PREDICTABLE PERFORMANCE IS SUBOPTIMAL AND WHAT IT MEANS FOR MODEL ROUTING

Anonymous authors
Paper under double-blind review

ABSTRACT

Large language models (LLMs) differ in capabilities and costs, motivating routing methods that adaptively choose a model for each query. These methods rely on *performance prediction*: given a query, estimate whether a model will answer it correctly. Most routing methods train separate predictors of candidate models' performance to guide model selection. Our work shows that performance prediction only beats a simple baseline when the candidate model is suboptimal on the task or the performance predictor adds significant capacity relative to the candidate model. Specifically, we take a model's self-confidence, the probability it assigns to its chosen answer, as a baseline and study how much a performance predictor can improve on it. First, we show that any performance predictor that outperforms a model's self-confidence can be used to improve the model's loss: the predictor's advantage transfers exactly in log loss through recalibration. Second, accounting for the predictor's additional parameters, we bound its advantage by the model's suboptimality plus the gain from those parameters. Third, we extend our theoretical results to two-model cascades, which route queries to a more capable model when the first model is predicted to fail. Experiments with 16 language models on four benchmarks show that predictors gain a larger advantage over self-confidence on more suboptimal models. This advantage shrinks as models improve through finetuning. Cascading exhibits the same two patterns: predictor-based routing offers larger gains over confidence-based routing for more suboptimal models, and these gains diminish after finetuning.

1 INTRODUCTION

Performance prediction estimates whether a model answers a query correctly (Kadavath et al., 2022; Shnitzer et al., 2024). These predictions can help routing systems decide when to trust an answer and when to call a stronger model (Geifman & El-Yaniv, 2017; Chen et al., 2024). By catching a weak model's errors accurately, routing can achieve better performance at the same cost. But what makes a model's errors predictable, and what does that reveal about the model?

In this paper, we study the predictability of a model's performance relative to a simple baseline: the model's *self-confidence*, the probability it assigns to its chosen answer. Our key observation is that *any predictor that beats self-confidence can improve the model by the same amount*. This means that a predictor's advantage over self-confidence is limited by the model's room for improvement. We formalize and test this connection, and extend the analysis to routing. Our contributions are:

Suboptimality bounds excess predictability. We connect performance prediction and model improvement through recalibration: replacing the model's confidence in its chosen answer with the predictor's estimate, and normalising the rest. We call the predictor's reduction in correctness log loss relative to self-confidence *excess predictability*. Recalibration reduces the model's answer log loss by exactly the same amount (Proposition 1). We assume a *scaling floor*: the lowest answer loss achievable on a fixed task at each parameter budget (Assumption 1). The recalibrated model cannot beat the floor at the combined parameter budget of the model and predictor. This gives the bound (Theorem 1)

$$\boxed{\textit{excess predictability} \;\leq\; \textit{suboptimality} \;+\; \textit{capacity gain}}\,.$$

Here, *suboptimality* is the model's gap to the floor at its own budget. The *capacity gain* is the decrease in the floor from adding the predictor's parameters.

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.

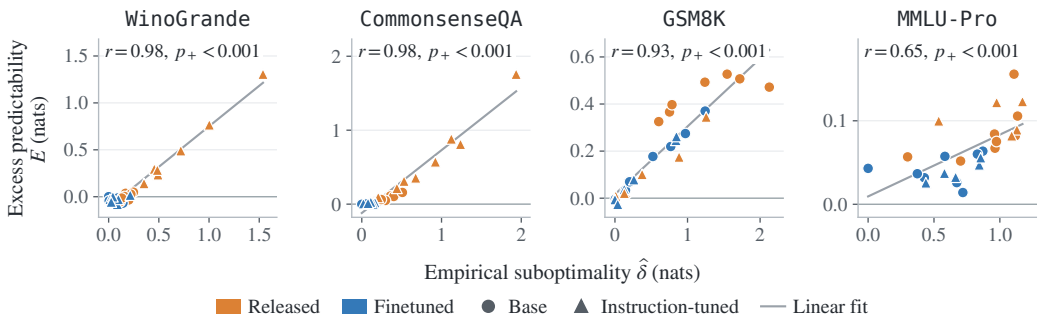

Figure 1: **Predictors have a larger advantage over self-confidence on more suboptimal models.** Each point is a released (orange) or finetuned (blue) model. Empirical suboptimality is the gap to the best observed answer log loss on each task. Excess predictability E is the reduction in correctness log loss achieved by a separately trained 4B predictor over self-confidence. Colors identify training stage; shapes identify the release type. Lines are least-squares fits; r is Pearson correlation and p_+ its one-sided p-value for positive correlation. See Section 3.4.1 for experimental details.

Greater excess predictability comes with greater suboptimality. Our upper bound does not imply that more suboptimal models are necessarily easier to predict beyond self-confidence. Empirically, however, we observe a positive association between the two. We conduct experiments on `WinoGrande`, `CommonsenseQA`, `GSM8K`, and `MMLU-Pro`, using 16 language models from eight families, ranging from 7 to 12B parameters. Using the gap to the best observed answer loss on each task as a proxy for suboptimality, we find a positive association with excess predictability on all four tasks, with Pearson correlations of 0.98, 0.98, 0.93, and 0.65, respectively (Figure 1). We also finetune the models, retaining the 59 model-task pairs whose development accuracy improves. Answer loss falls in all 59 retained pairs and excess predictability falls in 55 (Figure 2), despite training separate 4B predictors for the released and finetuned models.

Suboptimality bounds error capture beyond confidence routing. We extend our analysis to two-model cascades, where a router decides when to escalate a query to a stronger model. We compare a learned router with confidence routing, which escalates the weak model's lowest-confidence queries, at the same escalation rate. Any router that captures more weak-model errors can be converted into a predictor with positive excess predictability, even if the router does not explicitly estimate correctness (Lemma 1). This bounds the additionally captured weak-model errors by weak-model suboptimality and the capacity gain from the router's added parameters (Theorem 2). The cascade accuracy gain equals the additional weak-model errors captured minus the additional strong-model errors encountered. The same bound applies to the cascade accuracy gain if the learned router encounters at least as many strong-model errors as confidence routing (Corollary 2).

More suboptimal models have larger routing advantages. Using the same models and predictors, we compare predictor-based routing with confidence routing at equal escalation rates. We measure the *routing advantage* as the gain in cascade accuracy, averaged over escalation rates. Models with greater empirical suboptimality tend to have larger routing advantages on all four tasks, with Pearson correlations between 0.66 and 0.91 (Figure 4). This advantage shrinks after finetuning in 53 of the 59 retained pairs (Section 4.3).

Main takeaway. Together, these results suggest three sources of predictability: the model's self-confidence, its suboptimality, and the predictor's added capacity. Improving the underlying model generally reduces the additional value of learned prediction and routing beyond self-confidence.

2 Related Work

Performance prediction. Performance predictors, also called assessors, estimate whether a system will succeed on a given input (Hernández-Orallo et al., 2022; Zhou et al., 2022). They use model outputs (Yao & Koller, 2024), reference evaluations and question features (Pacchiardi et al., 2024),

follow-up queries (Sam et al., 2025), or internal activations before the model answers (Moreno Cencerrado et al., 2026). Kapoor et al. (2024) finetune language models on graded answers to predict their own correctness and that of other models. Xiao et al. (2026) find that models predict their own correctness no better than unrelated models do, and train predictors that work across models. Beyond binary correctness, Hsu et al. (2025) estimate continuous quality scores and uncertainty intervals for free-form generation. At the benchmark level, studies forecast performance from training compute (Owen, 2024) or scores on other benchmarks (Ruan et al., 2024). Others estimate benchmark scores from a small set of evaluated examples (Maia Polo et al., 2024; Zhang et al., 2025a). We study where predictability on individual inputs comes from, using self-confidence as the baseline and correctness log loss to measure a predictor's advantage.

Model routing and cascading. Performance predictions help routing systems balance cost and performance by choosing which models to use for each query (Ong et al., 2025). We use routing broadly to include cascading, where a cheaper model answers first and a router uses its output to decide whether to call a stronger model (Chen et al., 2024; Dekoninck et al., 2025). Routers learn which model to use from preference data, as in RouteLLM (Ong et al., 2025), benchmark results (Shnitzer et al., 2024; Mohammadshahi et al., 2024), or reward-model scores (Lu et al., 2024). Some predict query difficulty to guide model selection (Ding et al., 2024). RouterBench provides a common framework for evaluating routing methods (Hu et al., 2024). In cascades, routing rules often escalate to a larger model when the first model has low confidence or gives inconsistent answers (Varshney & Baral, 2022; Yue et al., 2024; Chuang et al., 2025). Learned routing rules use token-level uncertainty and model embeddings (Gupta et al., 2024) or verify the first model's answer before deciding whether to escalate (Aggarwal et al., 2024). Wang et al. (2024a) also improve cascades by training the first model to account for deferral. Dekoninck et al. (2025) derive an optimal cascading policy and prove that an existing routing policy is optimal; both depend on good estimates of answer quality. Jitkrittum et al. (2023) show when confidence alone gives an optimal deferral rule and when a learned rule (Narasimhan et al., 2022) is needed. We bound how many more of the first model's errors a learned router can catch than confidence routing under the same escalation rate.

Confidence and selective prediction. Language models can estimate whether their answers are correct (Kadavath et al., 2022) through stated confidence (Tian et al., 2023; Xiong et al., 2024) or agreement among sampled answers (Wang et al., 2023; Kuhn et al., 2023). Learned verifiers score candidate solutions to select better answers (Cobbe et al., 2021; Lightman et al., 2024), and the gap between generation and verification abilities is linked to self-improvement (Song et al., 2025). Confidence also guides decisions about when to answer. Selective prediction uses it to decide when to abstain (Chow, 1970; Geifman & El-Yaniv, 2017), while learning to defer also considers whether another expert will be correct (Mozannar & Sontag, 2020). For selective prediction, both calibration and how confidence ranks inputs matter (Rabanser & Papernot, 2025). Inputs with the same confidence can still differ in their probability of correctness; grouping loss measures this variation (Kull & Flach, 2015; Perez-Lebel et al., 2023). Melo et al. (2026) estimate grouping loss and excess decision risk to guide LLM cascades, and find that grouping loss decreases with model scale and instruction tuning. Our experiments examine how gains over confidence vary with answer-loss suboptimality across models of similar size and before and after task-specific finetuning.

Calibration and loss minimization. Language models can produce informative but poorly calibrated risk scores (Cruz et al., 2024). Post-hoc calibration adjusts confidence to match observed correctness (Platt, 2000; Guo et al., 2017; Kumar et al., 2019). Multicalibration extends this requirement to identifiable groups (Hebert-Johnson et al., 2018), including those defined by LLM embeddings and self-annotations (Detommaso et al., 2024). Loss minimization and multicalibration theory show how correcting predictable errors in probability estimates can reduce loss, including squared loss and cross-entropy (Gopalan et al., 2022; 2023; Błasiok et al., 2024). We show that any reduction in correctness log loss over self-confidence yields an equal reduction in answer log loss. Under our assumptions for combining models and counting parameters, this gain is bounded by the original model's suboptimality plus the improvement possible with the added parameters.

3 Predictable Performance is Suboptimal

We show that gains over self-confidence transfer exactly to lower answer loss and are bounded by model suboptimality and added predictor capacity. Proofs are deferred to Appendix A.1.

3.1 Problem Definition

Let (X, Y) be drawn from a fixed population, where $X \in \mathcal{X}$ is the query and $Y \in \mathcal{Y} = \{0, \ldots, K-1\}$ is the answer. A *model* $f \in \mathcal{F}$ is defined as a conditional probability mass function $f(y|x)$. Its point prediction is $\hat{f}(x) = \arg\max_y f(y \mid x)$, with ties broken by a fixed rule. We score any model f by its *answer loss* $L_Y(f) = \mathbb{E}[-\log f(Y \mid X)]$.

We define the correctness indicator $Z = \mathbb{1}\{Y = \hat{f}(X)\} \in \{0, 1\}$. A *performance predictor* $g \in \mathcal{G}$ is defined as a conditional probability mass function $g(z \mid x)$ that may read f's output on x. We require full support, $g(z \mid x) > 0$, and score g by its *correctness loss* $L_Z(g, f) = \mathbb{E}[-\log g(Z \mid X)]$.

The model's self-confidence provides a natural baseline: any model f induces a performance predictor f_Z with $f_Z(1 \mid x) = f(\hat{f}(x) \mid x)$. We abbreviate $L_Z(f) \coloneqq L_Z(f_Z, f)$.

The goal of performance prediction is to minimize $L_Z(g, f)$, where $L_Z(f)$ serves as a baseline. We call $L_Z(f) - L_Z(g, f)$ the *excess predictability* achieved by g relative to f's self-confidence; a performance predictor is *effective* when its excess predictability is positive.

3.2 Any Effective Performance Predictor Improves the Model

We use g to change how much probability f assigns to its chosen answer, while preserving the relative probabilities of the other answers.

Assuming $f_Z(0 \mid x) > 0$, define the alternative distribution and the *recalibrated model* by

$$f_{\mathrm{Alt}}(y \mid x) = f(y \mid x)/f_Z(0 \mid x), \qquad y \neq \hat{f}(x), \tag{1}$$

$$\mathcal{C}_g(f)(y \mid x) = \begin{cases} g(1 \mid x) & y = \hat{f}(x), \\ g(0 \mid x) f_{\mathrm{Alt}}(y \mid x) & y \neq \hat{f}(x). \end{cases} \tag{2}$$

Thus, $\mathcal{C}_g(f)$ assigns mass $g(1 \mid x)$ to $\hat{f}(x)$ and distributes the remaining mass among the other answers in their original proportions. Since only the correctness probabilities change, the following proposition transfers the predictor's improvement exactly to the answer loss.

Proposition 1 (Exact transfer). *For any f with $L_Y(f) < \infty$ and any performance predictor g,*

$$\boxed{L_Y\big(\mathcal{C}_g(f)\big) = L_Y(f) - \big(L_Z(f) - L_Z(g, f)\big).} \tag{3}$$

Recalibrating with an effective g strictly lowers the answer loss by exactly its excess predictability over f's self-confidence, $L_Z(f) - L_Z(g, f) > 0$. Recalibration is also hard to beat: any model with lower answer loss than $\mathcal{C}_g(f)$ must predict f's correctness better than g, or predict better among the alternative answers when f is wrong (Proposition 2 in Appendix A.2).

3.3 Predictable Performance is Suboptimal

For parameter budgets $P, Q \geq 0$, let $\mathcal{F}_P \subseteq \mathcal{F}$ denote the models with at most P parameters, and let $\mathcal{G}_Q \subseteq \mathcal{G}$ denote the performance predictors with at most Q parameters. Both sets are therefore nested in their respective budgets.

We allow models in $\mathcal{F}$ to use performance predictors and routers as components, and apply the same convention to predictors in $\mathcal{G}$. We count all parameters, including frozen weights, but count shared parameters only once. The fixed arithmetic and decision operations used below add no parameters. Thus, recalibrating $f \in \mathcal{F}_P$ with $g \in \mathcal{G}_Q$ gives a model $\mathcal{C}_g(f) \in \mathcal{F}_{P+Q}$. The loss floor and suboptimality statements below apply to the full set $\mathcal{F}$ on the fixed task distribution of (X, Y).

