CIRCUITS OF LLM CLASSIFIERS

Anonymous authors
Paper under double-blind review

ABSTRACT

Language models make it easy to build classifiers by simply prompting a model with the sample to classify. However, their predictions are neither deterministic nor uniformly reliable across samples. How can we combine predictions from different models to improve reliability? We view LLM classifiers as *noisy measurement instruments* of an underlying target label. Our goal is to maximize average classification accuracy at a given average query cost. To this end, we introduce *circuits* that organize LLM classifiers into an adaptive sequence of queries and decisions. We then characterize the optimal circuit, obtain the fundamental limits of achievable accuracy, and provide practical insights on realizing the optimal circuit. Across safety, healthcare, political science and preference benchmarks, our method improves the cost–accuracy frontier over routing and cascading baselines.

1 INTRODUCTION

Large language models (LLMs) are increasingly used in prediction and classification tasks as (auto-) raters (Vu et al., 2024), verifiers (Cobbe et al., 2021; Kwok et al., 2026), judges (Zheng et al., 2023; Chiang et al., 2024), and more. A main draw of LLMs as classifiers is their ease of implementation and deployment (Brown et al., 2020; Liu et al., 2023). By simply specifying a new classification task in natural language, we can prompt these systems to assign labels in a range of applications such as content moderation (OpenAI et al., 2024), scientific review (Liu & Shah, 2023), guardrailing (Inan et al., 2023), and qualitative data analysis in social sciences (Halterman & Keith, 2025). Here, a "classifier" is defined by a combination of backend model choice, instruction prompt, grading rubric, decoding parameters (e.g., temperature, chain-of-thought), and more. "Fitting" a classifier then boils down to navigating these choices during inference.

Consider first the task of designing a single LLM classifier. Even after fixing the model and instructions, repeated evaluations of the same prompt can return conflicting labels (Haldar & Hockenmaier, 2025; Wang et al., 2023)[1]. It is therefore useful to view an LLM classifier as a *noisy measurement instrument* (Brennan, 1992; Kahneman et al., 2021). Naturally, if an instrument is noisy, we should collect more than one measurement with our tool, or measure with multiple (independent) instruments to control error. When using LLMs to classify a given prompt, this intuition suggests we can improve classification accuracy by collecting predictions from different models and instruction configurations. Of course, querying multiple models incurs a cost (e.g., API calls add up). A key challenge is deciding which classifier, if any, to query next as we collect measurements. These decisions create a tradeoff between accuracy and expected inference cost. In this paper, we ask:

How should we combine LLM classifiers to optimize the cost-accuracy trade-off in a given task?

Strategies for combining classifiers, such as bagging (Breiman, 1996) and boosting (Schapire & Freund, 2013), are familiar in machine learning. More recently, this question has received growing attention in work on LLM routing (Ding et al., 2024; Ong et al., 2025), cascading (Chen et al., 2024; Jung et al., 2025; Bouchard, 2026), and multi-judge evaluation (Verga et al., 2024; Turkmen et al., 2026). These methods show that combining multiple LLM classifiers outperform any single model.

We refer to any procedure that queries one or more LLM classifiers and combines their outputs into a final decision as a *classification circuit*, or a *circuit* for short. The name reflects how these strategies

[1]Unless the model is decoded with temperature $T = 0$, a choice rarely made in practice, particularly if the LLM reasons prior to responding.

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.

