.82 / 0.28	2.08 / 1.52	0.56 / 0.24	4.50 / 2.66
SC4 uncertainty	1.70 / 1.10	2.04 / 1.25	0.63 / 0.43	0.98 / 0.64
SC4 error	3.92 / 2.12	5.58 / 3.06	8.12 / 4.64	7.53 / 4.25
SC4 advantage	4.00 / 2.59	5.42 / 3.44	7.80 / 4.37	7.45 / 4.23
SC8 uncertainty	1.79 / 1.05	2.53 / 1.43	0.87 / 0.58	1.74 / 1.10
SC8 error	3.92 / 2.23	5.10 / 2.99	8.44 / 4.80	7.61 / 4.28
SC8 advantage	4.16 / 2.66	5.34 / 3.55	7.41 / 4.44	7.05 / 4.02

Table 5 compares learned targets on the same features and test instances, including GBDT and Ridge estimates of advantage. Error prediction is competitive with direct advantage estimation in this compact, cheap-response-informed representation. The richer-information comparison in Appendix D examines how this relationship changes before the cheap response is available.

C.2 Supervision requirements

Training-subset fractions are 0.01, 0.02, 0.05, 0.10, 0.25, 0.50, and 1.00 of fit instances. Each seed permutes instances once and takes nested prefixes, preserving all trajectories within an instance.

Table 5: Difference from error GBDT at $r = 0.3$ (pp). Holm correction covers 24 pair–estimator–metric comparisons (R_{30} and AURC).

Pair	Estimator	Difference	95% CI	Holm p
1.5B→7B	Adv. Ridge	0.32	[-0.15, 0.87]	1.000
1.5B→7B	Adv. GBDT	0.00	[-0.58, 0.57]	1.000
1.5B→7B	Positive-gain	0.16	[-0.40, 0.74]	1.000
3B→14B	Adv. Ridge	0.00	[-0.58, 0.59]	1.000
3B→14B	Adv. GBDT	0.00	[-0.50, 0.56]	1.000
3B→14B	Positive-gain	0.56	[0.15, 1.03]	0.299
7B→32B	Adv. Ridge	0.16	[-0.49, 1.48]	1.000
7B→32B	Adv. GBDT	-0.32	[-0.97, 0.85]	1.000
7B→32B	Positive-gain	-0.24	[-0.98, 0.85]	1.000
14B→32B	Adv. Ridge	0.00	[-0.71, 1.02]	1.000
14B→32B	Adv. GBDT	-0.48	[-1.31, 0.48]	1.000
14B→32B	Positive-gain	-0.23	[-0.97, 0.72]	1.000

Seeds are 20260920 through 20260924. The corresponding mean trajectory counts are 22, 49.2, 128.8, 249, 637.8, 1,292.6, and 2,552. The same fixed learner and test set are used at every size. The 129-trajectory result concerns router supervision from paired model judgments.

Table 6: Same-subset learning comparison: mean R_{30} over five seeds. Approximately 129 and 1,293 paired trajectories. Error labels require A outcomes; advantage additionally uses B outcomes.

Pair	Error (129)	Adv. (129)	Error (1,293)	Adv. (1,293)
1.5B→7B	1.87	2.05	3.78	3.76
3B→14B	2.49	3.70	4.97	5.12
7B→32B	6.33	5.97	7.50	7.51
14B→32B	6.62	5.98	7.19	6.76

D Information Sets and Repository Transfer

D.1 Lexical and pretrained representations

Text features concatenate word unigrams/bigrams (12,000-feature cap) and character 3–5-grams (16,000-feature cap), each with minimum document frequency three and sublinear term frequency. The vocabulary is fitted on the fit partition only. The frozen `microsoft/codebert-base` checkpoint has 124,645,632 parameters. We mean-pool nonspecial tokens in first, middle, and final 510-token windows, then average windows. It reads the trajectory alone. Standardization is fitted on fit; the 768 encoder coordinates are scaled by $1/\sqrt{768}$. F_0 adds the original nine input-structure features; F_1 adds all 13 original features, including the cheap judgment. Target-repository labels and strong outputs never enter decision-time features. Repository names may occur naturally in the text.

Advantage and error heads use ridge regression with penalty 10 and the same design matrix. Every primary comparison uses 5,000 paired instance bootstrap draws, re-ranking both scores at the required budget. For the complete 32 score–representation–pair allocation tests, Holm correction controls one family; paired advantage-minus-error comparisons form a separate family of 16. These intervals condition on fitted models. Encoder inference on the local RTX 5080 took 21.2 seconds for 15,303 windows over 5,101 trajectories (38.9 seconds including loading, tokenization, and saving). This timing measures feature acquisition on the local hardware.

All 16 advantage-allocation intervals exclude zero. Their minimum lower endpoint is 1.67 points. The text F_1 intervals, in pair order, are [3.55, 5.47], [4.45, 6.85], [6.89, 9.77], [5.86, 8.74]. The complete individual intervals and adjusted tests are retained in the result archive.

Figure 4 separates the effects of information and prediction target over the full budget range. Input-only advantage and error scores differ most for the smaller pairs. Once the cheap response is ob-

served, their curves are much closer. Table 7 gives the paired contrasts at 30% and their multiplicity correction.