Empirical neural scaling laws suggest that the best loss achievable follows a predictable frontier as model size increases (Kaplan et al., 2020; Hoffmann et al., 2022). We idealize this as follows.

Assumption 1 (Scaling floor). *For each budget P, some model $f_P^* \in \mathcal{F}_P$ achieves the lowest loss $L_Y^*(P)$ in $\mathcal{F}_P$:*

$$L_Y(f) \geq L_Y^*(P) = L_Y(f_P^*) \qquad \textit{for every } f \in \mathcal{F}_P. \tag{4}$$

The floor $L_Y^*(P)$ is non-increasing in P, since a larger budget includes all models available at a smaller budget. Relative to this floor, define the *suboptimality at budget* P of $f \in \mathcal{F}_P$ as

$$\delta_P(f) \coloneqq L_Y(f) - L_Y^*(P) \geq 0. \tag{5}$$

The experiments use the gap to the best observed loss in a cohort of similar-size models as a proxy for this suboptimality (Section 3.4.2).

The recalibrated model uses at most $P + Q$ parameters, so its loss cannot fall below $L_Y^*(P + Q)$. Together with exact transfer, this limits how much g can improve on f's self-confidence.

Theorem 1 (Excess predictability bounded by suboptimality and added capacity)**.** *Under Assumption 1, every $f \in \mathcal{F}_P$ with $L_Y(f) < \infty$ and every $g \in \mathcal{G}_Q$ satisfy*

$$\boxed{L_Z(f) - L_Z(g, f) \leq \left[L_Y^*(P) - L_Y^*(P + Q)\right] + \delta_P(f)}. \tag{6}$$

Equation 6 separates two sources of excess predictability. For the loss-optimal model f_P^*, a predictor can recover at most the *scaling gain* $L_Y^*(P) - L_Y^*(P+Q)$, the decrease in the loss floor from adding Q parameters. A model farther from the floor increases this recoverable excess predictability by at most its suboptimality $\delta_P(f)$. Any improvement over f's self-confidence that exceeds the scaling gain therefore certifies $\delta_P(f) > 0$. In particular, if the performance predictor adds no parameters, then strictly positive excess predictability implies $\delta_P(f) > 0$.

Additional results. First, under diminishing returns, a predictor much smaller than the model has excess predictability bounded by the model's suboptimality plus a small capacity correction (Corollary 1, Appendix A.3). Second, suboptimality does not determine excess predictability: lowering answer loss does not necessarily lower correctness loss. Models can have the same maximum excess predictability yet differ in suboptimality because they assign different probabilities to the answers they do not choose (Proposition 3, Appendix A.4).

3.4 Experiments

Motivated by the preceding bound on excess predictability, we study two empirical questions on 16 language models across four benchmarks. Do more suboptimal models exhibit greater excess predictability over self-confidence? Does task-specific finetuning reduce this advantage?

3.4.1 Setup

Models and tasks. We evaluate 16 base and instruction-tuned models from eight families, with 7 to 12B parameters. We select four tasks, `WinoGrande` (Sakaguchi et al., 2020), `CommonsenseQA` (Talmor et al., 2019), `GSM8K` (Cobbe et al., 2021) and `MMLU-Pro` (Wang et al., 2024b). `WinoGrande` and `CommonsenseQA` are evaluated by scoring answer options, and `GSM8K` and `MMLU-Pro` are evaluated by generating answers. The binary task `WinoGrande` is a sanity check: predicting a fixed model's correctness is equivalent to predicting the answer. Appendix B.1 lists model releases and data splits, shared across models.

Evaluation. We use five-shot evaluation with `lm-eval-harness` (Gao et al., 2024). The answer distribution $f(\cdot \mid x)$ comes from normalized option likelihoods on the first two tasks and answer frequencies from 128 sampled solutions per question on the generation tasks. We select the highest-probability answer $\hat{f}$ and use its probability as self-confidence. On test questions, we measure answer log loss $L_Y(f)$ and correctness log loss $L_Z(f)$; cascades are evaluated by accuracy. Appendix B.2 details scoring, sampling, answer extraction, and probability estimation.

Finetuning. We use LoRA (Hu et al., 2022) to finetune on gold answers for the option-scoring tasks and on reasoning traces by Qwen3.5-122B-A10B for the generation tasks, selecting checkpoints on development performance. Our experiment design is built upon the idea that task-specific finetuning reduces suboptimality. Therefore, for each task, we consider all models whose development accuracy does not improve after finetuning to be anomalous and remove the corresponding model pairs from the analysis. This leaves 16, 16, 14, and 13 model pairs for the respective tasks. Appendix B.3 gives training and retention details.

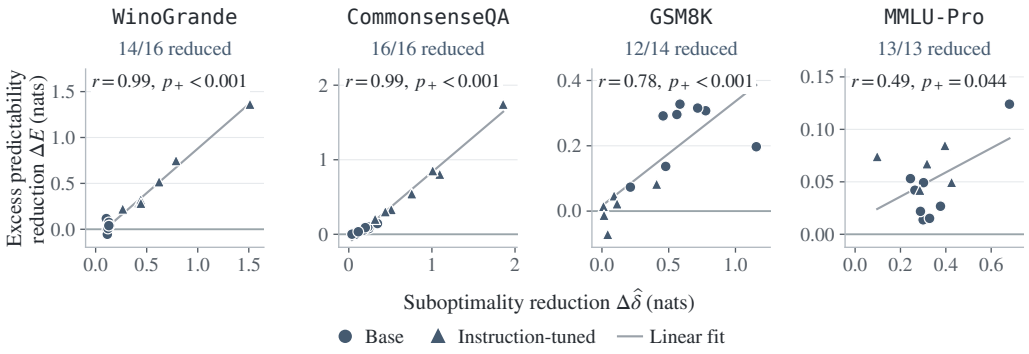

Figure 2: **Finetuning reduces excess predictability in 55 of 59 model-task pairs.** Each point compares one released model with its selected finetuned version; both axes show released minus finetuned values. $\Delta\widehat{\delta}$ is the reduction in answer log loss, and ΔE is the reduction in excess predictability. Points above zero have lower excess predictability after finetuning; counts report these pairs. Suboptimality decreases in all 59 pairs. Shapes identify the release type; lines are least-squares fits; r is Pearson correlation and p_+ its one-sided p-value for positive correlation.

Suboptimality. For each task, let $\mathcal{C}$ contain the released and finetuned versions of the retained models. We use their best observed answer loss as a proxy for the loss floor at similar model sizes:

$$\widehat{\delta}(f) = L_Y(f) - \min_{h \in \mathcal{C}} L_Y(h). \tag{7}$$

This reference is fixed for all models, so changes in empirical suboptimality equal changes in answer loss; changing the reference shifts all gaps equally and leaves correlations unchanged.

Performance predictor. We train a separate QWEN3-EMBEDDING-4B predictor (Zhang et al., 2025b) for each model, task, and finetuning stage. Its input contains the question, the model's final answer, option probabilities or sampled-answer frequencies, and, for generation tasks, a sampled reasoning trace ending in the majority answer. A linear head on the embedding corrects the self-confidence logit; we train the head and LoRA adapters from a Platt-calibrated initialization (Platt, 2000). We fit predictors on 85% of the development set and select the checkpoint and seed with lowest correctness loss on the remaining 15%. On the test set, excess predictability is

$$E(f) = L_Z(f) - L_Z(g, f). \tag{8}$$

Appendix B.4 gives the architecture and training details.

3.4.2 RESULTS

Greater excess predictability comes with greater suboptimality. Figure 1 shows a positive correlation between empirical suboptimality and excess predictability across all four tasks. We plot released and finetuned models together because the relationship concerns a model's suboptimality, regardless of how it was trained. Appendix C.2 shows the two groups separately; the association also holds among released models alone. The association is stronger on the simpler tasks, `WinoGrande` and `CommonsenseQA`, with Pearson correlations of 0.98 for both; on `WinoGrande` this is the expected sanity-check outcome (Section 3.4.1). On the generation-based tasks, `GSM8K` and `MMLU-Pro`, the correlations are 0.93 and 0.65, respectively. The weaker association on `MMLU-Pro` may partly reflect variation in alternative loss: models can differ in answer loss without differing in excess predictability (Appendix A.4).

Finetuning reduces excess predictability. Excess predictability falls in 55 of the 59 retained model-task pairs after finetuning (Figure 2): 14 of 16 on `WinoGrande`, all 16 on `CommonsenseQA`, 12 of 14 on `GSM8K`, and all 13 on `MMLU-Pro`. The four exceptions have relatively small reductions in suboptimality, indicating that finetuning improves their answer loss only modestly. Note that we train a new predictor for each finetuned model, yet its gain over self-confidence still shrinks. Larger reductions in suboptimality tend to accompany larger reductions in excess predictability, with Pearson correlations of 0.99, 0.99, 0.78, and 0.49 in task order.

During finetuning of LLAMA-3.1-8B base on `CommonsenseQA`, self-confidence catches up with separately retrained predictors: excess predictability falls from 0.107 to 0.004 nats (Figure 3). Appendix B.3 gives the trajectory setup.

Predictor capacity. We further compare 0.6B, 4B, and 8B predictors on the same models (Appendix C.3). The 0.6B predictor adds little beyond Platt calibration, a simple adjustment of the model's own confidence, whereas larger predictors generally achieve lower correctness log loss. Our main finding holds at all three predictor sizes: finetuning reduces mean excess predictability on every task. For the 4B predictor, gains over Platt-calibrated self-confidence also decrease after finetuning in 50 of 59 pairs (Appendix Table 8).

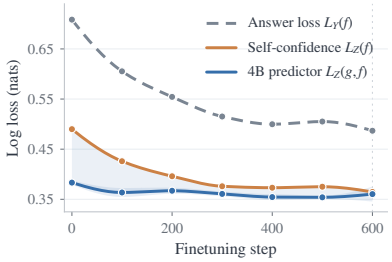

Figure 3: Answer and correctness losses during finetuning of LLAMA-3.1-8B on `CommonsenseQA`. Shading shows excess predictability $E(f)$; predictor loss averages three seeds.

4 SUBOPTIMALITY BOUNDS CASCADING

In a cascade, a router decides which queries to send from a weak model to a stronger model. We compare a learned router with the weak model's self-confidence at the same escalation rate. Cascades are judged by accuracy at that rate, so this section bounds error rates rather than log loss. The key connection is that a router that finds more weak-model errors also supplies a better performance predictor. Proofs are deferred to Appendices A.5–A.7.

4.1 CASCADING SETUP

Let $f_1 \in \mathcal{F}_{P_1}$ and $f_2 \in \mathcal{F}_{P_2}$ be the weak and strong models, with $P_1 < P_2$ and $L_Y(f_1) < \infty$. For $i = 1, 2$, write $Z_i = \mathbb{1}\{Y = \hat{f}_i(X)\}$ and $\epsilon_i(x) = \Pr(Z_i = 0 \mid X = x)$ for correctness and the actual error probability. The weak model estimates its error probability by $f_{1,Z}(0 \mid x) = 1 - f_1(\hat{f}_1(x) \mid x)$, assumed strictly between zero and one almost surely.

A budget-Q router assigns a score $r : \mathcal{X} \to \mathbb{R}$ using at most Q additional parameters beyond f_1. At escalation rate $\rho \in [0, 1]$, we escalate the highest-scoring ρ fraction of queries to f_2, breaking ties uniformly at random. Its *routing decision* $B_{r,\rho}(x)$ is one for escalation and zero for keeping f_1's answer. Self-confidence routing applies the same rule to $f_{1,Z}(0 \mid x)$, giving $B_{f,\rho}$. At a fixed rate, abbreviate these decisions as B_r, B_f, with $\mathbb{E}[B_r] = \mathbb{E}[B_f] = \rho$, and set $u = B_r - B_f$.

Writing $\mathrm{Err}(f)$ for a model's error rate, the difference between the two cascades decomposes as

$$\mathrm{Err}(f_{\mathrm{cas},B_r}) - \mathrm{Err}(f_{\mathrm{cas},B_f}) = -\underbrace{\mathbb{E}[u\epsilon_1(X)]}_{\text{additional weak errors captured}} + \underbrace{\mathbb{E}[u\epsilon_2(X)]}_{\text{additional strong errors encountered}}. \tag{9}$$

A router can improve on confidence by capturing more weak-model errors, encountering fewer strong-model errors, or both. We first bound the weak-error contribution, then state when this also bounds the cascade advantage.

4.2 FROM PERFORMANCE PREDICTION TO CASCADE BOUNDS

At a fixed escalation rate, B_f escalates the lowest-confidence ρ fraction of queries and therefore maximizes the predicted weak errors captured, $\mathbb{E}[B_f f_{1,Z}(0 \mid X)]$. Thus $\mathbb{E}[u\, f_{1,Z}(0 \mid X)] \le 0$. A router that captures more actual errors must therefore identify where self-confidence misestimates their probability:

$$\begin{aligned}\mathbb{E}[u\epsilon_1(X)] &= \mathbb{E}[u\, f_{1,Z}(0 \mid X)] + \mathbb{E}[u(\epsilon_1(X) - f_{1,Z}(0 \mid X))] \\ &\le \mathbb{E}[u(\epsilon_1(X) - f_{1,Z}(0 \mid X))].\end{aligned} \tag{10}$$

We can construct a performance predictor g_λ from the routing score r by shifting the weak model's confidence where its routing decisions differ from self-confidence routing:

$$\operatorname{logit} g_\lambda(0 \mid x) = \operatorname{logit} f_{1,Z}(0 \mid x) + \lambda u(x), \tag{11}$$

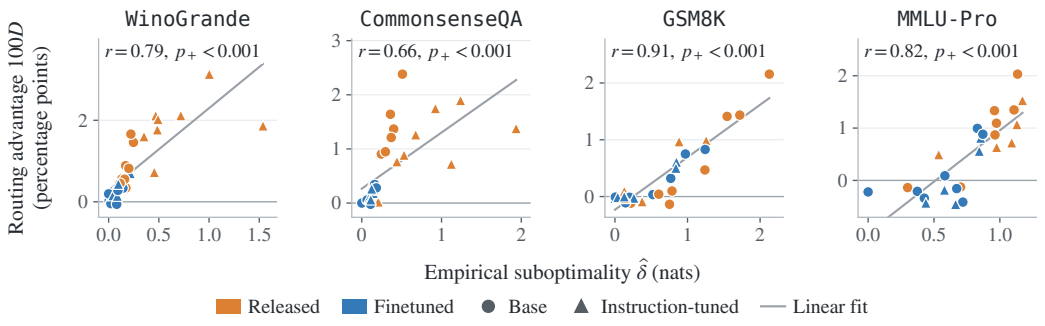

Figure 4: **More suboptimal models have larger routing advantages.** Each point is one released or finetuned weak-model version. Empirical suboptimality is the gap to the best observed answer log loss on each task. Routing advantage $100D$ is the cascade accuracy gain from a 4B predictor over self-confidence routing to QWEN3.5-122B-A10B, averaged over equal escalation rates and expressed in percentage points. Colors identify training stage; shapes identify the release type. Lines are least-squares fits; r is Pearson correlation and p_+ its one-sided p-value for positive correlation.

where $\mathrm{logit}(a) = \log(a/(1-a))$ and $g_\lambda(1 \mid x) = 1 - g_\lambda(0 \mid x)$. Lemma 1 in Appendix A.5 shows that, if $\mathbb{E}[u\epsilon_1(X)] > 0$, then there exists $\lambda > 0$ such that $L_Z(g_\lambda, f_1) < L_Z(f_1)$. In other words, a router that captures additional weak-model errors can be used to construct an effective predictor.