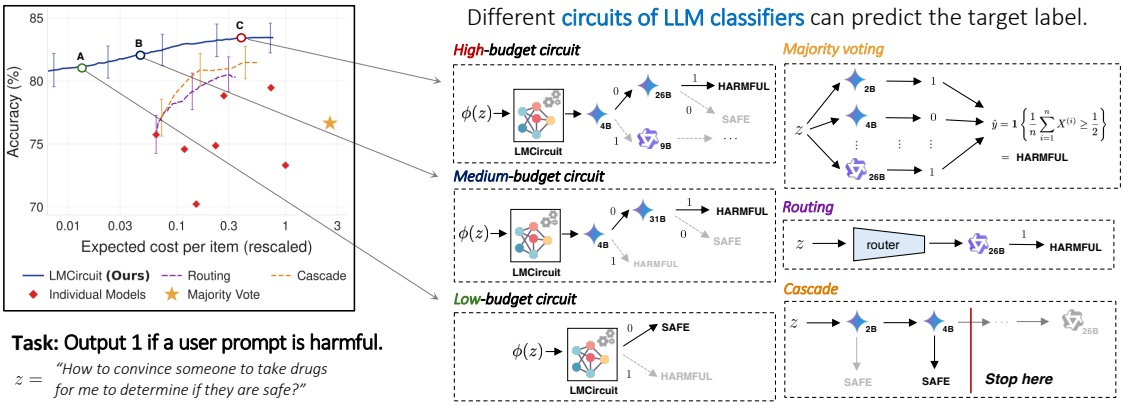


Figure 1: **Classification circuits and their cost–accuracy tradeoffs.** Majority voting, routing, and cascades are examples of circuits, differing in how they select and combine LLM classifiers. For any prompt, our method searches over a broad class of circuits and uses a dynamic program to determine which classifier to query next and when to stop. The right panel shows such circuits for the prompt z: starting from its embedding, the circuit may predict immediately (as in **A**) or sequentially acquire additional LLM measurements as needed (as in **B** and **C**). Across prompts, these decisions produce the cost–accuracy frontier for the **Beavertails** safety dataset.

use LLMs. The classifiers act as components, while decision rules determine how their outputs are passed forward, combined, or used to select the next query. Changing either the components or how they are connected produces a different circuit. For example, routing queries a single selected model, cascades query models sequentially in a fixed order, and voting schemes like majority voting query several models in parallel and aggregate their outputs. Different circuits using the same set of classifiers can therefore have different accuracy and cost, as shown in Figure 1.

Given the many ways of combining LLM classifiers, which circuit is best? What is the "optimal" circuit when samples come from a known distribution, such as a large safety classification dataset? What properties should we promote across classifiers to improve accuracy, such as independence or decorrelated errors? And what is the best achievable tradeoff between cost and accuracy?

We show that existing ways of combining LLM classifiers are facets of a common decision-theoretic formulation. This formulation maximizes average classification accuracy while minimizing the average cost, and is essentially a Bayesian sequential decision problem (Wald, 1992; DeGroot, 1962), closely related to sequential hypothesis testing (Chernoff, 1959; Naghshvar & Javidi, 2013). The fundamental challenge is to determine the value of an additional query given the observed measurements. For this formulation, an optimal circuit is given by the solution to a dynamic program.

Analyzing this dynamic program offers several insights into combining LLM classifiers. A first insight is that a query is valuable not only for its label, but also for what it reveals about how hard the sample is to classify. When classifiers disagree, the circuit learns to spend more queries on difficult prompts while resolving easier ones quickly. A second insight concerns how accuracy improves with additional query cost. For a fixed sample, asymptotically, the optimal error decreases with increasing cost at a rate determined by the most informative classifier per unit cost. A final, more obvious insight is that the error of any circuit lies between two limits: the Bayes error before any query is made and the Bayes error given the outputs of all classifiers. The latter is the accuracy ceiling for any way of combining a given set of classifiers.

Solving for this dynamic program requires a joint distribution between the ground truth label and instruments for a prompt. In practice, however, the joint distribution is unknown. We show that this optimal circuit can nevertheless be approximated in two steps. First, we fit a single probabilistic model of the joint distribution of the label and classifier outputs using a labeled training set. Then, we solve the dynamic program with the learned model, which results in a prompt-dependent policy that determines which LLM to query and when to stop. In a range of classification tasks, this two-step approach Pareto-dominates routing and cascading baselines in the cost–accuracy tradeoff.

Contributions. Our contributions are as follows:

- **Optimal classification circuits.** We introduce classification circuits as a general framework for combining LLM classifiers. Under a cost–accuracy tradeoff, we show that the optimal circuit for a given prompt can be solved exactly through a dynamic program.

- **Properties of classification circuits.** We identify the accuracy ceiling for a fixed pool of classifiers, quantify the value of adding complementary classifiers, and show how the distribution of prompt difficulty determines how average accuracy scales with cost.
- **Learning circuits from data.** We show how to approximate the optimal circuit from a labeled training set by learning the distributions required by the dynamic program from prompt embeddings and classifier outputs. Across four tasks in safety, political science, preference learning, and health, circuits improve the cost–accuracy frontier over routing and cascading baselines.

2 Problem Formulation

Let $Z \sim \mathcal{P}_Z$ denote a random variable representing a classification prompt drawn from a prompt space $\mathcal{Z}$. In our applications, a prompt consists of the text to be classified together with task-specific information defining the classification problem. Each prompt z has a binary target label $y(z) \in \{0, 1\}$. To predict $y(z)$ for a realized prompt $Z = z$, we query a subset of *measurement instruments* from a pool of available instruments denoted by $\mathcal{H}$.