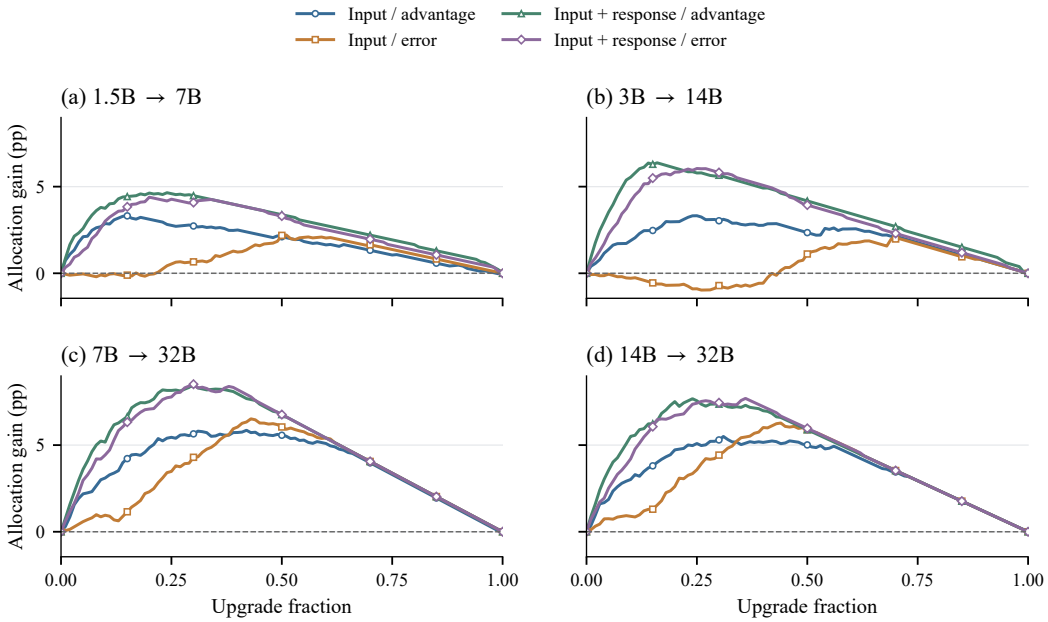

Figure 4: **Information and prediction target across budgets.** All four scores use the same word/character features and ridge learner. Each panel fixes the model pair; colors distinguish the two information sets and two targets. Curves evaluate exact fractional quotas from the frozen test scores. Zero denotes same-rate random allocation; intervals for the 30% contrasts are in Table 7.

Table 7: Matched-target comparison under richer representations. R_{30} and paired differences are percentage points; intervals are pointwise 95%. A star denotes a positive difference significant after Holm correction over all 16 comparisons.

Pair	Representation	Advantage	Error	Difference [CI]
1.5B→7B	Text F_0	2.73	0.66	2.07 * [0.98, 3.21]
1.5B→7B	Text F_1	4.48	4.08	0.40 [0.00, 0.78]
1.5B→7B	Encoder F_0	2.49	0.90	1.59 * [0.56, 2.74]
1.5B→7B	Encoder F_1	4.24	4.40	-0.16 [-0.33, 0.25]
3B→14B	Text F_0	3.03	-0.71	3.74 * [2.51, 5.31]
3B→14B	Text F_1	5.66	5.82	-0.16 [-0.56, 0.08]
3B→14B	Encoder F_0	2.64	-0.55	3.19 * [1.74, 4.55]
3B→14B	Encoder F_1	5.51	5.82	-0.31 [-0.58, 0.08]
7B→32B	Text F_0	5.65	4.30	1.35 [0.16, 2.74]
7B→32B	Text F_1	8.51	8.51	0.00 [-0.71, 0.67]
7B→32B	Encoder F_0	5.10	4.06	1.04 [-0.15, 2.58]
7B→32B	Encoder F_1	8.27	7.80	0.48 [0.08, 1.45]
14B→32B	Text F_0	5.30	4.42	0.88 [-0.40, 2.12]
14B→32B	Text F_1	7.37	7.45	-0.08 [-0.79, 0.68]
14B→32B	Encoder F_0	4.98	3.95	1.03 [-0.23, 2.26]
14B→32B	Encoder F_1	7.68	7.29	0.40 [-0.38, 1.16]

D.2 Transfer to an excluded repository

For each of 11 held-out repositories, training uses only original-fit trajectories from the other ten. Evaluation uses all trajectories of the target repository across the original splits. Problem instances never overlap between training and evaluation. Table 8 weights repositories equally; the primary held-out experiment weights trajectories equally. Both error and advantage GBDT are evaluated

under the same folds. For error GBDT, mean R_{30} is 3.97, 5.06, 7.01, 6.92, respectively, in the four configurations.

Table 8: Identical leave-one-repository-out folds; repository-equal means. Error and advantage use the same base learner and features.

Pair	Error R_{30}	Error AURC	Adv. R_{30}	Adv. AURC
1.5B→7B	3.97	2.39	4.10	2.48
3B→14B	5.06	2.98	5.57	3.46
7B→32B	7.01	4.20	6.52	3.65
14B→32B	6.92	4.03	6.30	3.78

For advantage GBDT, all 11 folds have positive R_{30} and AURC in each pair. The minimum fold AURC, multiplied by 100, is 1.75, 0.98, 1.44, and 2.75 in pair order.

Each outer fold excludes one repository from both the original fit and calibration partitions, and evaluates all trajectories from that repository. Text vocabularies and scaling are refitted without held-repository inputs. The same 11 folds and four original model pairs are used for every score. Table 9 gives equal-repository means and positive-fold counts. Counts are descriptive: overlapping training sets do not constitute independent repeated trials.

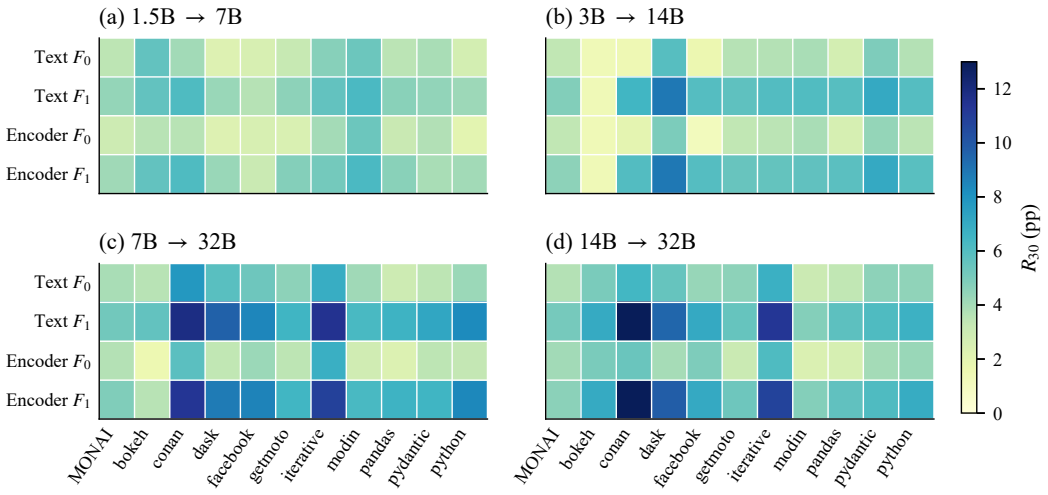

Figure 5: **Where transfer gains occur.** Each cell is a held-repository R_{30} estimate for advantage ridge; every panel uses the same color scale. All 176 estimates are positive. Columns denote repository groups; MONAI and facebook abbreviate Project-MONAI and facebookresearch. Table 9 includes matched error predictors.

Table 9: Repository transfer: mean R_{30} in points, with positive-fold counts in parentheses. Full per-repository R_{30}, AURC, accuracy, BA, and recalls are retained in the machine-readable results.