Recalibration with g_λ (see Equation 2) reuses f_1 and adds the budget-Q routing score r, the shared escalation rate ρ, and the correction strength λ. Therefore, the resulting model has at most P_1+Q+2 parameters. Applying Theorem 1 to f_1 and g_λ bounds the excess predictability. Combining this upper bound with Lemma 1 bounds the additional weak errors captured.

Theorem 2 (Suboptimality and capacity bound weak-error capture beyond confidence)**.** *Under Assumption 1, every budget-Q routing score r and escalation rate ρ induce decisions B_r and B_f whose difference $u = B_r - B_f$ satisfies*

$$\boxed{\mathbb{E}[u\epsilon_1(X)] \le \sqrt{\min\{\rho, 1-\rho\}\,[\delta_{P_1}(f_1) + L^*_Y(P_1) - L^*_Y(P_1+Q+2)]}.} \tag{12}$$

At fixed budgets and loss floor, reducing weak-model suboptimality tightens the bound on additional errors captured. By Equation 9, the same bound applies to cascade accuracy gains when $\mathbb{E}[u\epsilon_2(X)] \ge 0$ (Appendix A.7). Empirically, $\mathbb{E}[u\epsilon_2(X)]$ is nonnegative on average over escalation rates for most evaluated models (Appendix C.5).

4.3 Experiments

The cascade experiments test a distinct question from predictability: whether predictors rank questions better for escalation and thereby improve cascade accuracy. For example, Platt calibration with positive slope can reduce correctness log loss relative to raw self-confidence while preserving rankings. At fixed escalation rates, it therefore leaves routing decisions unchanged and provides no cascade advantage.

4.3.1 Setup

Following Section 4, the weak model f_1 answers each question first, and the routing decision determines whether to keep its answer or escalate to the strong model f_2.

Each model serves as f_1, both before and after finetuning. We use QWEN3.5-122B-A10B as f_2. On `WinoGrande` and `CommonsenseQA`, we finetune f_2 and select the checkpoint with the lowest development answer loss. On `GSM8K` and `MMLU-Pro`, we use its released version, which is already more accurate than all weak models.

We compare the selected 4B predictor g from Section 3.4.1 with the weak model's self-confidence as sources of routing scores, using error scores $g(0 \mid x)$ and $f_{1,Z}(0 \mid x)$. The thresholding rule from Section 4 gives decisions $B_{g,\rho}$ and $B_{f,\rho}$ at the same escalation rate ρ.

The *routing advantage* D measures how much predictor routing improves cascade accuracy over self-confidence routing, averaged over all escalation rates:

$$D = \int_0^1 \left[\mathrm{Err}(f_{\mathrm{cas},B_{f,\rho}}) - \mathrm{Err}(f_{\mathrm{cas},B_{g,\rho}})\right] d\rho. \tag{13}$$

We report $100D$ as the average accuracy gain in percentage points.

4.3.2 Results

Cascade accuracy across escalation rates. For Llama-3.1-8B base on `CommonsenseQA`, finetuning improves weak-model accuracy and brings self-confidence routing closer to predictor routing (Figure 5). Appendix C.4 shows the curves for all four tasks.

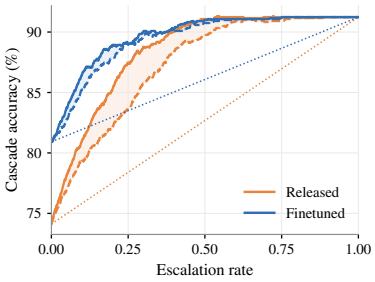

Figure 5: Llama-3.1-8B cascades on `CommonsenseQA` before and after finetuning: predictor (solid), confidence (dashed), and random (dotted) routing. Shading shows routing advantage D.

More suboptimal models have larger routing advantages. Figure 4 shows that more suboptimal models have larger routing advantages. Pearson correlations between empirical suboptimality and routing advantage range from 0.66 to 0.91 across the four tasks. Appendix C.2 shows released and finetuned models separately. Compared with excess predictability, routing advantage has a weaker correlation with suboptimality on three of the four tasks. One possible reason is that finding more weak-model errors does not always improve cascade accuracy: the strong model may also answer these questions incorrectly. Appendix C.5 separates the routing advantage into additional weak-model errors captured and additional strong-model errors encountered (see Equation 9). The captured errors alone are at least as strongly associated with suboptimality, with Pearson correlations of 0.80, 0.72, 0.91, and 0.86 in task order.

Finetuning reduces routing advantages. Similar to excess predictability, after finetuning, the routing advantage falls in 53 of the 59 retained model-task pairs (Appendix Figure 14). Larger reductions in suboptimality tend to accompany larger reductions in routing advantage, with Pearson correlations of 0.40 to 0.73. Averaged over the retained models, the routing advantage falls from 1.38 to 0.18 percentage points on `WinoGrande`, 1.09 to 0.18 on `CommonsenseQA`, 0.52 to 0.20 on `GSM8K`, and 0.94 to 0.07 on `MMLU-Pro` (Table 9).

5 Conclusion

The value of a separate performance predictor depends on the model it evaluates. We show that any excess predictability over self-confidence can be turned into an equal reduction in the model's answer loss. The predictor's advantage is therefore limited by the model's suboptimality plus the capacity the predictor adds, and the same limit governs how many more weak-model errors a router can catch than confidence routing. In our experiments, more suboptimal models have larger prediction and routing advantages, and finetuning lets self-confidence catch up on both. Improving the model can thus reduce the need for a separate predictor.

When self-confidence is unavailable, our analysis suggests three sources of predictive value: recovering self-confidence, exploiting suboptimality, and adding capacity. Recovering self-confidence alone can make a separate predictor useful, even when little room remains to surpass it.

Limitations and further directions. The theory gives an upper bound, so it does not imply that every reduction in answer loss will reduce a predictor's advantage. Our cascade bound controls additional weak-model errors captured; bounding the full routing advantage also requires a condition on strong-model errors. Future work should characterize the strong model's contribution and how to predict the benefit of escalation without running it.

STATEMENTS

AI USE STATEMENT

We used generative AI tools to assist with mathematical proofs, writing experimental code, running experiments, and to generate synthetic training data. Specifically, we used QWEN3.5-122B-A10B to generate reasoning traces for finetuning on `GSM8K` and `MMLU-Pro`, retaining only traces whose extracted final answers matched the gold answers (Appendix B.3). Additionally, we used generative AI tools to search for and identify related literature and to edit manuscript text for readability.

The authors formulated the hypotheses, developed the research methodology, designed the experiments, and interpreted the results without generative AI assistance. We did not use generative AI for translation or dataset cleaning; qualitative and thematic data analysis are not applicable to this work. We have reviewed all AI-assisted work. The authors take responsibility for the final content of this work, including text, mathematical claims, code, synthetic training data, and experimental results produced with the assistance of generative AI.

ETHICS STATEMENT

This work provides theoretical analysis and experiments on performance prediction and model routing using existing language models and benchmark datasets. The models and datasets may reflect existing biases, and our analyses of predictive performance and inference cost do not assess fairness, privacy, or safety in deployment. We do not identify additional ethical concerns specific to this study.

REPRODUCIBILITY STATEMENT

The theoretical assumptions are stated in Sections 3 and 4, and proofs are provided in Appendix A. Section 3.4.1 and Appendix B document the model releases, datasets and data splits, answer scoring, synthetic training-data generation, finetuning and predictor-training settings, model selection, and cascade evaluation procedures. Appendix C.1 describes the correlation estimates and uncertainty analysis.

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, pp. 131000–131034, 2024. doi: 10.52202/079017-4164. URL https://proceedings.neurips.cc/paper_files/paper/2024/hash/ecda225cb187b40ea8edc1f46b03ffda-Abstract-Conference.html.

Jarosław Błasiok, Parikshit Gopalan, Lunjia Hu, Adam Tauman Kalai, and Preetum Nakkiran. Loss minimization yields multicalibration for large neural networks. In *15th Innovations in Theoretical Computer Science Conference (ITCS 2024)*, volume 287, pp. 17:1–17:21, 2024. doi: 10.4230/LIPIcs.ITCS.2024.17. URL https://drops.dagstuhl.de/entities/document/10.4230/LIPIcs.ITCS.2024.17.

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. ISSN 2835-8856. 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. doi: 10.1109/TIT.1970.1054406.

Yu-Neng Chuang, Leisheng Yu, Guanchu Wang, Lizhe Zhang, Zirui Liu, Xuanting Cai, Yang Sui, Vladimir Braverman, and Xia Hu. Confident or seek stronger: Exploring uncertainty-based on-device LLM routing from benchmarking to generalization. *arXiv preprint arXiv:2502.04428*, 2025. URL https://arxiv.org/abs/2502.04428.

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

André F. Cruz, Moritz Hardt, and Celestine Mendler-Dünner. Evaluating language models as risk scores. In *Advances in Neural Information Processing Systems*, volume 37, 2024. doi: 10.52202/079017-3089. URL https://proceedings.neurips.cc/paper_files/paper/2024/hash/b0a4b3e384b4554e65a47ad1f6b0310a-Abstract-Datasets_and_Benchmarks_Track.html.

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*, volume 267, pp. 12987–13010, 2025. URL https://proceedings.mlr.press/v267/dekoninck25a.html.

Gianluca Detommaso, Martin Andres Bertran, Riccardo Fogliato, and Aaron Roth. Multicalibration for confidence scoring in LLMs. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pp. 10624–10641. PMLR, 2024. URL https://proceedings.mlr.press/v235/detommaso24a.html.

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 *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=02f3mUtqnM.

Falcon-LLM Team. The Falcon 3 family of open models, December 2024. URL https://huggingface.co/blog/falcon3.

Leo Gao, Jonathan Tow, Baber Abbasi, Stella Biderman, Sid Black, Anthony DiPofi, Charles Foster, Laurence Golding, Jeffrey Hsu, Alain Le Noac'h, Haonan Li, Kyle McDonell, Niklas Muennighoff, Chris Ociepa, Jason Phang, Laria Reynolds, Hailey Schoelkopf, Aviya Skowron, Lintang Sutawika, Eric Tang, Anish Thite, Ben Wang, Kevin Wang, and Andy Zou. The language model evaluation harness, 2024. URL https://zenodo.org/records/12608602.

Yonatan Geifman and Ran El-Yaniv. Selective classification for deep neural networks. In *Advances in Neural Information Processing Systems*, volume 30, pp. 4878–4887, 2017. URL https://proceedings.neurips.cc/paper/2017/hash/4a8423d5e91fda00bb7e46540e2b0cf1-Abstract.html.

Gemma Team. Gemma 3 technical report. *arXiv preprint arXiv:2503.19786*, 2025. URL https://arxiv.org/abs/2503.19786.

Parikshit Gopalan, Adam Tauman Kalai, Omer Reingold, Vatsal Sharan, and Udi Wieder. Omnipredictors. In *13th Innovations in Theoretical Computer Science Conference (ITCS 2022)*, volume 215, pp. 79:1–79:21, 2022. doi: 10.4230/LIPIcs.ITCS.2022.79. URL https://drops.dagstuhl.de/entities/document/10.4230/LIPIcs.ITCS.2022.79.

Parikshit Gopalan, Lunjia Hu, Michael P. Kim, Omer Reingold, and Udi Wieder. Loss minimization through the lens of outcome indistinguishability. In *14th Innovations in Theoretical Computer Science Conference (ITCS 2023)*, volume 251, pp. 60:1–60:20, 2023. doi: 10.4230/LIPIcs.ITCS.2023.60. URL https://drops.dagstuhl.de/entities/document/10.4230/LIPIcs.ITCS.2023.60.

Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, et al. The Llama 3 herd of models. *arXiv preprint arXiv:2407.21783*, 2024. URL https://arxiv.org/abs/2407.21783.

Chuan Guo, Geoff Pleiss, Yu Sun, and Kilian Q. Weinberger. On calibration of modern neural networks. In *Proceedings of the 34th International Conference on Machine Learning*, volume 70 of *Proceedings of Machine Learning Research*, pp. 1321–1330. PMLR, 2017. URL https://proceedings.mlr.press/v70/guo17a.html.

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

Ursula Hebert-Johnson, Michael Kim, Omer Reingold, and Guy Rothblum. Multicalibration: Calibration for the (computationally-identifiable) masses. In *Proceedings of the 35th International Conference on Machine Learning*, volume 80, pp. 1939–1948, 2018. URL https://proceedings.mlr.press/v80/hebert-johnson18a.html.

Alejandro Hernández-Cano, Alexander Hägele, Allen Hao Huang, et al. Apertus: Democratizing open and compliant LLMs for global language environments. *arXiv preprint arXiv:2509.14233*, 2025. URL https://arxiv.org/abs/2509.14233.

José Hernández-Orallo, Wout Schellaert, and Fernando Martínez-Plumed. Training on the test set: Mapping the system-problem space in AI. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 36, pp. 12256–12261, 2022. doi: 10.1609/aaai.v36i11.21487. URL https://ojs.aaai.org/index.php/AAAI/article/view/21487.

Jordan Hoffmann, Sebastian Borgeaud, Arthur Mensch, Elena Buchatskaya, Trevor Cai, Eliza Rutherford, Diego de Las Casas, Lisa Anne Hendricks, Johannes Welbl, Aidan Clark, Tom Hennigan, Eric Noland, Katie Millican, George van den Driessche, Bogdan Damoc, Aurelia Guy, Simon Osindero, Karen Simonyan, Erich Elsen, Jack W. Rae, Oriol Vinyals, and Laurent Sifre. Training compute-optimal large language models. *arXiv preprint arXiv:2203.15556*, 2022. URL https://arxiv.org/abs/2203.15556.

Chi-Yang Hsu, Alexander Braylan, Yiheng Su, Matthew Lease, and Omar Alonso. PIE: Performance interval estimation for free-form generation tasks. *arXiv preprint arXiv:2509.07309*, 2025. doi: 10.48550/arXiv.2509.07309. URL https://arxiv.org/abs/2509.07309v2. Version 2, January 2026.

Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. LoRA: Low-rank adaptation of large language models. In *International Conference on Learning Representations*, 2022. URL https://openreview.net/forum?id=nZeVKeeFYf9.

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. doi: 10.48550/arXiv.2403.12031. URL https://arxiv.org/abs/2403.12031.