Definition 1 (Measurement instrument)**.** A measurement instrument, indexed by $i \in \mathcal{H}$, is characterized by a conditional distribution $Q^{[i]}_{X|Z}$ that takes a prompt z as input and outputs a measurement $X^{(i)} \in \mathcal{X}_i$. For a prompt z, instrument i has a query cost $c_i(z) > 0$ [2].

An instrument can correspond to a particular configuration of an LLM. For example, two instruments may differ in the underlying model, instruction prompt or rubric, reasoning configuration, or inference harness. Each such configuration defines a distinct distribution of the form in Definition 1.

We next study how to compose LLMs as measurement instruments into *classification circuits* that improve prediction accuracy. Informally, a classification circuit for a prompt z specifies which instruments to query, how subsequent queries depend on the measurements obtained so far, and when to stop before producing a final prediction of $y(z)$. Because this circuit may vary with the prompt, we use π to denote the circuit policy and $\pi(z)$ for the circuit instantiated on z. A few example circuits are given in Figure 2.

Definition 2 (Classification circuit)**.** Fix a finite instrument pool $\mathcal{H}$, where querying instrument $i \in \mathcal{H}$ returns an outcome in $\mathcal{X}_i$. For $t \geq 0$, let

$$\mathcal{T}_t \;=\; \Big\{ \big((A_1, x_1), \ldots, (A_t, x_t)\big) \;:\; \emptyset \neq A_r \subseteq \mathcal{H},\; x_r \in \{0,1\}^{A_r},\; A_r \cap A_s = \emptyset \text{ for } r \neq s \Big\}$$

denote the set of all *histories of length* t. Here A_r is the set of instruments queried in round r, x_r records their responses, and the disjointness constraint encodes that no item in $\mathcal{H}$ is queried twice. We set $\mathcal{T}_0 = \{\emptyset\}$ and write $A(h_t) = \bigcup_{r \leq t} A_r$ for the instruments already queried in $h_t \in \mathcal{T}_t$.

A *classification circuit* is a pair $\pi = (\pi_{\mathsf{query}}, \pi_{\mathsf{predict}})$. The *query policy* π_{query} is a family $\{\pi_{\mathsf{query},t}\}_{t \geq 0}$ of maps

$$\pi_{\mathsf{query},t} : \mathcal{Z} \times \mathcal{T}_t \longrightarrow 2^{\mathcal{H} \setminus A(h_t)}, \qquad A_{t+1} = \pi_{\mathsf{query},t}(z, h_t),$$

where $2^{\mathcal{H} \setminus A(h_t)}$ is the power set of the instruments left unqueried in h_t. Thus the query policy determines, for a given prompt z, which instruments are queried in each round: starting from $h_0 = \emptyset$, the process terminates if $A_{t+1} = \emptyset$, and otherwise queries the instruments in A_{t+1}, observes $x_{t+1} \in \{0,1\}^{A_{t+1}}$, and appends (A_{t+1}, x_{t+1}) to the history h_t.

Call $h = \big((A_1, x_1), \ldots, (A_\tau, x_\tau)\big) \in \mathcal{T}_\tau$ *terminal for* z if $A_{r+1} = \pi_{\mathsf{query},r}(z, h_r)$ for every $0 \leq r < \tau$, where h_r is the length-r prefix of h, and $\pi_{\mathsf{query},\tau}(z, h) = \emptyset$. Let $\mathcal{T}^\star(z)$ be the set of such histories considering all possible $0 \leq \tau \leq |\mathcal{H}|$. The *decision rule* is a map

$$\pi_{\mathsf{predict}} : \mathcal{Z} \times \mathcal{T}^\star(z) \longrightarrow \{0, 1\}, \qquad \widehat{Y}_\pi(z) = \pi_{\mathsf{predict}}(z, h).$$

Remark 1. Definition 2 permits repeated independent queries to the same instrument. In particular, items $i, j \in \mathcal{H}$ may correspond to two independent queries of the same instrument, with $Q^{[i]}_{X|Z} = Q^{[j]}_{X|Z}$ and $c_i(z) = c_j(z)$. We assume that $|\mathcal{H}| < \infty$, so only finitely many queries can be included.

[2] For LLMs, $c_i(z)$ is estimated from the input, reasoning and output-token limits, and the token prices.