Score	1.5B→7B	3B→14B	7B→32B	14B→32B
Text F_0 / advantage	3.77 (11/11)	3.32 (11/11)	4.80 (11/11)	4.70 (11/11)
Text F_0 / error	0.79 (8/11)	-0.83 (1/11)	3.83 (11/11)	3.83 (11/11)
Text F_1 / advantage	4.88 (11/11)	5.84 (11/11)	7.91 (11/11)	7.41 (11/11)
Text F_1 / error	4.80 (11/11)	5.82 (11/11)	7.94 (11/11)	7.50 (11/11)
Encoder F_0 / advantage	3.27 (11/11)	3.12 (11/11)	3.73 (11/11)	4.18 (11/11)
Encoder F_0 / error	1.27 (10/11)	-0.50 (1/11)	3.67 (11/11)	3.96 (11/11)
Encoder F_1 / advantage	4.73 (11/11)	5.66 (11/11)	7.48 (11/11)	7.35 (11/11)
Encoder F_1 / error	4.48 (11/11)	5.54 (11/11)	7.06 (11/11)	7.09 (11/11)

D.3 FROZEN THRESHOLDS

For word/character ridge, calibration targets a 30% call fraction and fixes the score threshold and uniform boundary selection. Table 10 reports the resulting test rates and gains relative to random allocation at those same realized rates. This complements the exact-quota curves without using test outcomes to choose deployment thresholds.

Table 10: Frozen calibration thresholds for text-advantage routing. Calibration targets 30% calls, then fixes the threshold and boundary randomization. Deployment rates are allowed to vary. Gain compares with random allocation at the same actual test fraction; percentages throughout.

Pair	F_0 calls	F_0 gain	F_1 calls	F_1 gain
1.5B→7B	28.80	2.79	30.95	4.43
3B→14B	31.26	3.10	31.11	5.57
7B→32B	33.33	5.60	32.86	8.21
14B→32B	32.62	5.23	32.78	7.20

E FINAL DECISIONS AND THE AVAILABLE ACTION SET

E.1 CLASS-SENSITIVE EVALUATION

Table 11 specifies what the SWE action utilities measure. There are 72 resolved and 1,185 unresolved test trajectories. Accuracy is the agreement of a judge with these outcomes, while balanced accuracy averages the two class recalls. A constant unresolved judgment has accuracy 94.27%, balanced accuracy 50%, and resolved recall 0%. The allocation curves compare the declared two model actions at the same upgrade fraction; class-sensitive metrics complement that comparison.

Table 11: Model judgment metrics on the SWE test set (%).

Model	Accuracy	Balanced acc.	Resolved recall	Unresolved recall
Qwen 1.5B	60.38	67.90	76.39	59.41
Qwen 3B	60.22	71.73	84.72	58.73
Qwen 7B	65.87	76.68	88.89	64.47
Qwen 14B	67.62	76.96	87.50	66.41
Qwen 32B	79.40	74.72	69.44	80.00

To target balanced accuracy during fitting, let π_{fit} be the resolved frequency in fit and weight each correct judgment by $1/(2\pi_{\text{fit}})$ or $1/[2(1-\pi_{\text{fit}})]$ according to its gold class. Regress the resulting paired utility difference on the same 13 features. Test labels are used only for evaluation. Invalid judgments count as incorrect, including in the denominator of their class recall.

Final-label GBDT controls train on fit labels, using either nine pre-call structural features or all 13 cheap-output features. Classification thresholds maximize accuracy or BA on calibration only. Label-remapping controls select a map from the three cheap prediction states (invalid, unresolved, resolved) on fit. These actions require no strong-model call. All accuracy-optimized controls choose constant unresolved on this test set (94.27% accuracy, 50% BA). The class-balanced controls in Table 13 show why allocation gains within a model pair should be distinguished from superiority over a larger action set.

E.2 INEXPENSIVE DECISIONS AND REMAINING CALL VALUE

We retain the existing 2,552/1,292/1,257 fit/calibration/test trajectories and their instance-disjoint split. The candidate list is constant unresolved, constant resolved, the cheap judge, pre-call structural GBDT, cheap-output GBDT, and pre-call text logistic regression. GBDT uses the preceding control's fixed settings. The text classifier uses TF–IDF with at most 12,000 features, minimum document frequency 3, word unigrams/bigrams and sublinear term frequency, followed by balanced logistic regression ($C = 1$, liblinear, at most 500 iterations). It reads only the trajectory field, including its full available text; no judge output or benchmark label is appended. No explicit repository

Table 12: Routed class metrics at 30% upgrading. BA gain is against equal-rate random allocation; 5,000 instance-bootstrap draws.

Pair	Policy	Acc.	BA	Recall+	Recall–	BA gain [95% CI]
1.5B→7B	Random	62.03	70.53	80.14	60.93	0.00 [0.00, 0.00]
1.5B→7B	Acc.-Ridge	66.27	76.24	87.50	64.98	5.71 [3.07, 8.79]
1.5B→7B	BA-GBDT	65.12	76.28	88.89	63.68	5.75 [2.87, 8.62]
3B→14B	Random	62.44	73.30	85.56	61.04	0.00 [0.00, 0.00]
3B→14B	Acc.-Ridge	67.70	78.30	90.28	66.33	5.01 [1.72, 8.10]
3B→14B	BA-GBDT	66.35	77.59	90.28	64.89	4.29 [1.53, 7.43]
7B→32B	Random	69.93	76.09	83.06	69.13	0.00 [0.00, 0.00]
7B→32B	Acc.-Ridge	78.28	75.44	72.22	78.65	-0.66 [-4.18, 2.52]
7B→32B	BA-GBDT	73.75	78.90	84.72	73.08	2.81 [0.32, 5.10]
14B→32B	Random	71.15	76.29	82.08	70.49	0.00 [0.00, 0.00]
14B→32B	Acc.-Ridge	78.68	74.34	69.44	79.24	-1.94 [-5.64, 1.31]
14B→32B	BA-GBDT	73.99	77.73	81.94	73.51	1.44 [-0.92, 3.55]

Table 13: Lightweight final-label alternatives. Classifier thresholds use calibration labels only; remapping is fit-selected. All values in percent.

Pair	Final action	Acc.	BA	Recall+	Recall–
1.5B→7B	Remap	61.18	74.84	90.28	59.41
1.5B→7B	Pre-call	66.11	74.85	84.72	64.98
1.5B→7B	Cheap-output	72.00	76.02	80.56	71.48
3B→14B	Remap	60.62	75.20	91.67	58.73
3B→14B	Pre-call	66.11	74.85	84.72	64.98
3B→14B	Cheap-output	69.93	78.83	88.89	68.78
7B→32B	Remap	65.95	76.72	88.89	64.56
7B→32B	Pre-call	66.11	74.85	84.72	64.98
7B→32B	Cheap-output	73.75	78.90	84.72	73.08
14B→32B	Remap	67.62	76.96	87.50	66.41
14B→32B	Pre-call	66.11	74.85	84.72	64.98
14B→32B	Cheap-output	77.96	75.92	73.61	78.23

feature is supplied, although names may appear in the trajectory. Vocabulary and coefficients fit only on fit. Calibration maximizes BA over probability thresholds and selects the best candidate, resolving ties by the fixed candidate order. The selected text classifier has test accuracy 75.82%, BA 79.35%, resolved recall 83.33%, and unresolved recall 75.36%.