Albert Q. Jiang, Alexandre Sablayrolles, Arthur Mensch, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Florian Bressand, Gianna Lengyel, Guillaume Lample, Lucile Saulnier, Lélio Renard Lavaud, Marie-Anne Lachaux, Pierre Stock, Teven Le Scao, Thibaut Lavril, Thomas Wang, Timothée Lacroix, and William El Sayed. Mistral 7B. *arXiv preprint arXiv:2310.06825*, 2023. URL https://arxiv.org/abs/2310.06825.

Wittawat Jitkrittum, Neha Gupta, Aditya K. Menon, Harikrishna Narasimhan, Ankit Rawat, and Sanjiv Kumar. When does confidence-based cascade deferral suffice? In *Advances in Neural Information Processing Systems*, volume 36, pp. 9891–9906, 2023. doi: 10.52202/075280-0431. URL https://proceedings.neurips.cc/paper_files/paper/2023/hash/1f09e1ee5035a4c3fe38a5681cae5815-Abstract-Conference.html.

Saurav Kadavath, Tom Conerly, Amanda Askell, Tom Henighan, Dawn Drain, Ethan Perez, Nicholas Schiefer, Zac Hatfield-Dodds, Nova DasSarma, Eli Tran-Johnson, et al. Language models (mostly) know what they know. *arXiv preprint arXiv:2207.05221*, 2022.

Jared Kaplan, Sam McCandlish, Tom Henighan, Tom B. Brown, Benjamin Chess, Rewon Child, Scott Gray, Alec Radford, Jeffrey Wu, and Dario Amodei. Scaling laws for neural language models. *arXiv preprint arXiv:2001.08361*, 2020. URL https://arxiv.org/abs/2001.08361.

Sanyam Kapoor, Nate Gruver, Manley Roberts, Katherine Collins, Arka Pal, Umang Bhatt, Adrian Weller, Samuel Dooley, Micah Goldblum, and Andrew Gordon Wilson. Large language models must be taught to know what they don't know. In *Advances in Neural Information Processing Systems*, volume 37, 2024. URL https://proceedings.neurips.cc/paper_files/paper/2024/hash/9c20f16b05f5e5e70fa07e2a4364b80e-Abstract-Conference.html.

Lorenz Kuhn, Yarin Gal, and Sebastian Farquhar. Semantic uncertainty: Linguistic invariances for uncertainty estimation in natural language generation. In *The Eleventh International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=VD-AYtP0dve.

Meelis Kull and Peter Flach. Novel decompositions of proper scoring rules for classification: Score adjustment as precursor to calibration. In *Machine Learning and Knowledge Discovery in Databases (ECML PKDD 2015)*, Lecture Notes in Computer Science, pp. 68–85. Springer, 2015. doi: 10.1007/978-3-319-23528-8_5.

Ananya Kumar, Percy Liang, and Tengyu Ma. Verified uncertainty calibration. In *Advances in Neural Information Processing Systems*, volume 32, 2019. URL https://proceedings.neurips.cc/paper/2019/hash/f8c0c968632845cd133308b1a494967f-Abstract.html.

Hunter Lightman, Vineet Kosaraju, Yuri Burda, Harrison Edwards, Bowen Baker, Teddy Lee, Jan Leike, John Schulman, Ilya Sutskever, and Karl Cobbe. Let's verify step by step. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=v8L0pN6EOi.

Keming Lu, Hongyi Yuan, Runji Lin, Junyang Lin, Zheng Yuan, Chang Zhou, and Jingren Zhou. Routing to the expert: Efficient reward-guided ensemble of large language models. In *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pp. 1964–1974. Association for Computational Linguistics, 2024. doi: 10.18653/v1/2024.naacl-long.109. URL https://aclanthology.org/2024.naacl-long.109/.

Felipe Maia Polo, Lucas Weber, Leshem Choshen, Yuekai Sun, Gongjun Xu, and Mikhail Yurochkin. tinyBenchmarks: Evaluating LLMs with fewer examples. In *Proceedings of the*

41st International Conference on Machine Learning, volume 235 of *Proceedings of Machine Learning Research*, pp. 34303–34326. PMLR, 2024. URL https://proceedings.mlr.press/v235/maia-polo24a.html.

Sébastien Melo, Gaël Varoquaux, and Marine Le Morvan. Epistemic uncertainty quantification to improve decisions from black-box models. In *The Fourteenth International Conference on Learning Representations*, 2026. URL https://openreview.net/forum?id=JfiwaTxhI8.

Alireza Mohammadshahi, Arshad Rafiq Shaikh, and Majid Yazdani. Routoo: Learning to route to large language models effectively. *arXiv preprint arXiv:2401.13979*, 2024. doi: 10.48550/arXiv.2401.13979. URL https://arxiv.org/abs/2401.13979v3. Version 3.

Iván Vicente Moreno Cencerrado, Arnau Padrés Masdemont, Anton Gonzalvez Hawthorne, David Demitri Africa, and Lorenzo Pacchiardi. No answer needed: Predicting LLM answer accuracy from question-only linear probes. In *ICLR 2026 Workshop on Principled Design for Trustworthy AI*, 2026. URL https://arxiv.org/abs/2509.10625.

Hussein Mozannar and David Sontag. Consistent estimators for learning to defer to an expert. In *Proceedings of the 37th International Conference on Machine Learning*, volume 119, pp. 7076–7087, 2020. URL https://proceedings.mlr.press/v119/mozannar20b.html.

Harikrishna Narasimhan, Wittawat Jitkrittum, Aditya K. Menon, Ankit Rawat, and Sanjiv Kumar. Post-hoc estimators for learning to defer to an expert. In *Advances in Neural Information Processing Systems*, volume 35, pp. 29292–29304, 2022. doi: 10.52202/068431-2124. URL https://proceedings.neurips.cc/paper_files/paper/2022/hash/bc8f76d9caadd48f77025b1c889d2e2d-Abstract-Conference.html.

Isaac Ong, Amjad Almahairi, Vincent Wu, Wei-Lin Chiang, Tianhao Wu, Joseph E. Gonzalez, M. Waleed Kadous, and Ion Stoica. RouteLLM: Learning to route LLMs from preference data. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=8sSqNntaMr.

David Owen. How predictable is language model benchmark performance? *arXiv preprint arXiv:2401.04757*, 2024. URL https://arxiv.org/abs/2401.04757.

Lorenzo Pacchiardi, Lucy G. Cheke, and José Hernández-Orallo. 100 instances is all you need: Predicting the success of a new LLM on unseen data by testing on a few instances. In *KDD 2024 Workshop on Evaluation and Trustworthiness of Generative AI Models*, 2024. URL https://arxiv.org/abs/2409.03563. arXiv:2409.03563.

Alexandre Perez-Lebel, Marine Le Morvan, and Gaël Varoquaux. Beyond calibration: Estimating the grouping loss of modern neural networks. In *The Eleventh International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=6w1k-IixnL8.

John C. Platt. Probabilistic outputs for support vector machines and comparisons to regularized likelihood methods. In Alexander J. Smola, Peter Bartlett, Bernhard Schölkopf, and Dale Schuurmans (eds.), *Advances in Large Margin Classifiers*. MIT Press, 2000.

Qwen Team. Qwen3.5: Towards native multimodal agents, February 2026. URL https://qwen.ai/blog?id=qwen3.5.

Stephan Rabanser and Nicolas Papernot. What does it take to build a performant selective classifier? In *Advances in Neural Information Processing Systems*, volume 38, 2025. URL https://openreview.net/forum?id=pcwI5cNFJK.

Yangjun Ruan, Chris J. Maddison, and Tatsunori Hashimoto. Observational scaling laws and the predictability of language model performance. In *Advances in Neural Information Processing Systems*, volume 37, 2024. URL https://proceedings.neurips.cc/paper_files/paper/2024/hash/1cded4f97cf5f01a284c574110b7e3b9-Abstract-Conference.html.

Keisuke Sakaguchi, Ronan Le Bras, Chandra Bhagavatula, and Yejin Choi. WinoGrande: An adversarial winograd schema challenge at scale. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pp. 8732–8740, 2020. doi: 10.1609/aaai.v34i05.6399. URL https://ojs.aaai.org/index.php/AAAI/article/view/6399.

Dylan Sam, Marc Finzi, and Zico Kolter. Predicting the performance of black-box language models with follow-up queries. In *Advances in Neural Information Processing Systems*, volume 38, pp. 176307–176342, 2025. doi: 10.52202/085713-5305. URL https://doi.org/10.52202/085713-5305.

Tal Shnitzer, Anthony Ou, Mírian Silva, Kate Soule, Yuekai Sun, Justin Solomon, Neil Thompson, and Mikhail Yurochkin. Large language model routing with benchmark datasets. In *First Conference on Language Modeling*, 2024. URL https://openreview.net/forum?id=Zb0ajZ7vAt.

Yuda Song, Hanlin Zhang, Carson Eisenach, Sham Kakade, Dean Foster, and Udaya Ghai. Mind the gap: Examining the self-improvement capabilities of large language models. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://arxiv.org/abs/2412.02674.

Sharath Turuvekere Sreenivas, Saurav Muralidharan, Raviraj Joshi, Marcin Chochowski, Ameya Sunil Mahabaleshwarkar, Gerald Shen, Jiaqi Zeng, Zijia Chen, Yoshi Suhara, Shizhe Diao, Chenhan Yu, Wei-Chun Chen, Hayley Ross, Oluwatobi Olabiyi, Ashwath Aithal, Oleksii Kuchaiev, Daniel Korzekwa, Pavlo Molchanov, Mostofa Patwary, Mohammad Shoeybi, Jan Kautz, and Bryan Catanzaro. LLM pruning and distillation in practice: The Minitron approach. *arXiv preprint arXiv:2408.11796*, 2024. URL https://arxiv.org/abs/2408.11796.

Alon Talmor, Jonathan Herzig, Nicholas Lourie, and Jonathan Berant. CommonsenseQA: A question answering challenge targeting commonsense knowledge. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pp. 4149–4158. Association for Computational Linguistics, 2019. doi: 10.18653/v1/N19-1421. URL https://aclanthology.org/N19-1421/.

Team Olmo, Allyson Ettinger, Amanda Bertsch, Bailey Kuehl, et al. Olmo 3. *arXiv preprint arXiv:2512.13961*, 2025. URL https://arxiv.org/abs/2512.13961.

Katherine Tian, Eric Mitchell, Allan Zhou, Archit Sharma, Rafael Rafailov, Huaxiu Yao, Chelsea Finn, and Christopher Manning. Just ask for calibration: Strategies for eliciting calibrated confidence scores from language models fine-tuned with human feedback. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pp. 5433–5442. Association for Computational Linguistics, 2023. doi: 10.18653/v1/2023.emnlp-main.330. URL https://aclanthology.org/2023.emnlp-main.330/.

Neeraj Varshney and Chitta Baral. Model cascading: Towards jointly improving efficiency and accuracy of NLP systems. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pp. 11007–11021. Association for Computational Linguistics, 2022. doi: 10.18653/v1/2022.emnlp-main.756. URL https://aclanthology.org/2022.emnlp-main.756/.

Congchao Wang, Sean Augenstein, Keith Rush, Wittawat Jitkrittum, Harikrishna Narasimhan, Ankit Singh Rawat, Aditya Krishna Menon, and Alec Go. Cascade-aware training of language models. *arXiv preprint arXiv:2406.00060*, 2024a. URL https://arxiv.org/abs/2406.00060.

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

Yubo Wang, Xueguang Ma, Ge Zhang, Yuansheng Ni, Abhranil Chandra, Shiguang Guo, Weiming Ren, Aaran Arulraj, Xuan He, Ziyan Jiang, Tianle Li, Max Ku, Kai Wang,

Alex Zhuang, Rongqi Fan, Xiang Yue, and Wenhu Chen. MMLU-Pro: A more robust and challenging multi-task language understanding benchmark. In *Advances in Neural Information Processing Systems*, volume 37, pp. 95266–95290, 2024b. doi: 10.52202/079017-3018. URL https://papers.nips.cc/paper_files/paper/2024/hash/ad236edc564f3e3156e1b2feafb99a24-Abstract-Datasets_and_Benchmarks_Track.html. Datasets and Benchmarks Track.

Hanqi Xiao, Vaidehi Patil, Hyunji Lee, Elias Stengel-Eskin, and Mohit Bansal. Generalized correctness models: Learning calibrated and model-agnostic correctness predictors from historical patterns. In *Proceedings of the 43rd International Conference on Machine Learning*, 2026. URL https://arxiv.org/abs/2509.24988.

Miao Xiong, Zhiyuan Hu, Xinyang Lu, Yifei Li, Jie Fu, Junxian He, and Bryan Hooi. Can LLMs express their uncertainty? an empirical evaluation of confidence elicitation in LLMs. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=gjeQKFxFpZ.

Yuekun Yao and Alexander Koller. Predicting generalization performance with correctness discriminators. In *Findings of the Association for Computational Linguistics: EMNLP 2024*, pp. 11725–11739. Association for Computational Linguistics, 2024. doi: 10.18653/v1/2024.findings-emnlp.686. URL https://aclanthology.org/2024.findings-emnlp.686/.

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 *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=6okaSfANzh.

Guanhua Zhang, Florian E. Dorner, and Moritz Hardt. How benchmark prediction from fewer data misses the mark. In *Advances in Neural Information Processing Systems*, volume 38, 2025a. doi: 10.52202/085713-4528. URL https://proceedings.nips.cc/paper_files/paper/2025/hash/c57bacb5ea621582235c54ce2a83136d-Abstract-Conference.html.

Yanzhao Zhang, Mingxin Li, Dingkun Long, Xin Zhang, Huan Lin, Baosong Yang, Pengjun Xie, An Yang, Dayiheng Liu, Junyang Lin, Fei Huang, and Jingren Zhou. Qwen3 Embedding: Advancing text embedding and reranking through foundation models. *arXiv preprint arXiv:2506.05176*, 2025b. URL https://arxiv.org/abs/2506.05176.

Lexin Zhou, Fernando Martínez-Plumed, José Hernández-Orallo, Cèsar Ferri, and Wout Schellaert. Reject before you run: Small assessors anticipate big language models. In *Proceedings of the Workshop on AI Evaluation Beyond Metrics (EBeM 2022)*, volume 3169 of *CEUR Workshop Proceedings*. CEUR-WS.org, 2022. URL https://ceur-ws.org/Vol-3169/paper4.pdf.