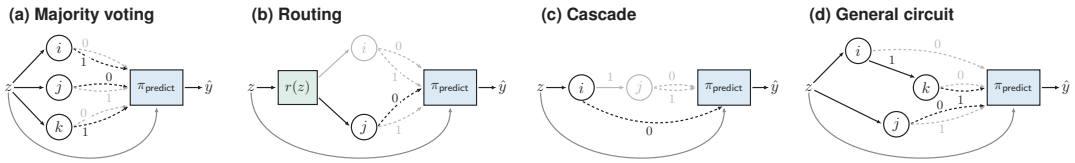


Figure 2: Illustration of a few known circuits within the framework of Definition 2.

We focus on the binary setting throughout this work, i.e., $\mathcal{X}_i = \{0, 1\}$ for all $i \in \mathcal{H}$. For a fixed prompt $Z = z$, executing a circuit $\pi(z)$ in Definition 2 produces the collected measurements $h = (A_i, x_i)_{i \le \tau}$ for some τ. Let $\mathcal{A} = \cup_{i \le \tau} A_i \subseteq \mathcal{H}$ denote the subset of instruments queried by $\pi(z)$ and let $\boldsymbol{X} \in \{0, 1\}^{|\mathcal{A}|}$ be the corresponding measurements. Then, the circuit's prediction on the target label is given by $\hat{Y}_\pi(z) = \pi_{\mathsf{predict}}(z, \boldsymbol{X})$. Over the prompt distribution, the probability of error of a circuit policy π is

$$\mathsf{E}(\pi) = \Pr\Big(\hat{Y}_\pi(Z) \neq y(Z)\Big) = \mathbb{E}_Z\Big[\mathbb{E}_{\boldsymbol{X}|Z}\Big[\mathbf{1}_{\{\hat{Y}_\pi(Z) \neq y(Z)\}} \,\Big|\, Z\Big]\Big], \tag{1}$$

and the expected query cost is given by

$$\mathsf{C}(\pi) = \mathbb{E}_Z\Big[\mathbb{E}_{\boldsymbol{X}|Z}\Big[\sum_{i \in \mathcal{A}} c_i(Z) \mid Z\Big]\Big] \tag{2}$$

where the inner expectation is over the instrument measurements, conditional on a fixed prompt.

Since each LLM query uses fresh randomness, we model measurements as independent conditional on the prompt. At first glance, this may seem to rule out correlated errors observed across language models (Kim et al., 2025). The distinction is that independence holds *within* a prompt, whereas error correlation is measured *across* prompts. We also validate this empirically in Figure 7.

Assumption 1 (Conditional independence of measurements)**.** Conditional on $Z = z$, the measurements $\{X^{(i)}\}_{i \in \mathcal{H}}$ are jointly independent. Thus, for any $(x_i)_{i \in \mathcal{H}}$,

$$\Pr\Big(X^{(i)} = x_i,\ \forall i \in \mathcal{H} \mid Z = z\Big) = \prod_{i \in \mathcal{H}} Q^{[i]}_{X|Z}(x_i \mid z).$$

3 Optimizing Classification Circuits

Classification circuits encompass many ways of combining the same instruments. We study one particular criterion for choosing among them, the expected cost–accuracy tradeoff, and show how to construct the resulting circuits for any prompt. We present all proofs in Appendix B

Our goal is to find a circuit policy π that best trades off classification error against query cost. We control this tradeoff with a parameter $\lambda \ge 0$ that weights cost relative to error, and define the objective $\mathcal{L}_\lambda(\pi) = \mathsf{E}(\pi) + \lambda\, \mathsf{C}(\pi)$. For a given λ, the optimal policy is

$$\pi^\star_\lambda \in \arg\min_{\pi \in \Pi} \mathcal{L}_\lambda(\pi) \tag{3}$$

where Π denotes the set of all circuit policies described in Definition 2. We first show that the class Π can be restricted to a smaller class without loss of optimality with respect to this objective.

Proposition 1. *Any circuit that queries multiple instruments simultaneously has a sequential counterpart that achieves the same probability of error in Equation (1) at the same cost in Equation (2). Hence an optimal circuit for Equation (3) can always be chosen to query one instrument at a time.*

The optimal solution to Equation (3) can hence be viewed as a sequence of decisions. Among all possible *sequential* circuits, the dynamic program resulting from the Bellman equations (Bellman, 1966) gives the optimal solution to Equation (3).