For residual training, three stratified instance-grouped folds supply out-of-fold predictions for the selected cheap classifier, using the calibration-selected cutoff. A GBDT regresses the fit-prevalence-weighted correctness difference between B and this prediction. Router inputs are the original 13 pre-B features plus the cheap classifier's probability and label, also out of fold for fitting. Evaluation uses the full-fit classifier. The fusion experiment replaces B's terminal label by a GBDT trained on the concatenated 13 cheap-judge and 13 strong-judge features; its cutoff is calibrated for BA. Out-of-fold predictions for this endpoint form its residual training target. The routing features never include the strong response.

Calibration chooses a rate in $\{0, .01, .025, .05, .1, .2, .3\}$ maximizing BA minus a price times the call fraction; ties choose the smaller rate. A score threshold and uniform boundary probability are then fixed before test. We report all prices $\{0, .01, .025, .05, .1, .2\}$ in the artifacts; Table 14 gives zero penalty, the unpenalized endpoint. These are marginal strong-call prices after the common cheap prefix, not measured from-scratch costs. A zero-rate policy can skip the unused prefix; any positive-rate router shown here purchases the cheap observation on every input. Intervals use 5,000 paired problem-instance resamples with the fitted policies and thresholds fixed. All primary nonzero gains have intervals containing zero; Holm correction across the four pairs within each endpoint construction rejects none. A degenerate zero interval indicates the selected no-call policy is exactly its baseline, not a population equivalence test.

Table 14: Remaining value over the calibration-selected text-only classifier. Strong-call fraction and BA are percentages; gain intervals are BA percentage points. Calibration selects the rate without test outcomes.

Pair	Paid endpoint	Strong calls	BA	Gain [95% CI]
1.5B→7B	Label	1.67	79.53	0.19 [-0.75, 1.82]
3B→14B	Label	0.00	79.35	0.00 [0.00, 0.00]
7B→32B	Label	10.02	80.19	0.84 [-2.75, 3.99]
14B→32B	Label	22.12	77.08	-2.27 [-6.61, 1.91]
1.5B→7B	Fusion	0.00	79.35	0.00 [0.00, 0.00]
3B→14B	Fusion	5.89	79.56	0.21 [-1.81, 2.21]
7B→32B	Fusion	0.00	79.35	0.00 [0.00, 0.00]
14B→32B	Fusion	5.17	79.68	0.34 [-1.73, 2.34]

E.3 Richer classifiers and joint deferral

Balanced logistic classifiers ($C = 1$, maximum 400 iterations) are fitted to each of the four representations. Probability cutoffs maximize calibration BA. Selection among these four candidates uses calibration only. The word/character and encoder candidates form a separate family from the word-only classifier above. Joint policies are compared against the word-only reference (79.35% BA).

The residual regressor uses three instance-grouped folds to obtain out-of-fold cheap predictions, and targets fit-prevalence-weighted correctness improvement over the calibration-selected candidate. Vocabulary and scaling use outer-fit inputs and are shared across inner folds; label predictors are out of fold. Routing features contain the original vector plus the selected classifier's probability and label. All prices $\{0, .01, .05, .1\}$ and calibration rates $\{0, .01, .025, .05, .1, .2, .3\}$ are retained. The rate maximizes calibration BA minus price times call fraction, with lower-rate tie breaking. Test uses the resulting frozen threshold.

Joint policies use a linear softmax implementation of the consistent classification–deferral surrogate of Mozannar & Sontag (2020): a class-weighted $-\log p_y$ term for each fit item and an additional $-\log p_{\text{defer}}$ term when the strong judgment is correct. We instantiate this established loss with linear text features. Calibration selects the class cutoff and a threshold on $p_{\text{defer}} - p_{\text{chosen class}}$ using the same rate/price grids. Since thresholds are frozen, actual test call fractions can exceed 30%. F_1 policies additionally acquire the cheap judgment on every input.

Table 15: Joint policies at zero call penalty versus the word-only classifier. Calls and BA are percentages; difference intervals are BA points. All policies are unchanged from calibration; the reference classifier has BA 79.35% in every row. All 32 feature/pair/price settings are retained, with Holm correction across 32 comparisons.

Pair	Features	Calls	BA	Difference [CI]
1.5B→7B	Text F_0	32.62	80.88	1.53 [-2.05, 5.47]
1.5B→7B	Text F_1	31.74	80.33	0.99 [-2.60, 4.94]
3B→14B	Text F_0	33.65	80.36	1.01 [-2.98, 5.24]
3B→14B	Text F_1	32.46	80.40	1.05 [-2.33, 4.61]
7B→32B	Text F_0	34.84	77.54	-1.80 [-7.06, 2.99]
7B→32B	Text F_1	4.85	80.69	1.35 [-3.02, 6.04]
14B→32B	Text F_0	34.84	77.54	-1.80 [-7.06, 2.99]
14B→32B	Text F_1	34.37	78.15	-1.19 [-6.62, 3.93]

E.4 Distilling a fixed policy

The distillation teacher is the original GBDT advantage router with a calibration-frozen 30% threshold. Three grouped out-of-fold teacher routers generate fit targets, with deterministic hash-based draws for fractional selection. Multinomial logistic students learn the resulting terminal labels (including invalid outputs as a separate class). F_0 students make no judge calls; F_1 students use the

Table 16: Sensitivity of joint-policy BA to call penalty. Each cell gives test BA / strong-call percentage. Price selection and thresholds use calibration only.