A Proofs and Additional Results

A.1 Proofs for Section 3

Proof of Proposition 1. On a correct prediction, the model assigns the gold answer probability $f_Z(1 \mid X)$; on an incorrect prediction, it assigns $f_Z(0 \mid X) f_{\mathrm{Alt}}(Y \mid X)$. Taking negative logarithms gives

$$-\log f(Y \mid X) = -\log f_Z(Z \mid X) + \mathbb{1}\{Z = 0\}\big(-\log f_{\mathrm{Alt}}(Y \mid X)\big). \tag{14}$$

Taking expectations yields

$$L_Y(f) = L_Z(f) + \mathbb{E}\left[\mathbb{1}\{Z = 0\}\big(-\log f_{\mathrm{Alt}}(Y \mid X)\big)\right].$$

Recalibration replaces f_Z by g and keeps the same alternative distribution, so Equation 2 gives

$$\begin{aligned} L_Y\big(\mathcal{C}_g(f)\big) &= \mathbb{E}[-\log \mathcal{C}_g(f)(Y \mid X)] \\ &= L_Z(g, f) + \mathbb{E}\left[\mathbb{1}\{Z = 0\}\big(-\log f_{\mathrm{Alt}}(Y \mid X)\big)\right]. \end{aligned} \tag{15}$$

Subtracting the two expected-loss decompositions cancels the common alternative-loss term and proves the result. □

Proof of Theorem 1. The recalibrated model has budget at most $P + Q$, so the scaling floor and Proposition 1 give

$$L_Y^*(P + Q) \le L_Y\big(\mathcal{C}_g(f)\big) = L_Y(f) - \big[L_Z(f) - L_Z(g, f)\big]. \tag{16}$$

Substituting Equation 5 and rearranging proves Equation 6. □

A.2 No Improvement Without a Better Component

This section states and proves the result summarized after Proposition 1. With an effective predictor, recalibration improves the correctness probabilities while preserving the alternative distribution. To see what a further improvement would require, consider any model $h(y \mid x)$ and split its probabilities according to whether f's answer is correct:

$$h_Z(1 \mid x) = h(\hat{f}(x) \mid x), \qquad h_Z(0 \mid x) = 1 - h_Z(1 \mid x). \tag{17}$$

Every model h can be viewed as answering two questions: (i) whether f's prediction is correct and (ii) if it is wrong, which alternative answer is correct.

The quantity $h_Z(1 \mid x)$ answers the first question. Once $\hat{f}(x)$ has been ruled out, the distribution $h(y \mid x)/h_Z(0 \mid x)$ answers the second. Define the conditional loss over the alternatives:

$$L_{Y|Z=0}(h) \coloneqq \mathbb{E}\left[-\log \frac{h(Y \mid X)}{h_Z(0 \mid X)} \;\middle|\; Z = 0 \right]. \tag{18}$$

This is the log loss for the second question, evaluated only when f is wrong. In binary classification there is only one alternative answer, so $L_{Y|Z=0}(h) = 0$ for every h. With more than two answers, this term measures how well h distinguishes among the alternatives.

Proposition 2 (No improvement without a better component)**.** *For every model h,*

$$L_Y(h) = L_Z(h_Z, f) + \Pr(Z = 0) L_{Y|Z=0}(h), \tag{19}$$

$$L_Y\big(\mathcal{C}_g(f)\big) = L_Z(g, f) + \Pr(Z = 0) L_{Y|Z=0}(f). \tag{20}$$

If the four component losses are finite and $L_Y(h) < L_Y\big(\mathcal{C}_g(f)\big)$, then

$$L_Z(h_Z, f) < L_Z(g, f) \qquad \textit{or} \qquad L_{Y|Z=0}(h) < L_{Y|Z=0}(f). \tag{21}$$

Proof of Proposition 2. Factor h into its probability that f is correct and its distribution over alternative answers, then take expected negative logarithms to obtain Equation 19. Recalibration changes only the correctness probabilities, giving Equation 20. If neither inequality in Equation 21 held, comparing the two decompositions would give $L_Y(h) \geq L_Y\big(\mathcal{C}_g(f)\big)$, a contradiction. □

Thus beating recalibration requires better performance predictions, better predictions among the alternative answers, or both.

A.3 Small Predictors Under Diminishing Returns

When the predictor is much smaller than the model, its excess predictability is bounded by the model's suboptimality plus a small capacity correction. The following corollary makes this precise under diminishing returns, expressed by a convex loss floor.

Corollary 1 (Small predictors have little advantage beyond model suboptimality)**.** *Under Assumption 1, suppose L_Y^* is convex. For $P > 0$ and $Q \geq 0$, every $f \in \mathcal{F}_P$ with $L_Y(f) < \infty$ and every $g \in \mathcal{G}_Q$ satisfy*

$$L_Z(f) - L_Z(g, f) \leq \frac{2Q}{P}\big[L_Y^*(P/2) - L_Y^*(P)\big] + \delta_P(f). \tag{22}$$

The capacity correction is $2Q/P$ times the decrease in the loss floor from $P/2$ to P. When $Q \ll P$, this correction is small, so the excess predictability is bounded essentially by $\delta_P(f)$. Thus a nearly optimal model leaves little room for a much smaller predictor to outperform its self-confidence.

Proof of Corollary 1. By convexity, the loss reduction per parameter over $[P, P+Q]$ is no greater than over $[P/2, P]$, giving

$$L_Y^*(P) - L_Y^*(P+Q) \leq \frac{2Q}{P}\big[L_Y^*(P/2) - L_Y^*(P)\big].$$

Substituting this bound into Equation 6 proves the result. □

A.4 Suboptimality Does Not Determine Excess Predictability

Proposition 1 shows that a reduction in correctness loss gives an equal reduction in answer loss. Theorem 1 therefore bounds excess predictability by the answer-loss reduction allowed by capacity gain and suboptimality.

However, lowering answer loss does not necessarily lower correctness loss. We can decompose answer loss as

$$L_Y(f) = L_Z(f) + \Pr(Z = 0) L_{Y|Z=0}(f), \tag{23}$$

where $L_{Y|Z=0}(f) \coloneqq \mathbb{E}[-\log f_{\mathrm{Alt}}(Y \mid X) \mid Z = 0]$ is the loss over alternative answers when the model's chosen answer is wrong. A model can lower this alternative loss while keeping its chosen answer and self-confidence unchanged. At a fixed parameter budget, this reduces suboptimality without changing the correctness-prediction problem.

Proposition 3 (Lower suboptimality with unchanged maximum excess predictability)**.** *Let $f \in \mathcal{F}_P$ have finite answer loss and $\Pr(Z = 0) > 0$. For $\lambda \in (0, 1)$, let $h \in \mathcal{F}_P$ keep f's point prediction and self-confidence and mix its alternatives with the uniform distribution,*

$$h_{\mathrm{Alt}}(y \mid x) = (1 - \lambda) f_{\mathrm{Alt}}(y \mid x) + \frac{\lambda}{K - 1}. \tag{24}$$

Then f and h have the same maximum excess predictability at every predictor budget,

$$\sup_{g \in \mathcal{G}_Q} \big[L_Z(h) - L_Z(g, h)\big] = \sup_{g \in \mathcal{G}_Q} \big[L_Z(f) - L_Z(g, f)\big], \tag{25}$$

while $\delta_P(h) < \delta_P(f)$ if $L_{Y|Z=0}(f) > \log(K - 1)$.

At fixed parameter budgets, the bound in Theorem 1 decreases with suboptimality, but the maximum excess predictability need not decrease with it. Models with the same maximum excess predictability can therefore differ in suboptimality by their alternative loss alone.

With two answers, alternative loss is zero, so this source of separation is absent. As K grows, the alternative distribution has more degrees of freedom, allowing changes in answer loss that correctness prediction does not capture.

Proof of Proposition 3. Mixing with a fixed constant is a fixed arithmetic operation, so $h \in \mathcal{F}_P$; the common point prediction gives the same Z, and the common self-confidence gives $L_Z(h) = L_Z(f)$. For $\lambda < 1$, Equation 24 is invertible, so the outputs of f and h on x determine each other through fixed arithmetic. Any $g \in \mathcal{G}_Q$ reading f's output therefore has a counterpart in $\mathcal{G}_Q$ reading h's output with the same correctness loss, and conversely, which proves Equation 25. Applying Equation 23 to f and h and subtracting gives

$$\delta_P(f) - \delta_P(h) = \Pr(Z = 0)\big[L_{Y|Z=0}(f) - L_{Y|Z=0}(h)\big]. \tag{26}$$

Since $-\log$ is convex, $L_{Y|Z=0}(h) \le (1-\lambda) L_{Y|Z=0}(f) + \lambda \log(K-1)$, which is below $L_{Y|Z=0}(f)$ when $L_{Y|Z=0}(f) > \log(K - 1)$. □

A.5 Routing Induces Performance Prediction

The following lemma quantifies the prediction gain supplied by a router that captures additional weak-model errors. It provides the supporting result for the proof of Theorem 2 in Appendix A.6.

We use the notation of Section 4, suppressing X in B_r, B_f, and u. Independent tie-breaking draws are treated as auxiliary inputs, suppressed in the notation and adding no parameters; all expectations include this randomness. For each model, correctness and the actual error probability are

$$Z_i = \mathbb{1}\{Y = \hat{f}_i(X)\}, \quad \epsilon_i(x) = \Pr(Z_i = 0 \mid X = x), \quad i = 1, 2. \tag{27}$$

At escalation rate ρ, the routing decision satisfies

$$B_{r,\rho}(x) \in \{0, 1\}, \qquad \mathbb{E}[B_{r,\rho}(X)] = \rho. \tag{28}$$

The confidence decision also satisfies $\mathbb{E}[B_f] = \rho$, and we write $u = B_r - B_f$.

At a fixed escalation rate, B_f escalates the lowest-confidence ρ fraction of queries and therefore maximizes the predicted weak errors captured, $\mathbb{E}[B_f f_{1,Z}(0 \mid X)]$. Hence

$$\mathbb{E}[u\, f_{1,Z}(0 \mid X)] \le 0. \tag{29}$$

As in Equation 10, additional weak-error capture is therefore bounded by the confidence residual $\mathbb{E}[u(\epsilon_1(X) - f_{1,Z}(0 \mid X))]$. To convert this residual into a prediction gain, use the logit correction g_λ defined in Equation 11. For $\lambda > 0$, this raises predicted error where $B_r = 1$ and $B_f = 0$, and lowers it where $B_r = 0$ and $B_f = 1$. At $\lambda = 0$, it recovers confidence.

Lemma 1 (Routing induces performance prediction beyond confidence)**.** *For any routing score r at a fixed rate ρ with $\mathbb{E}[u^2] > 0$, the predictors in Equation 11 satisfy*

$$\sup_{\lambda \in \mathbb{R}} \big[L_Z(f_1) - L_Z(g_\lambda, f_1)\big] \ge \frac{2\big(\mathbb{E}[u(\epsilon_1(X) - f_{1,Z}(0 \mid X))]\big)^2}{\mathbb{E}[u^2]}. \tag{30}$$

If $\mathbb{E}[u\epsilon_1(X)] > 0$, then there exists $\lambda > 0$ such that $L_Z(g_\lambda, f_1) < L_Z(f_1)$. If $\mathbb{E}[u^2] = 0$, then $g_\lambda = f_{1,Z}$ almost surely for every λ.

Proof of Lemma 1. If $\mathbb{E}[u^2] = 0$, then $u = 0$ almost surely, so Equation 11 gives $g_\lambda = f_{1,Z}$ for every λ. Henceforth assume $\mathbb{E}[u^2] > 0$. We first bound the loss after shifting the confidence logit, then choose the shift that gives the largest guaranteed improvement.

Step 1: bound the change in loss on each query. Fix a query x and a realization of the tie-breaking randomness, which is independent of the answer given x. Write $a(x) = \operatorname{logit} f_{1,Z}(0 \mid x)$. For a predicted error logit s, let $\sigma(s) = 1/(1 + e^{-s})$ be the corresponding error probability. Since the weak model is wrong with probability $\epsilon_1(x)$, the conditional expected binary log loss is

$$\begin{aligned} \ell_x(s) &= -\epsilon_1(x) \log \sigma(s) - [1 - \epsilon_1(x)] \log[1 - \sigma(s)] \\ &= \log(1 + e^s) - \epsilon_1(x) s. \end{aligned}$$

Differentiating with respect to s gives

$$\ell_x'(s) = \sigma(s) - \epsilon_1(x), \qquad \ell_x''(s) = \sigma(s)[1 - \sigma(s)] \le \frac{1}{4}.$$

Taylor's theorem therefore bounds the loss at any shift $h \in \mathbb{R}$ by

$$\ell_x(a(x) + h) \le \ell_x(a(x)) + [f_{1,Z}(0 \mid x) - \epsilon_1(x)]h + \frac{h^2}{8}.$$

The factor $1/8$ is the Taylor factor $1/2$ times the curvature bound $1/4$. Substituting $h = \lambda u(x)$ and rearranging yields

$$\ell_x(a(x)) - \ell_x(a(x) + \lambda u(x)) \ge \lambda u(x)[\epsilon_1(x) - f_{1,Z}(0 \mid x)] - \frac{\lambda^2 u(x)^2}{8}.$$

Step 2: average over queries and tie-breaking. The two losses on the left are the conditional losses of $f_{1,Z}$ and g_λ, respectively. Taking expectations over both queries and tie-breaking thus gives

$$L_Z(f_1) - L_Z(g_\lambda, f_1) \ge \lambda \mathbb{E}[u(\epsilon_1(X) - f_{1,Z}(0 \mid X))] - \frac{\lambda^2}{8}\mathbb{E}[u^2]. \tag{31}$$

These expected losses are finite: $L_Z(f_1) \le L_Y(f_1) < \infty$, and the displayed upper bound, together with $|u| \le 1$, bounds the corrected loss for every finite λ.

Step 3: choose the correction strength. For this optimization, abbreviate the confidence residual and the squared routing difference as

$$A = \mathbb{E}[u(\epsilon_1(X) - f_{1,Z}(0 \mid X))], \qquad V = \mathbb{E}[u^2] > 0.$$

Completing the square in the right-hand side of Equation 31 gives

$$\lambda A - \frac{\lambda^2 V}{8} = \frac{2A^2}{V} - \frac{V}{8}\left(\lambda - \frac{4A}{V}\right)^2.$$

Thus $\lambda_* = 4A/V$ guarantees excess predictability of at least $2A^2/V$. The supremum over all λ is at least the excess predictability at λ_*, proving Equation 30. Finally, if B_r captures more weak-model errors, then Equation 10 gives $A \ge \mathbb{E}[u\epsilon_1(X)] > 0$. Consequently $\lambda_* > 0$ and $2A^2/V > 0$, so this correction is an effective performance predictor. □

A.6 Proofs for Section 4

Using the notation and supporting lemma from Appendix A.5, we prove Theorem 2 by combining the prediction gain with the loss-floor bound from Theorem 1.

Proof of Theorem 2. Recalibration with g_λ reuses f_1 and adds the budget-Q routing score r, the shared escalation rate ρ, and the correction strength λ. The resulting model has at most $P_1 + Q + 2$ parameters and belongs to $\mathcal{F}_{P_1+Q+2}$. Exact transfer and the loss floor therefore bound the excess predictability for every λ, as in Theorem 1:

$$L_Z(f_1) - L_Z(g_\lambda, f_1) \le \delta_{P_1}(f_1) + L_Y^*(P_1) - L_Y^*(P_1 + Q + 2). \tag{32}$$

Write the upper bound on the excess predictability as

$$\Delta = \delta_{P_1}(f_1) + L_Y^*(P_1) - L_Y^*(P_1 + Q + 2).$$

This quantity is nonnegative because suboptimality is nonnegative and the loss floor is non-increasing in the parameter budget. If $\mathbb{E}[u^2] = 0$, the routing decisions agree almost surely and the left side is zero. Now assume $\mathbb{E}[u^2] > 0$.