Pair	Features	0	0.01	0.05	0.10
1.5B→7B	Text F_0	80.88 / 32.62	80.88 / 32.62	81.49 / 21.80	81.81 / 10.58
1.5B→7B	Text F_1	80.33 / 31.74	80.33 / 31.74	80.98 / 23.47	80.98 / 23.47
3B→14B	Text F_0	80.36 / 33.65	80.92 / 21.64	82.60 / 4.77	82.60 / 4.77
3B→14B	Text F_1	80.40 / 32.46	80.40 / 32.46	80.84 / 11.61	80.63 / 4.69
7B→32B	Text F_0	77.54 / 34.84	77.54 / 34.84	81.15 / 0.72	81.15 / 0.72
7B→32B	Text F_1	80.69 / 4.85	80.69 / 4.85	80.69 / 4.85	80.69 / 4.85
14B→32B	Text F_0	77.54 / 34.84	77.54 / 34.84	81.15 / 0.72	81.15 / 0.72
14B→32B	Text F_1	78.15 / 34.37	78.15 / 34.37	78.15 / 34.37	78.65 / 2.70

Table 17: Residual routing against the calibration-selected word/character or encoder classifier. Each cell gives BA / strong-call percentage. Selected-baseline BA is shown explicitly; the word-only reference has 79.35% BA.

Pair	Selected BA	0	0.01	0.05	0.10
1.5B→7B	77.63	76.20 / 29.28	76.20 / 29.28	76.29 / 19.33	77.55 / 1.19
3B→14B	75.80	74.92 / 29.81	74.92 / 10.10	75.40 / 4.53	75.19 / 0.88
7B→32B	75.81	76.48 / 5.33	76.48 / 5.33	76.48 / 5.33	76.48 / 5.33
14B→32B	75.14	75.31 / 1.19	75.31 / 1.19	75.31 / 1.19	75.31 / 1.19

cheap judge and no strong judge. Agreement and final BA measure different properties. Table 18 reports both; distillation is not selected by test performance.

Table 18: Distillation of the fixed teacher policy. BA and teacher agreement are percentages. Student inference makes no strong calls; teacher supervision still uses paired cached outcomes.

Pair	Teacher BA	F_0 BA	F_0 agree	F_1 BA	F_1 agree
1.5B→7B	74.77	78.89	92.20	76.19	94.99
3B→14B	77.65	79.48	93.56	77.67	96.58
7B→32B	77.31	77.31	89.02	79.58	91.49
14B→32B	75.56	76.87	88.23	78.93	92.28

E.5 Expanded actions under repository shift

For the expanded-action transfer control, we additionally refit the original word-only classifier within each outer fold. Its mean repository BA is 76.83%. Table 19 retains every repository for joint, residual, and distilled policies at zero call penalty. All policies use the same word-only reference. The table reports descriptive BA differences across the overlapping repository folds.

Table 19: Expanded-action transfer relative to the word-only classifier trained without the target repository. Cells give mean BA difference in points and positive-fold counts. F_1 acquires the cheap output; positive-rate joint/residual policies also call the strong judge. Distilled policies make no strong calls. All price settings remain in the result archive.

Policy	1.5B→7B	3B→14B	7B→32B	14B→32B
Residual	-0.84 (5/11)	1.02 (2/11)	0.49 (4/11)	-0.10 (7/11)
Joint F_0	-4.24 (3/11)	-3.87 (3/11)	-2.76 (4/11)	-2.76 (4/11)
Joint F_1	-0.75 (4/11)	-0.54 (4/11)	-2.36 (3/11)	-3.20 (5/11)
Distilled F_0	0.76 (5/11)	0.75 (4/11)	1.28 (7/11)	-1.43 (6/11)
Distilled F_1	-1.36 (4/11)	1.83 (5/11)	0.36 (4/11)	2.02 (8/11)

F THE COST OF ACQUIRING ROUTING INFORMATION

The input-only direct router uses F_0, while the cascade first obtains the cheap response and routes with F_1. With cheap-call price one and strong-call price c_s, equal expected spending matches a cascade rate $r = 0.3$ to a direct rate $q = 0.3c_s/(c_s - 1)$. These are comparisons under declared call prices; feature extraction and hardware latency are outside this price model.

F.1 FULL-TEXT ROUTING

Both policies use the same word/character representation and ridge penalty. Each bootstrap draw resamples problem instances and re-ranks both policies at their respective equal-price budgets. Figure 6 shows all five price ratios and paired intervals. At ratios of at least five, the cascade has a positive interval in every pair. At ratio two, the two 32B configurations have negative point estimates and intervals crossing zero: the value of the acquired response depends on the direct calls that its cost displaces.

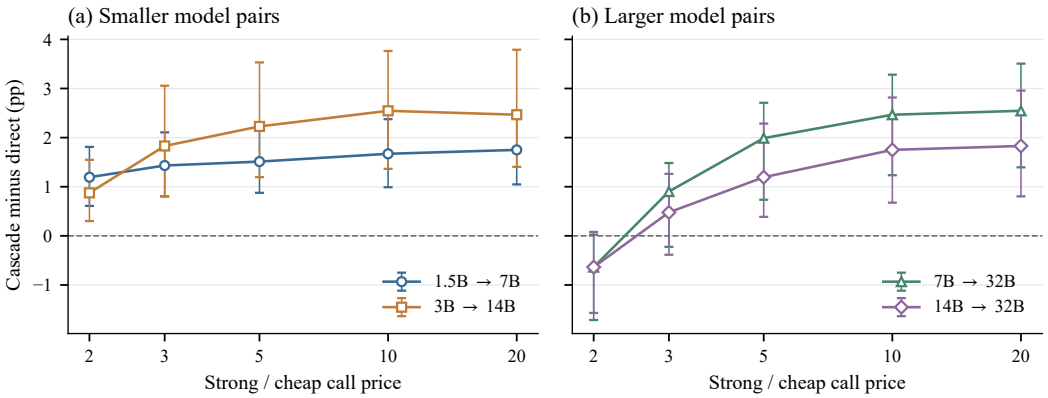

Figure 6: **Acquisition value depends on relative call price.** Points compare the cascade with the informed direct text router using the same frozen outcomes. Vertical bars are pointwise paired 95% intervals. Lines connect the five declared price settings; they are not a measured GPU-cost frontier. Both policies have the same expected call expenditure.

F.2 COMPACT STRUCTURAL ROUTING

The pre-call model uses trajectory length and eight structural counts, excluding cheap prediction, confidence, confidence distance, and validity. Both routers use the same GBDT configuration and fit outcomes. The cascade rate is 0.3 and the direct rate is $0.3c_s/(c_s - 1)$. Paired instance bootstrap reranks both policies at their respective rates in each draw. Table 20 gives the corresponding compact-feature comparison.

Table 20: Cascade accuracy gain over an informed direct router at equal expected price (pp). Last column: 95% interval at $c_s = 10$.