Step 1: bound the confidence residual using the excess predictability. Lemma 1 gives a lower bound on the maximum excess predictability obtained by correcting confidence along u. Equation 32 bounds the excess predictability of every such correction by Δ, so it also bounds their supremum. Together, these bounds give

$$\frac{2\big(\mathbb{E}[u(\epsilon_1(X) - f_{1,Z}(0 \mid X))]\big)^2}{\mathbb{E}[u^2]} \le \sup_{\lambda \in \mathbb{R}} [L_Z(f_1) - L_Z(g_\lambda, f_1)] \le \Delta.$$

Multiplying by $\mathbb{E}[u^2]/2$ and taking nonnegative square roots yields

$$|\mathbb{E}[u(\epsilon_1(X) - f_{1,Z}(0 \mid X))]| \le \sqrt{\frac{\mathbb{E}[u^2]}{2}\,\Delta}.$$

By Equation 10, this also bounds the additional weak errors captured, $\mathbb{E}[u\epsilon_1(X)]$.

Step 2: bound the difference between routing decisions using the escalation rate. Since $B_r, B_f \in \{0, 1\}$, we have $u^2 = |B_r - B_f| = \mathbb{1}\{B_r \ne B_f\}$. The absolute difference is at most both $B_r + B_f$ and $(1 - B_r) + (1 - B_f)$, so

$$u^2 \le B_r + B_f, \qquad u^2 \le (1 - B_r) + (1 - B_f).$$

Taking expectations and using $\mathbb{E}[B_r] = \mathbb{E}[B_f] = \rho$ gives both $\mathbb{E}[u^2] \le 2\rho$ and $\mathbb{E}[u^2] \le 2(1 - \rho)$. Combining the two steps gives

$$\mathbb{E}[u\epsilon_1(X)] \le \sqrt{\frac{\mathbb{E}[u^2]}{2}\,\Delta} \le \sqrt{\min\{\rho, 1 - \rho\}\Delta}.$$

Substituting the definition of Δ proves Equation 12. □

A.7 Cascade Advantage over Confidence Routing

To pass from weak-error capture to cascade performance, we must also account for strong-model errors on escalated queries. The cascade prediction and error rate are

$$\hat{f}_{\mathrm{cas},B_r}(x) = (1 - B_r(x))\hat{f}_1(x) + B_r(x)\hat{f}_2(x). \tag{33}$$

$$\mathrm{Err}(f_{\mathrm{cas},B_r}) = \mathrm{Err}(f_1) - \mathbb{E}\big[B_r\big(\epsilon_1(X) - \epsilon_2(X)\big)\big]. \tag{34}$$

Subtracting this identity for confidence routing gives Equation 9. If the learned router encounters at least as many strong-model errors as confidence routing, its cascade advantage is at most its additional weak-error capture.

Corollary 2 (Cascade advantage over confidence routing)**.** *Under the conditions of Theorem 2, if* $\mathbb{E}[u\epsilon_2(X)] \geq 0$*, then*

$$\mathrm{Err}(f_{\mathrm{cas},B_f}) - \mathrm{Err}(f_{\mathrm{cas},B_r}) \leq \sqrt{\min\{\rho, 1-\rho\}\,[\delta_{P_1}(f_1) + L_Y^*(P_1) - L_Y^*(P_1 + Q + 2)]}. \tag{35}$$

Proof of Corollary 2. By Equation 9, the cascade accuracy gain equals $\mathbb{E}[u\epsilon_1(X)] - \mathbb{E}[u\epsilon_2(X)] \leq \mathbb{E}[u\epsilon_1(X)]$, which Theorem 2 bounds. □

In a longer cascade, the analysis applies at each stage to its incoming queries, with f_2 representing a fixed downstream policy.

B Additional Experimental Setup

B.1 Models and Data Splits

Table 1 lists the 16 releases evaluated before and after task-specific finetuning. A *model pair* refers to a release and its selected finetuned checkpoint on one task.

Base release	Instruction-tuned release
Qwen3.5-9B-Base	Qwen3.5-9B
Gemma-3-12B-PT	Gemma-3-12B-IT
Llama-3.1-8B	Llama-3.1-8B-Instruct
Apertus-8B-2509	Apertus-8B-Instruct-2509
OLMo-3-1025-7B	OLMo-3-7B-Instruct
Falcon3-7B-Base	Falcon3-7B-Instruct
Mistral-7B-v0.3	Mistral-7B-Instruct-v0.3
Mistral-NeMo-Minitron-8B-Base	Mistral-NeMo-Minitron-8B-Instruct

Table 1: Base and instruction-tuned releases from eight model families, giving 16 models per task: Qwen3.5 (Qwen Team, 2026), Gemma 3 (Gemma Team, 2025), Llama 3.1 (Grattafiori et al., 2024), Apertus (Hernández-Cano et al., 2025), OLMo 3 (Team Olmo et al., 2025), Falcon3 (Falcon-LLM Team, 2024), Mistral 7B (Jiang et al., 2023), and Mistral-NeMo-Minitron (Sreenivas et al., 2024).

Table 2 gives the split sizes. For `WinoGrande`, `CommonsenseQA`, and `GSM8K`, we hold out development questions from the original training set and reserve five remaining questions as demonstrations. The first two tasks use their original validation sets for testing; `GSM8K` uses its test set. We split `MMLU-Pro`'s 12,032 labeled questions into 6,000 for teacher generation, 3,000 for development, and 3,032 for test, stratified by subject and source. Its official validation set supplies five demonstrations per subject. Demonstrations are shared across models and excluded from all other splits; `MMLU-Pro` metrics pool subjects. The 85/15 development partition for predictor fitting and selection is shared across model versions and predictor sizes.

			Development split		
Task	Finetuning	Development	Fit (85%)	Select (15%)	Test
`WinoGrande`	32,313	8,080	6,868	1,212	1,267
`CommonsenseQA`	6,394	3,342	2,840	502	1,221
`GSM8K`	5,088	2,241	1,904	337	1,319
`MMLU-Pro`	5,233	3,000	2,550	450	3,032

Table 2: Questions used at each stage, excluding demonstrations. Predictor fitting and selection partition the development set. For `GSM8K` and `MMLU-Pro`, finetuning counts include only questions with an accepted teacher solution.

B.2 Answer Scoring and Losses

Prompts and option scores. Base models receive plain-text demonstrations; instruction-tuned models use user/assistant turns in their native chat templates, with thinking disabled. On `CommonsenseQA`, the option score is the answer completion's log likelihood, without length normalization. On `WinoGrande`, each option fills the blank, and we score the shared continuation under that filled context.

Sampled-answer probabilities. Generation uses 128 samples per question at temperature 0.9 and top-$p = 1$, with a 2,048-token output limit. Let $c_x(j)$ count each distinct extracted answer in S_x, including a separate category for extraction failures. We reserve a small mass for unseen answers in category u:

$$p(j \mid x) = \frac{c_x(j)}{128.5} \quad (j \in S_x), \qquad p(u \mid x) = \frac{0.5}{128.5}. \tag{36}$$

We predict the most frequent successfully extracted answer and use its probability as self-confidence. An unsampled gold answer receives probability $p(u \mid x)$ when computing answer loss. If all samples fail extraction, the prediction is incorrect with confidence $p(u \mid x)$.

Loss conventions. All losses are in nats; binary cross-entropy clips probabilities to $[10^{-6}, 1 - 10^{-6}]$. Table 3 gives the reference losses for empirical suboptimality (Equation 7), held fixed before and after finetuning.

Task	Loss (nats)	Finetuned model attaining the reference
`WinoGrande`	0.3229	Mistral-NeMo-Minitron-8B-Base
`CommonsenseQA`	0.3390	Qwen3.5-9B-Base
`GSM8K`	0.1378	Qwen3.5-9B
`MMLU-Pro`	0.7247	Qwen3.5-9B-Base

Table 3: Reference answer losses for empirical suboptimality, computed over the released and finetuned models retained on each task.

B.3 TASK-SPECIFIC FINETUNING AND MODEL RETENTION

Teacher solutions. For the generation tasks, QWEN3.5-122B-A10B receives each question without demonstrations and is asked to reason step by step and end with the required answer format. It generates four solutions at temperature 0.7 and top-$p = 0.9$, with a 2,048-token limit. We keep one whose extracted answer matches the gold answer, omitting questions with no accepted solution. These targets are generated once and shared across models and seeds.

Optimization and selection. LoRA uses rank 8, scaling factor 32, and dropout 0.1 on attention query and value projections; for QWEN3.5-9B, it also adapts fused query–key–value projections in linear-attention layers. We train three seeds at learning rate 5×10^{-5}, weight decay 0.01, and effective batch size 8. Only the gold completion or accepted teacher solution contributes to the loss. Table 4 gives the schedules and selection criteria. Greedy accuracy selects generation-task checkpoints; reported results use 128-sample majority answers.

Task	Steps	Token limit	Checkpoint selection
WinoGrande	2,000	256	Lowest development answer loss
CommonsenseQA	1,200	512	Lowest development answer loss
GSM8K	1,500	4,096	Highest development greedy accuracy
MMLU-Pro	1,200	6,144	Highest development greedy accuracy

Table 4: Finetuning settings for the weak models. We save four checkpoints per seed, one at each quarter of training, and select among the 12 candidates from three seeds.

Model retention. We retain a model pair when finetuning increases development accuracy. Accuracy uses the highest-scoring option or the 128-sample majority answer, depending on the task. The rule uses no test labels and retains 16, 16, 14, and 13 pairs in task order. All reported comparisons, including predictor-size means and empirical loss references, use these retained pairs.

Table 5 shows development accuracy for all models. Finetuning improves accuracy for every model on the two option-scoring tasks. Accuracy falls for two instruction-tuned releases on GSM8K and three on MMLU-Pro; all five are among the most accurate releases on their respective tasks. The illustrative model, LLAMA-3.1-8B base, is retained on every task.

	WinoGrande		CommonsenseQA		GSM8K		MMLU-Pro	
Model	Rel.	FT	Rel.	FT	Rel.	FT	Rel.	FT
QWEN3.5-9B-BASE	88.6	93.3	81.5	83.7	94.3	96.0	66.8	76.7
QWEN3.5-9B	81.9	93.7	81.5	83.6	95.9	96.1	78.9†	77.3†
GEMMA-3-12B-PT	89.5	94.2	76.5	82.4	87.1	92.7	53.5	63.0
GEMMA-3-12B-IT	82.1	93.1	76.2	80.4	93.4	95.4	64.6†	62.6†
LLAMA-3.1-8B	88.3	93.4	73.1	78.8	70.6	82.7	41.6	50.2
LLAMA-3.1-8B-INSTRUCT	84.2	93.3	75.7	79.7	96.1	96.8	53.1	57.5
APERTUS-8B-2509	87.8	92.1	71.9	78.7	67.2	75.5	35.1	46.0
APERTUS-8B-INSTRUCT-2509	86.3	92.4	70.2	78.2	78.0	83.0	40.4	46.2
OLMO-3-1025-7B	86.2	91.3	71.9	80.2	90.9	94.4	41.7	58.3
OLMO-3-7B-INSTRUCT	80.5	89.8	74.8	79.5	96.8†	95.9†	59.6†	56.2†
FALCON3-7B-BASE	87.3	91.2	76.3	78.8	91.6	94.0	50.5	60.3
FALCON3-7B-INSTRUCT	81.5	91.7	78.2	81.4	95.4	95.5	60.5	61.3
MISTRAL-7B-V0.3	89.5	93.9	70.3	80.1	55.3	77.6	36.8	46.0
MISTRAL-7B-INSTRUCT-V0.3	87.8	94.3	71.2	80.0	73.8	80.6	39.9	45.7
MISTRAL-NEMO-MINITRON-8B-BASE	89.9	94.2	84.2	86.3	81.5	88.6	42.4	53.3
MISTRAL-NEMO-MINITRON-8B-INSTRUCT	82.9	92.6	80.1	85.3	97.9†	96.5†	49.3	54.1

Table 5: **Development accuracy (%) of released models (Rel.) and their selected finetuned checkpoints (FT).** † marks a model whose finetuned checkpoint's development accuracy does not exceed that of its released model. Marked pairs are excluded from all reported comparisons, including Figure 8.

Trajectory experiment. For Figure 3, we rerun LLAMA-3.1-8B base with seed 0 on CommonsenseQA and evaluate every 100 steps through step 600. Step 0 reuses the original results; later checkpoints each use a new Platt map and the mean loss of three newly trained 4B predictors.

B.4 PERFORMANCE PREDICTOR TRAINING

Inputs. The predictor receives the question, the model's final answer, its answer distribution, and a prompt to judge correctness. On generation tasks, it also receives one sampled reasoning trace ending in the majority answer. The answer distribution is represented by all option probabilities on the option-scoring tasks, the five most frequent answers on `GSM8K`, and all sampled option counts on `MMLU-Pro`. Inputs exclude demonstrations, chat-template tokens, and gold answers, and are never truncated.

Architecture. We encode the input with QWEN3-EMBEDDING-4B to obtain $\phi(x)$ and apply a linear correction to the self-confidence logit:

$$g(1 \mid x) = \sigma(a + b \ \mathrm{logit}\, f_Z(1 \mid x) + \mathrm{head}(\phi(x))) \,. \tag{37}$$

Here a, b are learned scalars and σ is the sigmoid function.

Initialization and training. For each evaluated model, we fit Platt calibration (Platt, 2000) by minimizing binary cross-entropy on the predictor-fitting split:

$$f_Z^{\mathrm{Platt}}(1 \mid x) = \sigma(\alpha + \beta \ \mathrm{logit}\, f_Z(1 \mid x)) \,. \tag{38}$$

This map is shared across predictor sizes and initializes $(a, b) = (\alpha, \beta)$ in Equation 37; the linear head starts at zero. The QWEN3-EMBEDDING backbone uses the final token's hidden state and LoRA on query and value projections with rank 16, scaling factor 32, and dropout 0.05. We train the adapters, head, and a, b for up to three epochs at learning rate 10^{-4} and effective batch size 16, with weight decay 10^{-4} on the head only. Among ten seeds and all epochs, including epoch zero, we select the lowest correctness loss on the predictor-selection split.

B.5 CASCADE EVALUATION

Strong model. We use QWEN3.5-122B-A10B as the strong model. On `WinoGrande` and `CommonsenseQA`, we finetune it and select the checkpoint with the lowest development answer loss. These runs use the weak models' training data, prompts, and optimization settings for 2,000 and 1,600 steps, respectively. We save four checkpoints at quarter intervals and select the lowest development answer loss among them and the released model. Evaluation uses the weak models' demonstrations and option-scoring rules.

On `GSM8K` and `MMLU-Pro`, the released strong model generates 128 samples in thinking mode at temperature 1.0, top-$p = 0.95$, top-$k = 20$, and presence penalty 1.5, with a 32,768-token limit. Answer extraction and majority selection follow the weak-model evaluation.

Accuracy and integration. For either routing score r, we sort the n test questions in decreasing order, breaking ties uniformly at random. At $\rho = k/n$, we escalate the first k questions. Expected cascade accuracy is

$$1 - \mathrm{Err}(f_{\mathrm{cas},B_{r,\rho}}) = \frac{1}{n}\mathbb{E}\left[\sum_{i=1}^{n} Z_{1,i} + \sum_{i=1}^{n} B_{r,\rho}(x_i)(Z_{2,i} - Z_{1,i})\right], \tag{39}$$

where $Z_{1,i}$ and $Z_{2,i}$ indicate weak- and strong-model correctness, and the expectation is over tie-breaking with answers held fixed. We linearly interpolate between attainable rates and integrate with the trapezoidal rule. Accuracy is linear across each tied group, so its endpoints suffice. We report routing advantage $100D$ in percentage points averaged over escalation rates.

C Additional Experimental Results

All results use the retained pairs; predictors use the selected 4B backbone unless another size is stated. Values listed in task order refer to `WinoGrande`, `CommonsenseQA`, `GSM8K`, and `MMLU-Pro`.

C.1 Correlation Estimates and Uncertainty

We report Pearson's r and Spearman's r_s, with one-sided p-values p_+ for positive correlation. We use SciPy's standard tests, assuming independent observations and making no adjustment for multiple comparisons.

Comparison	Task	Pearson r	p_+	Spearman r_s	p_+
$\widehat{\delta}$ vs. E	`WinoGrande`	0.98	< 0.001	0.73	< 0.001
	`CommonsenseQA`	0.98	< 0.001	0.94	< 0.001
	`GSM8K`	0.93	< 0.001	0.94	< 0.001
	`MMLU-Pro`	0.65	< 0.001	0.74	< 0.001
$\Delta\widehat{\delta}$ vs. ΔE	`WinoGrande`	0.99	< 0.001	0.90	< 0.001
	`CommonsenseQA`	0.99	< 0.001	0.97	< 0.001
	`GSM8K`	0.78	< 0.001	0.89	< 0.001
	`MMLU-Pro`	0.49	0.044	0.20	0.259
$\widehat{\delta}$ vs. D	`WinoGrande`	0.79	< 0.001	0.92	< 0.001
	`CommonsenseQA`	0.66	< 0.001	0.87	< 0.001
	`GSM8K`	0.91	< 0.001	0.77	< 0.001
	`MMLU-Pro`	0.82	< 0.001	0.85	< 0.001
$\Delta\widehat{\delta}$ vs. ΔD	`WinoGrande`	0.63	0.004	0.76	< 0.001
	`CommonsenseQA`	0.42	0.054	0.56	0.011
	`GSM8K`	0.73	0.001	0.51	0.031
	`MMLU-Pro`	0.40	0.087	0.18	0.277

Table 6: **Correlations for the main 4B comparisons.** p_+ tests a positive association using the tests described in Appendix C.1. Endpoint comparisons contain 32, 32, 28, and 26 model versions in task order; paired changes contain 16, 16, 14, and 13 model pairs.

C.2 Released and Finetuned Models Separately

Figures 6 and 7 repeat Figures 1 and 4 with released and finetuned models in separate panels, using the same models and the selected 4B predictors. Among released models alone, excess predictability correlates with empirical suboptimality at 1.00, 0.98, 0.89, and 0.49 in task order, and routing advantage at 0.67, 0.43, 0.91, and 0.81. The main-text associations therefore do not depend on contrasting the two groups. Among finetuned models, routing advantage remains positively correlated with suboptimality on every task, at 0.79, 0.81, 0.94, and 0.64, and excess predictability at 0.14, 0.65, 0.99, and 0.32. The weak value on `WinoGrande` reflects the narrow range after finetuning: all finetuned models lie within 0.22 nats of the best observed loss, and 14 of 16 have slightly negative excess predictability. On `MMLU-Pro`, both groups show the weaker association discussed in Section 3.4.2.

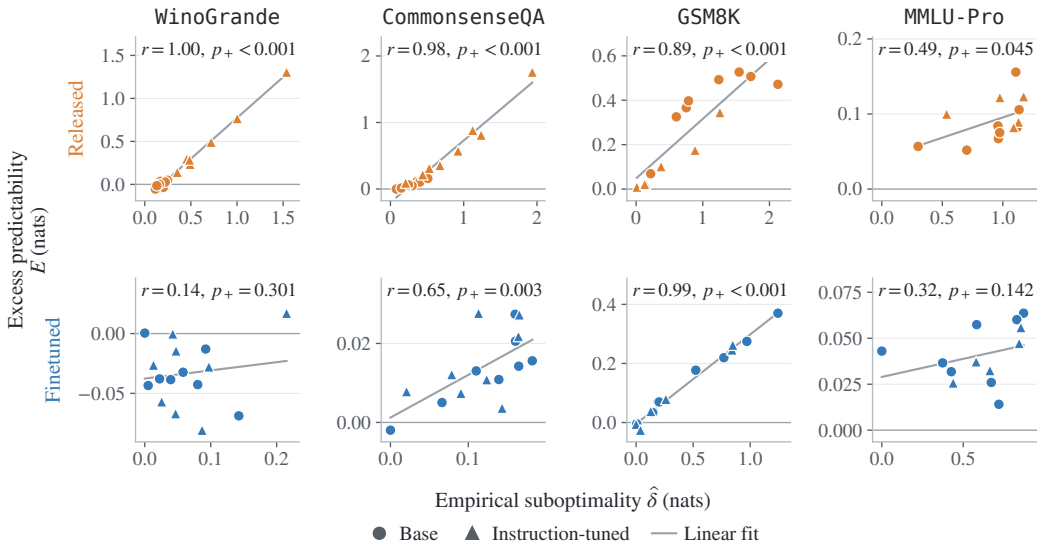

Figure 6: **Excess predictability of the 4B predictor, with released and finetuned models shown separately.** Each column is a task; the top row shows released models and the bottom row their finetuned versions, using the same points as Figure 1. Excess predictability E is the predictor's correctness-log-loss gain over self-confidence; empirical suboptimality $\widehat{\delta}$ is the gap to the best observed answer log loss on each task. Circles denote base models and triangles instruction-tuned models. Lines are least-squares fits within each panel; r is Pearson correlation and p_+ its one-sided, unadjusted p-value for positive correlation. Correlations and p-values are in Table 7. Axis ranges differ across panels.

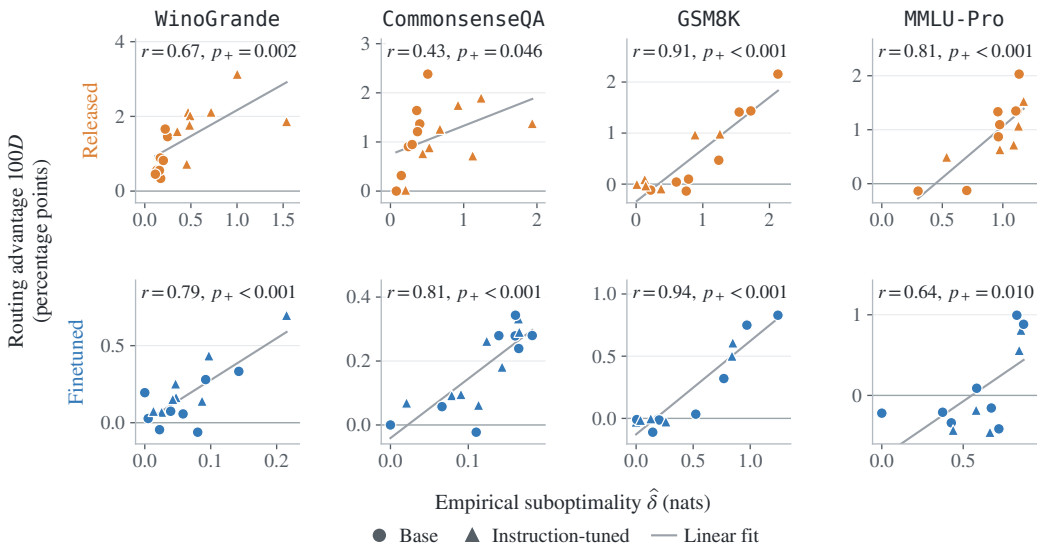

Figure 7: **Routing advantage of the 4B predictor, with released and finetuned models shown separately.** Routing advantage $100D$ (percentage points) versus empirical suboptimality $\widehat{\delta}$ (nats), using the same points as Figure 4. $100D$ is the cascade accuracy gain over self-confidence routing, averaged over equal escalation rates; $\widehat{\delta}$ is the gap to the best observed answer log loss on each task. Columns are tasks; orange (top) denotes released models and blue (bottom) their finetuned versions. Circles denote base releases and triangles instruction-tuned releases. Lines are least-squares fits; r is Pearson correlation and p_+ its one-sided, unadjusted p-value for positive correlation. Axis ranges differ across panels.

Comparison	Task	Pearson r	p_+	Spearman r_s	p_+
$\widehat{\delta}$ vs. E (released)	WinoGrande	1.00	< 0.001	0.92	< 0.001
	CommonsenseQA	0.98	< 0.001	0.97	< 0.001
	GSM8K	0.89	< 0.001	0.90	< 0.001
	MMLU-Pro	0.49	0.045	0.60	0.015
$\widehat{\delta}$ vs. E (finetuned)	WinoGrande	0.14	0.301	-0.03	0.547
	CommonsenseQA	0.65	0.003	0.64	0.004
	GSM8K	0.99	< 0.001	0.98	< 0.001
	MMLU-Pro	0.32	0.142	0.44	0.066
$\widehat{\delta}$ vs. D (released)	WinoGrande	0.67	0.002	0.86	< 0.001
	CommonsenseQA	0.43	0.046	0.59	0.008
	GSM8K	0.91	< 0.001	0.81	< 0.001
	MMLU-Pro	0.81	< 0.001	0.80	< 0.001
$\widehat{\delta}$ vs. D (finetuned)	WinoGrande	0.79	< 0.001	0.56	0.011
	CommonsenseQA	0.81	< 0.001	0.82	< 0.001
	GSM8K	0.94	< 0.001	0.82	< 0.001
	MMLU-Pro	0.64	0.010	0.63	0.010

Table 7: **Within-stage correlations for 4B predictors.** p_+ tests a positive association using the tests described in Appendix C.1. Each stage contains 16, 16, 14, and 13 models in task order.

C.3 PREDICTOR CAPACITY

We ask how much a larger predictor adds beyond confidence and whether that additional benefit shrinks after finetuning the model it evaluates. We compare two-parameter Platt calibration with predictors on the 0.6B, 4B, and 8B QWEN3-EMBEDDING backbones, using the same retained pairs for every size: 16, 16, 14, and 13 in task order.

What larger predictors add. Figure 8 compares predictor gains over raw confidence and Platt-calibrated confidence. On released models, Platt calibration accounts for most of the mean 4B excess predictability on the first three tasks, whereas most of the gain on `MMLU-Pro` lies beyond calibration. The 0.6B predictor adds little beyond calibration, while larger predictors generally improve on it. The 8B predictor has the lowest mean correctness loss in seven of eight task–stage groups. The exception is finetuned `WinoGrande`, where all three sizes perform worse than both baselines.

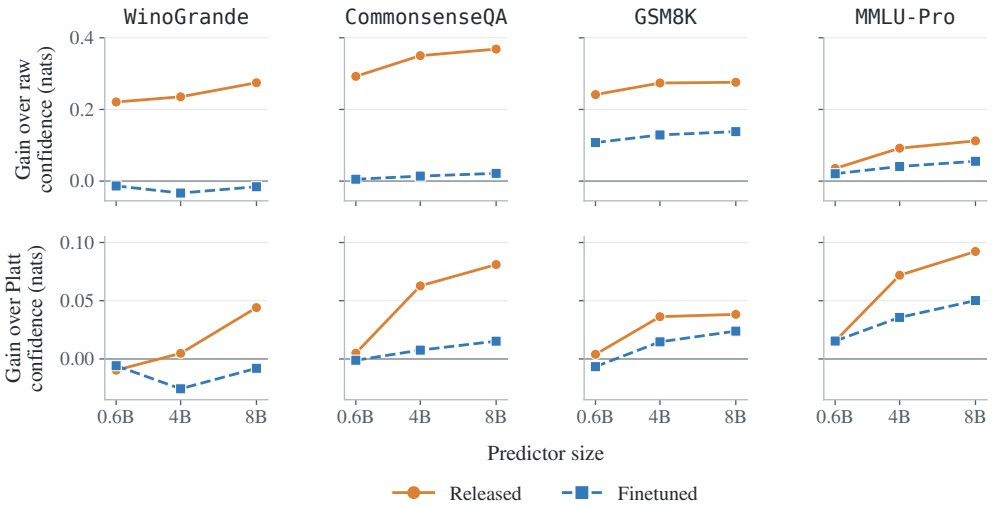

Figure 8: **Predictor gains over self-confidence and Platt calibration.** Mean baseline loss minus predictor loss over the retained models on each task; positive values favor the predictor. Columns show tasks; rows compare against raw and Platt-calibrated confidence. Orange solid lines with circles: released models; blue dashed lines with squares: finetuned models. Predictor size is on the horizontal axis; vertical scales are shared within each row. Lines connect the three evaluated sizes.

Gains beyond Platt calibration also decrease. Since Platt calibration accounts for much of the released models' excess predictability, one might suspect that finetuning reduces excess predictability merely by calibrating self-confidence. Theorem 1 bounds excess predictability including gains from calibration. This control tests whether the observed reduction after finetuning extends to predictor gains beyond a fitted two-parameter Platt map of self-confidence. Let c_f be a model's development-fitted Platt calibrator (Appendix B.4) and g_f its selected 4B predictor. The predictor's gain beyond calibration,

$$R(f) = L_Z(c_f, f) - L_Z(g_f, f), \tag{40}$$

excludes the part of excess predictability $E(f)$ that a two-parameter calibrator recovers. Table 8 reports $R(f)$ before and after finetuning on the retained pairs. This remaining gain also decreases after finetuning: in 50 of the 59 pairs, with every task's mean reduction having a 95% interval above zero. The intervals use 10,000 bootstrap resamples of test questions with NumPy seed 0 and percentile limits, sharing question multiplicities across all models and both stages within each task; they condition on the fitted models and predictors. Across released and finetuned endpoints, Pearson correlations between $\widehat{\delta}$ and R remain positive at 0.45, 0.67, 0.90, and 0.78 in task order, compared with 0.98, 0.98, 0.93, and 0.65 for E. The paired-change correlations weaken substantially: correlations between $\Delta\widehat{\delta}$ and ΔR are 0.24, 0.40, 0.55, and 0.25, compared with 0.99, 0.99, 0.78, and 0.49 for

ΔE. On finetuned `WinoGrande`, the mean gain is negative: the selected predictor underperforms Platt confidence, as in Figure 8. These results concern gains beyond Platt calibration, rather than beyond every confidence-only calibrator.