Pair	$c_s = 2$	3	5	10	20	CI at 10
1.5B→7B	0.32	0.66	1.35	1.59	1.91	[0.87, 2.37]
3B→14B	0.64	1.67	1.83	2.07	2.31	[1.31, 3.06]
7B→32B	-0.24	1.59	2.78	3.42	3.82	[2.41, 4.53]
14B→32B	0.08	1.03	2.63	3.18	3.26	[1.94, 4.17]

G Answer Correctness and Sequential Sampling

G.1 Objective answer-level evaluation

The binary-score subset uses all 57 MMLU subjects, HellaSwag, ARC-Challenge, Winogrande, and MBPP. Seven repeated normalized prompts are removed before the existing sample-ID hash split; no identical normalized prompt crosses fit and test. Fractionally graded GSM8K and MT-Bench entries in the complete RouterBench release are excluded from this accuracy analysis. The released correctness scores concern the model's answer, not a prediction of whether another answer is correct. We reuse these scores rather than independently regrading the generations.

TF–IDF uses word unigrams/bigrams, minimum document frequency 3, at most 12,000 features, and sublinear term frequency; its vocabulary is fit-only. Ridge uses penalty 10 and LSQR, fitting each model's binary outcome. Subtracting the linear quality predictions is equivalent to direct linear advantage regression with the same design and penalty. Pair orientation uses fit mean accuracy. All 55 pairs are reported in the machine-readable archive; the three illustrated pairs were specified before this analysis. Text-error uses the same representation and base learner. Category means use 20-observation shrinkage. Bootstrap draws resample questions and re-evaluate budget thresholds. We compute 5,000-draw intervals for the three representative pairs and all category-only pairs; positivity of the other text-pair point estimates is not a simultaneous significance claim.

After deduplication, the binary-outcome subset contains 27,241 questions: 13,644 fit, 6,825 calibration, and 6,772 test. This differs from the complete graded-utility release in Appendix B. Table 21 breaks down the represented-family evaluation, with a separate 30% allocation within each family.

Table 21: Answer allocation within represented task families. Entries are R_{30} in points; each family receives its own 30% budget.

Task family	Test n	Mistral→Mixtral	Mistral→GPT-3.5	GPT-3.5→GPT-4
arc-challenge	362	0.99	1.10	-0.33
hellaswag	2486	3.44	1.81	2.47
mbpp	110	-1.00	-3.00	-0.73
mmlu	3515	4.25	2.90	2.82
winogrande	299	1.94	0.40	0.77

G.2 Within-task allocation and unseen task families

The three representative RouterBench pairs and their orientations are retained from the preceding answer-level evaluation. The within-task control fixes the upgrade fraction to 30% separately for every `eval_name`, including individual MMLU subjects. Word TF–IDF and ridge models are fit on the original 13,644 fit questions. Each of 5,000 paired bootstrap replicates resamples within each category, keeps its sample count fixed, and recomputes the category's ranking and quota. The weighted average uses the original task mixture. Holm-adjusted $p < 0.001$ holds for all three within-task gains. This control isolates allocation among questions from the same task.

A separate transfer test removes every question from one task family before fitting the vocabulary and predictors. Evaluation uses only that family's questions in the original test partition. We retain all five families and all three original pair orientations (Table 22). Within-task allocation on represented tasks and transfer to an unseen task family are distinct properties; the latter is heterogeneous here.

G.3 Incremental answer sampling

The 14B actor supplies nested prefixes of one, four, and eight independently graded answer samples. A majority vote chooses the answer; ties select the earliest candidate. Table 23 evaluates the two adjacent upgrades using prefix-aware advantage ridge and a 30% quota. Their small or uncertain gains motivate evaluating the complete stopping decision alongside each local upgrade.

Table 22: Held-out task-family answer allocation: R_{30} and pointwise 95% intervals in points. Family training excludes all target-family items.

Held family	Test n	Mistral→Mixtral	Mistral→GPT-3.5	GPT-3.5→GPT-4
arc-challenge	362	1.27 [-1.55, 3.78]	0.72 [-1.96, 3.65]	0.50 [-1.24, 2.10]
hellaswag	2486	0.14 [-0.64, 0.96]	-0.19 [-1.59, 1.13]	-0.47 [-1.57, 0.63]
mbpp	110	-1.91 [-5.55, 2.00]	-2.09 [-7.00, 1.46]	0.18 [-3.27, 3.91]
mmlu	3515	-0.01 [-0.81, 1.02]	-0.51 [-1.43, 0.45]	1.36 [0.51, 2.10]
winogrande	299	1.61 [-0.47, 3.61]	-0.94 [-4.65, 2.37]	-0.57 [-3.18, 2.47]

Table 23: Fixed actor with majority answers from nested prefixes of 1, 4, or 8 samples. Prefix-aware advantage Ridge, 30% upgrading; 5,000 paired draws. Tied votes select the earliest candidate, without gold-label selection.

Task	Samples	n	Random	Routed	Gain [95% CI]
GSM8K	1→4	327	95.50	95.72	0.21 [-0.43, 1.07]
GSM8K	4→8	327	95.81	96.02	0.21 [0.00, 0.64]
MATH500	1→4	118	85.51	87.29	1.78 [0.00, 4.15]
MATH500	4→8	118	87.03	86.44	-0.59 [-1.78, 0.51]

G.4 Stopping and continuation

We use nested 1/4/8-sample 14B caches with independently graded final answers. Fit and calibration follow the original question split; test has 118 MATH500 and 327 GSM8K questions. At each decision, inputs are question TF–IDF (fit-only vocabulary, 3,000 features) and observed-prefix vote share, number of answer classes, valid-parse fraction, log cumulative generated tokens, and mean ANLL. Future answers, future features, and correctness are excluded from policy inputs.

Backward fitted value comparisons first estimate stopping versus continuing at four samples, then at one sample. Separate ridge regressors (penalty 10) estimate the stop value and the continuation target; costs are measured in units of median one-sample tokens on fit. The continuation target uses the fitted downstream maximum. A myopic baseline regresses the adjacent-budget correctness increment minus token price; the vote rule always reaches four samples and continues when the leading vote share is below 0.75. All five prices $\{0, .005, .01, .02, .05\}$ are reported. Zero price is the unpenalized endpoint of this fixed grid.

At zero price, continuation saves 62.4% of generated tokens relative to fixed eight on MATH500, with an observed accuracy difference of 0.85 points (paired 95% interval $[0, 2.54]$). On GSM8K it saves 40.5% and agrees with fixed eight in correctness on every test item. The degenerate bootstrap interval in the latter case reflects identical observed outcomes, not a population noninferiority guarantee. On MATH500, continuation and fixed four also agree in observed correctness, with 24.8% fewer tokens. Table 24 gives the accuracy and generated-token expenditure for every policy setting. Figure 7 visualizes the fixed policies and the zero-penalty comparison. The cost metric is generated tokens; fitting costs and hardware latency are separate quantities.

Table 24: Complete stopping-policy grid. Accuracy is in percent and tokens are mean generated tokens per question. MATH500 has 118 test questions; GSM8K has 327. The price-independent vote rule is printed once.

		MATH500		GSM8K	
Policy	λ	Accuracy	Tokens	Accuracy	Tokens
Fixed 1	–	84.75	4,060.3	95.41	1,976.3
Fixed 4	–	87.29	16,309.0	95.72	7,958.9
Fixed 8	–	86.44	32,643.7	96.02	15,946.5
Vote	0	86.44	17,906.6	95.72	8,108.8
Myopic	0	87.29	14,407.3	96.02	9,661.1
Myopic	0.005	87.29	7,750.0	96.02	2,394.3
Myopic	0.01	87.29	7,594.7	96.02	2,229.7
Myopic	0.02	85.59	5,048.0	95.41	1,990.6
Myopic	0.05	84.75	4,060.3	95.41	1,990.6
Continuation	0	87.29	12,266.6	96.02	9,489.2
Continuation	0.005	87.29	8,204.2	96.02	2,408.2
Continuation	0.01	87.29	7,909.4	96.02	2,229.7
Continuation	0.02	87.29	7,617.4	96.02	2,224.5
Continuation	0.05	84.75	4,103.1	95.41	1,990.6

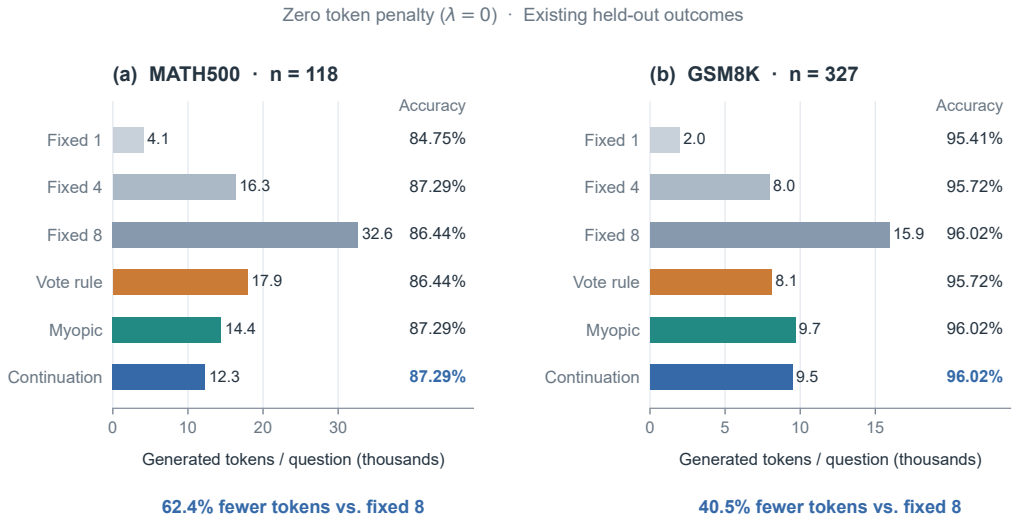

Figure 7: **Generated-token expenditure and answer accuracy at zero token penalty.** Bars show mean generated tokens per test question; the adjacent column reports observed accuracy. All fixed policies and all policies at $\lambda = 0$ are shown. Continuation uses 62.4% fewer tokens than fixed eight on MATH500 and 40.5% fewer on GSM8K. These are held-out point estimates, not a population noninferiority guarantee; the text reports paired uncertainty, and Table 24 retains the complete penalty grid. Costs exclude fitting and hardware latency.

H Complementary Judges and Verification Actions

The Qwen2.5-Coder-1.5B/Gemma-3-4B SWE pairing illustrates the specialization mechanism. Across all 5,101 trajectories, the cheap "not resolved" class contains 57.4% of examples, with cheap/strong accuracies of 99.4%/96.1%. The "resolved" class contains 42.6%, with accuracies of 10.8%/31.7%. This latter class receives confidence 1.00 yet offers the larger audit return. These descriptive slices explain the mechanism; policy gains are evaluated on held-out data.

Table 25 reports this pairing and two complementary RewardBench 2 (Malik et al., 2026) judge settings. The RewardBench action B is Qwen3-8B; action A is Qwen2.5-Coder-1.5B or Gemma-

Table 25: **Held-out allocation gain at a 30% upgrade budget.** Values are accuracy percentage points over same-rate random upgrading. Cheap-output scores use prediction and/or confidence; advantage scores add pre-call metadata. Logistic and isotonic cheap-error scores are reported separately. Bold marks the largest point estimate in each row.

	Cheap-output scores				Advantage + metadata	
Setting	Conf.	Bit	Logistic	Isotonic	Ridge	GBDT
SWE-Gym	-2.96	3.36	3.36	3.36	**4.62**	4.30
RewardBench 1.5B	-0.62	1.72	0.55	0.94	4.26	**5.23**
RewardBench 4B	1.98	**3.94**	3.35	3.35	3.16	2.96

Table 26: **SWE routing retains value after paying for its signal.** The same one-bit policy is evaluated at five strong-to-cheap cost ratios. Gains and 95% problem-instance bootstrap intervals are percentage points against a direct cost-matched mixture. Predictions are fixed across these cost scenarios.

Strong/cheap cost ratio c_s	2	3	5	10	20
Test gain	2.96	4.35	5.04	5.43	5.59
95% interval	$[1.99, 4.00]$	$[3.50, 5.26]$	$[4.17, 5.98]$	$[4.52, 6.40]$	$[4.66, 6.58]$

3-4B. Item-level fit/calibration/test counts are 884/464/512 and 885/464/512, respectively. The accompanying SWE budget-curve study groups instances, with 2,588/1,247/1,266 trajectories. These comparisons use the complete score configurations specified in this appendix. SWE ridge gains 4.62 points $[3.50, 5.85]$; RewardBench 1.5B GBDT gains 5.23 $[2.53, 7.93]$, and the RewardBench 4B bit rule gains 3.94 $[1.98, 5.93]$.

H.1 Judge-pair protocol

The complementary judge study uses the following fixed protocol; an input manifest records the paired-cache hashes. Fit, calibration, and test membership are assigned by a deterministic SHA-256 hash of the declared group identifier with salt `compute-upgrade-v1`. SWE uses problem instance identifiers, yielding 1,148/550/553 groups across the three splits. RewardBench uses item identifiers. All reported batch comparisons in Table 25 use the same split and quota within each setting.