Task	Mean gain beyond Platt, R (nats)			Pairs with a decrease
	Before finetuning	After finetuning	Before − after [95% CI]	
`WinoGrande`	0.005	−0.026	0.030 [0.010, 0.050]	12/16
`CommonsenseQA`	0.063	0.008	0.055 [0.045, 0.065]	14/16
`GSM8K`	0.036	0.015	0.022 [0.014, 0.029]	12/14
`MMLU-Pro`	0.072	0.036	0.036 [0.030, 0.042]	12/13

Table 8: **Excess predictability beyond Platt-calibrated self-confidence also decreases after finetuning.** For each model, R is the Platt-calibrated correctness loss minus the predictor's correctness loss; positive values favor the predictor. The first two value columns average R over the retained released models and over their finetuned versions. The third averages the paired difference, with a 95% bootstrap interval over test questions. The last column counts the pairs in which R decreases after finetuning.

Finetuning reduces the benefit of additional capacity. The additional gain from increasing predictor size from 0.6B to 8B falls after finetuning in 49 of the 59 retained pairs, with 9 of the 10 exceptions on `GSM8K`. Across model versions, this additional gain correlates with empirical suboptimality at 0.60, 0.56, 0.73, and 0.89 in task order. These associations do not identify the capacity contribution in Theorem 1, which bounds excess predictability without decomposing the observed gain into suboptimality and capacity contributions.

Robustness across predictor sizes. Figures 9–12 repeat the main-text 4B comparisons with 0.6B and 8B predictors, using the same retained models. At 8B, both excess predictability and routing advantage remain positively correlated with suboptimality on every task. The 0.6B results are less consistent: the correlation is negative for excess predictability on `MMLU-Pro`, and for routing advantage on both `WinoGrande` and `MMLU-Pro`. On these tasks, the 0.6B predictor adds little beyond calibration. Across all three sizes, larger reductions in suboptimality after finetuning accompany larger reductions in excess predictability, with Pearson correlations of at least 0.99, 0.98, 0.73, and 0.48 in task order.

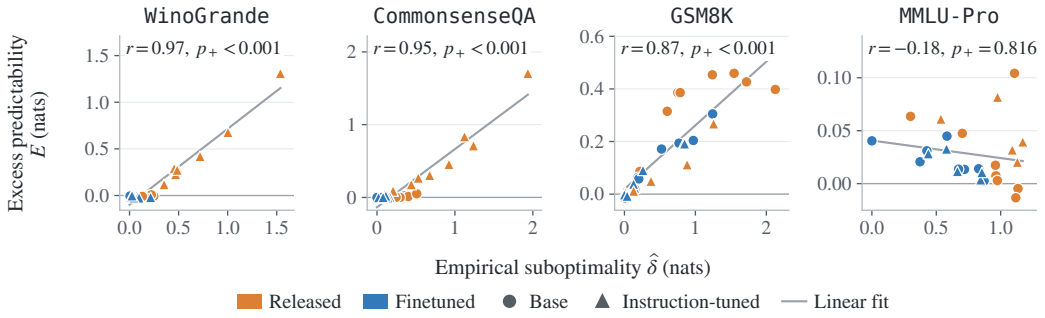

Figure 9: **Excess predictability with the 0.6B predictor.** E is the predictor's correctness-log-loss gain over self-confidence; $\widehat{\delta}$ is the gap to the best observed answer log loss on each task. Orange denotes released models and blue finetuned models; circles denote base releases and triangles instruction-tuned releases. Lines are least-squares fits; r is Pearson correlation and p_+ its one-sided, unadjusted p-value for positive correlation.

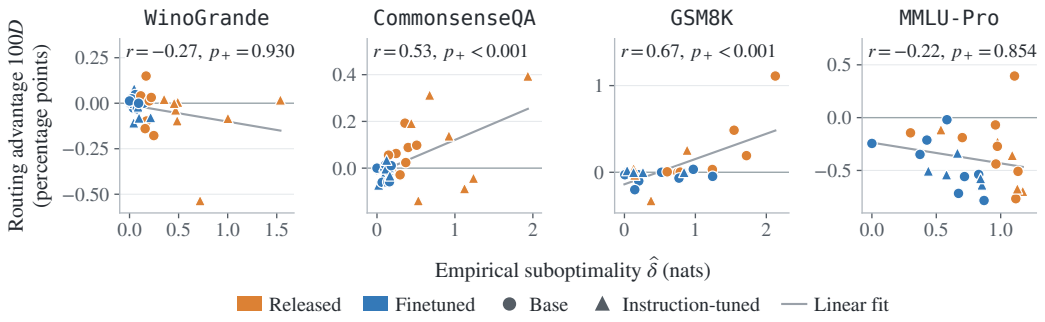

Figure 10: **Routing advantage with the 0.6B predictor.** $100D$ is the cascade accuracy gain over self-confidence routing, averaged over equal escalation rates, in percentage points; $\widehat{\delta}$ is the gap to the best observed answer log loss on each task. Orange denotes released models and blue finetuned models; circles denote base releases and triangles instruction-tuned releases. Lines are least-squares fits; r is Pearson correlation and p_+ its one-sided, unadjusted p-value for positive correlation.

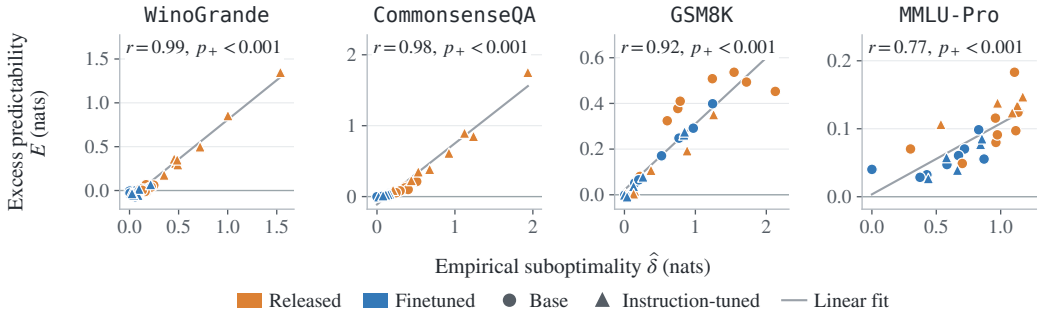

Figure 11: **Excess predictability with the 8B predictor.** E is the predictor's correctness-log-loss gain over self-confidence; $\widehat{\delta}$ is the gap to the best observed answer log loss on each task. Orange denotes released models and blue finetuned models; circles denote base releases and triangles instruction-tuned releases. Lines are least-squares fits; r is Pearson correlation and p_+ its one-sided, unadjusted p-value for positive correlation.

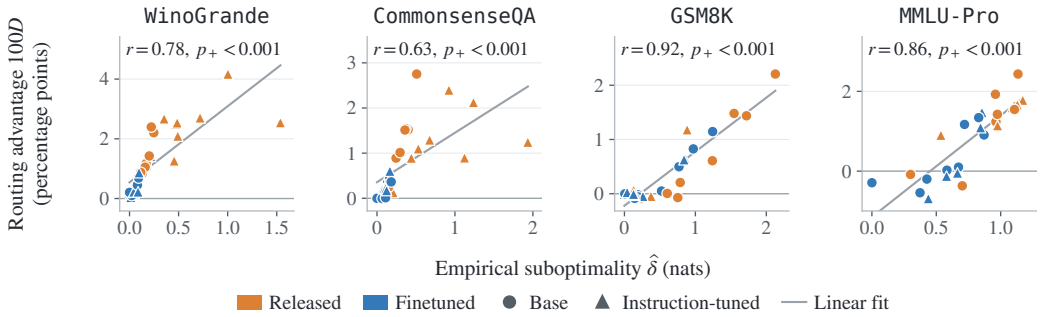

Figure 12: **Routing advantage with the 8B predictor.** $100D$ is the cascade accuracy gain over self-confidence routing, averaged over equal escalation rates, in percentage points; $\widehat{\delta}$ is the gap to the best observed answer log loss on each task. Orange denotes released models and blue finetuned models; circles denote base releases and triangles instruction-tuned releases. Lines are least-squares fits; r is Pearson correlation and p_+ its one-sided, unadjusted p-value for positive correlation.

C.4 Cascade Accuracy Across Tasks

Figure 13 extends the main-text `CommonsenseQA` example to all four tasks, using LLAMA-3.1-8B base before and after finetuning. The gap between predictor and self-confidence routing measures the benefit of choosing different questions at the same escalation rate.

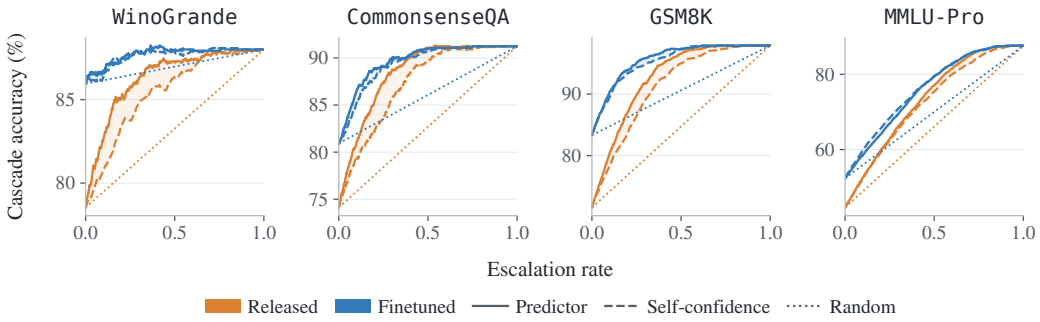

Figure 13: **Cascade accuracy across all four tasks.** We pair LLAMA-3.1-8B base with QWEN3.5-122B-A10B, finetuned on `WinoGrande` and `CommonsenseQA` and released on the other two tasks. Orange denotes the released weak model and blue its finetuned version. Solid curves use the selected 4B predictor g, dashed curves use self-confidence $f_{1,Z}$, and dotted lines show random routing. Shading shows the accuracy difference between the two routers.

Finetuning reduces routing advantage. Figure 14 compares changes in answer loss and routing advantage across the 59 retained model-task pairs.

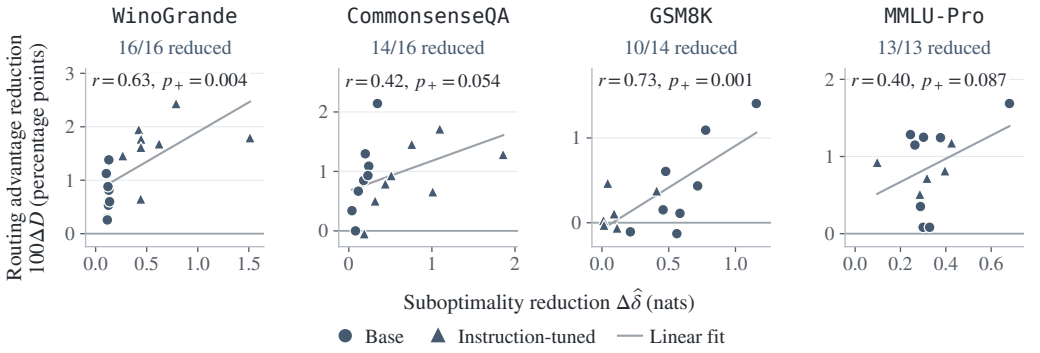

Figure 14: **Finetuning reduces routing advantage in 53 of 59 model-task pairs.** Each point compares one weak model before and after finetuning; both axes show released minus finetuned values. $\Delta\widehat{\delta}$ is the reduction in answer log loss. $100\Delta D$ is the reduction in cascading advantage. Points above zero have lower routing advantage after finetuning; counts report these pairs. Shapes identify the release type; lines are least-squares fits; r is Pearson correlation and p_+ its one-sided p-value for positive correlation.

C.5 How Strong-Model Errors Affect Routing Gains

Two contributions. By Equation 9, the routing advantage at each escalation rate is $\mathbb{E}[u\epsilon_1(X)] - \mathbb{E}[u\epsilon_2(X)]$ with $u = B_{g,\rho} - B_{f,\rho}$. The first term is the *weak-error term*: weak errors escalated by predictor routing minus those escalated by self-confidence routing. The second is the *strong-error term*: strong errors encountered by predictor routing minus those encountered by self-confidence routing. We estimate both on the test questions by replacing ϵ_1 and ϵ_2 with the observed error indicators $1 - Z_{1,i}$ and $1 - Z_{2,i}$, averaging over tie-breaking as in Equation 39, and integrating over escalation rates as for D. Table 9 reports both terms.

Task	Models	$100D$	Weak-error term	Strong-error term
`WinoGrande`	Released	1.38	1.80	0.42
	Finetuned	0.18	0.21	0.03
`CommonsenseQA`	Released	1.09	1.26	0.17
	Finetuned	0.18	0.17	-0.01
`GSM8K`	Released	0.52	0.59	0.07
	Finetuned	0.20	0.24	0.03
`MMLU-Pro`	Released	0.94	1.75	0.82
	Finetuned	0.07	0.94	0.87

Table 9: **Routing advantage split into two contributions.** Means over retained models with 4B predictors, in percentage points averaged over escalation rates. Both error terms are differences from self-confidence routing; $100D$ equals the weak-error term minus the strong-error term, up to rounding.

Strong errors usually reduce the gain. Predictor routing captures more weak-model errors, but the strong model also answers some of the escalated questions incorrectly. Averaged over escalation rates, the strong-error term is nonnegative for 91 of 118 retained model versions: in these cases, strong errors offset some of the gain from capturing additional weak errors. A bootstrap with 500 resamples of test questions gives a 95% interval entirely above zero in 40 cases and entirely below zero in one. The effect is largest on `MMLU-Pro`, where the strong model is 87.6% accurate. For finetuned weak models on this task, capturing additional weak errors contributes 0.94 percentage points, but strong errors subtract 0.87 points, leaving a routing advantage of 0.07 points. On `GSM8K`, the strong model is 97.9% accurate, so strong errors have less effect.

Connection to the bound. Theorem 2 bounds the additional weak errors captured in terms of suboptimality and predictor capacity. At any escalation rate where the strong-error term is nonnegative, Corollary 2 carries this bound over to the routing advantage. The 91 cases above satisfy the corresponding comparison after averaging over escalation rates. Empirically, suboptimality is at least as strongly correlated with the weak-error term as with the routing advantage D. The correlations are 0.80, 0.72, 0.91, and 0.86 in task order, compared with 0.79, 0.66, 0.91, and 0.82 for D. This is consistent with strong errors adding variation to the routing advantage in Figure 4. Avoiding strong errors can also increase the routing advantage, but the largest observed contribution is only 0.12 percentage points. Without the sign condition, the strong-error contribution to the gain at rate ρ is at most $\min\{\mathrm{Err}(f_2), \rho, 1 - \rho\}$. The worst-case allowances averaged over escalation rates are much larger than the measured contributions: 10.6, 8.0, 2.1, and 10.8 points in task order.