The feature vector for cheap-error prediction is the numerical cheap label, verbal confidence, and absolute distance of confidence from $1/2$. Logistic regression uses regularization parameter $C = 1$ and a maximum of 2,000 iterations. The isotonic score is an increasing fit-only mapping of the logistic error probabilities. A monotone mapping preserves strict score ordering but can create ties; these ties receive the same fixed hash rule used elsewhere.

Advantage predictors append two deterministic numerical encodings of the repository or RewardBench subset identifier, each obtained from a separately salted hash. These features are available before the strong call. Ridge regression uses penalty 10. Histogram gradient boosting uses maximum depth 2, 60 iterations, and ℓ_2 regularization 3. The random seed is 20260917. Table 25 compares these complete scores, including their stated feature sets, rather than a feature-matched intervention on the training target.

The evaluated rates are $\{0, .05, .10, .20, .30, .40, .50, .70, .90, 1\}$. For n test items, each score selects $\mathrm{round}(rn)$, ordered first by descending score and then by a fixed hash of item identifier. Equation 13 uses the realized fraction. Ranking uses the fitted score and item identifier. Figures connect the measured rates and report these exact-quota batch allocations.

At the cited operating points, 1,000 bootstrap draws resample declared test groups with replacement. SWE draws entire problem-instance clusters and retains all their trajectories; RewardBench draws items. The interval applies to the mean observed contrasts $\Delta_i(S_i - \widehat{r})$ with the evaluated allocation and rate held fixed. It describes uncertainty at that operating point and does not construct a simultaneous band over methods or budgets.

H.2 THE FIXED ONE-BIT POLICY

The SWE one-bit policy and cost sweep use the original instance-hash split: 2,552 fit, 1,292 calibration, and 1,257 test trajectories. The complementary judge-curve study uses its separately specified split. The fit-selected rule audits the cheap "resolved" class; on test it upgrades 548 trajectories. The original logistic policy and the bit rule agree on fit and calibration and differ on one test item; the table consistently uses the bit rule. The class-conditional descriptions summarize all 5,101 non-pilot trajectories.

The one-bit intervals resample the 553 test problem instances with replacement for 2,000 draws, retaining every trajectory within each sampled instance. Each draw recomputes the router's upgrade rate, the direct mixture probability from Equation 9, and the trajectory-weighted gain. All resampled mixtures are feasible for the five displayed cost ratios. Percentile endpoints give the pointwise 95% intervals. These intervals quantify a frozen rule's held-out gain while accounting for trajectories sharing a problem instance.

The split and prompt perturbations retain their saved learned-router evaluation protocols. The SWE 20-split values and leave-one-repository-out mean are stability summaries. They are separate from the one-bit replay and intervals in Table 26. The fresh cheap-side passes change the cheap observations while retaining the cached strong audit and the same allocation question.

H.3 RETRIEVAL FOR FACTUALITY

We use the public annotated FaStfact-Bench release of Wan et al. (2025), from the upstream `Yingjia-Wan/FastFact` repository at commit `a852d43f6ea4`. The file `fastfact-bench_annotated.jsonl` contains 400 documents. The descriptive coverage analysis retains 380 documents and 6,968 claims: `supported` is positive, while refuted, insufficient-evidence, conflicting-evidence, unverifiable, unsupported, and irrelevant labels are negative; non-claim and unrecognized labels are excluded. Cheap decisions, eligibility (`pre-label` equal to `unsure`), and retrieval outcomes come from the release's stored fields, including `verification_result`; we do not run fresh web retrieval. The supplementary `DATA_PROVENANCE.json` records the full commit, file SHA-256, and extraction rules. Document identifiers used in the locked split preserve the original filtered-file order.

Let n_d be the number of claims in document d, T_d the number of accepted supported claims, F_d the number of accepted unsupported claims, and C_d the retrieval-call count. The document utility is

$$U_d = \frac{T_d - 4F_d - 0.02C_d}{n_d}.$$

The comparator retains the cheap decisions and makes no retrieval calls. The paid rule retrieves on release-labelled `unsure` claims with stored verification. Supported coverage divides accepted supported claims by all supported claims; false accepts among accepts divide accepted unsupported claims by all accepted claims. These two descriptive rates have different denominators and are not terms to subtract directly as a utility.

The claim-level summary uses 380 documents, 6,968 claims, and 1,876 retrievals. Document bootstrap intervals for the coverage increase and false-accept-rate change are $[19.18, 23.83]$ and $[-2.54, -1.54]$ percentage points, respectively. The locked utility evaluation retains its original, narrower label mapping, recognizing only supported/refuted labels present in this release. It therefore uses 6,223 claims (1,388 retrievals) in the same 380 documents, split into 81 fit, 146 calibration, and 153 test documents. It reports the separate test gain 0.235 and calibration lower bound 0.129. These estimates do not use the descriptive coverage sample or its claim denominator. No outcomes for retrieving on non-eligible claims are required by this comparison.

The FaStFact one-sided calibration lower bound uses bounded paired document-utility contrasts and fit-predictable betting stakes at level 0.05. For a contrast $X \in [a, b]$, normalization $Y = (X - a)/(b - a)$ gives a bounded variable in $[0, 1]$. For a candidate null mean m, nonnegative betting factors $1 + \lambda(Y_i - m)$, with admissible stake fixed from fit information, form a test supermartingale under the corresponding conditional-mean null. Inverting the test yields a lower mean bound, then transforming back yields the utility bound (Waudby-Smith & Ramdas, 2024). Test effect sizes are reported separately from the calibration bound.

H.4 SYMBOLIC AND MODEL VERIFICATION

The 720-item math cache contains symbolic algebra and word problems, with original fit/calibration/test sizes 363/216/141. A pre-call context rule uses SymPy for algebra and Gemma-3-4B for word problems. A verifier flag means abstention. For acceptance indicator a, correctness label y, and call cost c, the utility is

$$U = a\big(y - 9(1-y)\big) - 0.02c.$$

SymPy costs one unit and the model judge four units. The direct mixture selects these actions independently of the input with model probability $q = (\bar{c}_{\text{router}} - 1)/3$, matching the router's average cost on the evaluated split. The test utility gain is 1.168. Across 20 alternative hash splits, the mean gain is 1.252 and the observed range is 0.964–1.487. These values summarize split sensitivity.

I EVIDENCE AND REPRODUCTION

The archive contains paired outputs, splits, input hashes, prompts, model settings, predictions, and complete numerical comparisons, including unfavorable outcomes and full parameter grids. Its root `MANIFEST.json` records file hashes; the README maps analyses to scripts and results and specifies dependencies and commands.

Plots use frozen records without refitting; the overview uses the supplied diagram. Acquisition times and token counts remain separate from declared call prices. Offline verification checks integrity, split separation, primary metrics, sampling records, and the FaStFact extraction and certificate.