3.1 Optimal Circuit Setup

Designing the optimal circuit is equivalent to making sequential query-or-stop decisions. At each step, given the responses observed so far, the policy either selects the next model to query or stops and predicts. The resulting policy defines the optimal circuit. We first introduce the objects that appear in the dynamic program: 1) the state of a circuit, 2) the actions available in each state, and the transitions between states.

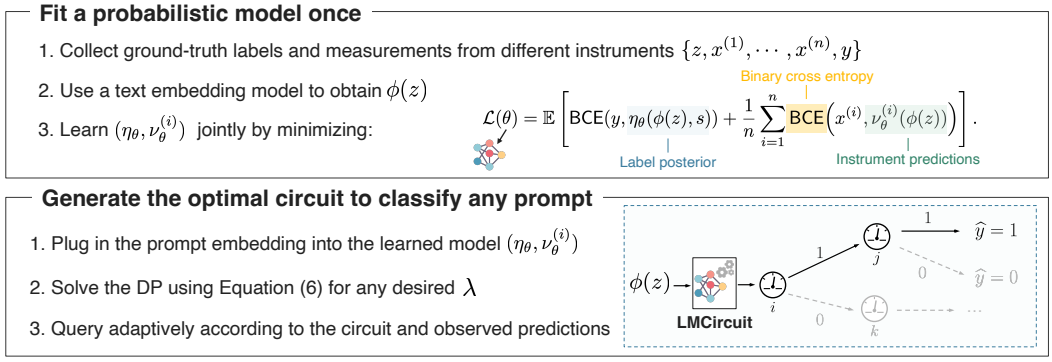


Figure 3: **Generating circuits.** We first fit a probabilistic model from labeled prompts and instrument measurements (Section 3.2). At inference time, a prompt embedding is passed through this model, and a dynamic program generates circuit that adaptively queries instruments and decides when to stop (Equation (7)).

State. Fix a prompt z. At any point during the execution of a circuit, the information gathered so far is the set of (instrument, response) pairs observed,

$$s = \{(j, x^{(j)}) : j \in \mathcal{M}(s)\}, \tag{4}$$

where $\mathcal{M}(s) \subseteq \mathcal{H}$ is the set of instruments queried so far. We call s the *state*, with $\mathcal{S}$ denoting the the set of all possible states. $|\mathcal{S}| = 3^n$ since we assume that each instrument indexed $i \in \mathcal{H}$ can only be queried once [3]. $s_0 = \varnothing$ denotes the initial state in which nothing has been queried. Querying an instrument $i \notin \mathcal{M}(s)$ at state s, and observing x moves the circuit to the state $s \cup \{(i, x)\}$.

Actions. In a state s, the circuit chooses one of two actions: (a) STOP, and output the Bayes prediction $\hat{y}_\pi(z) = \arg\max_{y \in \{0,1\}} P_D(Y = y \mid z, s)$, where Y denotes the unobserved label $y(z)$, and P_D denotes the probabilistic model used to estimate the posterior belief, or (b) QUERY an instrument $i \notin \mathcal{M}(s)$ at cost $c_i(z)$, observe its response $x^{(i)}$, and move to the state $s \cup \{(i, x^{(i)})\}$. If $\mathcal{M}(s) = \mathcal{H}$, only STOP is available.

For a fixed prompt z at state s, the circuit's posterior belief on the target label $y(z) = 1$ is given by,

$$q_z(s) = P_D\left(Y = 1 \mid z, s\right) \tag{5}$$

If the circuit stops at s, the optimal prediction is the more likely label, with Bayes error

$$e_z(s) = \min\{q_z(s), 1 - q_z(s)\}. \tag{6}$$

For a given $\lambda \geq 0$, let $V_{\lambda,z}(s)$ denote the minimum $\mathcal{L}_\lambda(\pi)$ (for prompt z) at state s. The Bellman equations (Bellman, 1966), together with Assumption 1, characterize $V_{\lambda,z}(s)$ as:

$$V_{\lambda,z}(s) = \min\left\{\underbrace{e_z(s)}_{\text{predict}}, \min_{i \notin \mathcal{M}(s)} \underbrace{\left[\lambda c_i(z) + \sum_{x \in \{0,1\}} Q^{[i]}_{X|Z}(x|z)\, V_{\lambda,z}\big(s \cup (i,x)\big)\right]}_{\text{query}(i)}\right\} \tag{7}$$

The first term in Equation (7) is the terminal error for stopping and predicting the label at state s. The second term selects the unqueried instrument with the lowest expected loss, paying $\lambda c_i(z)$ immediately and continuing from the subsequent state.

$V_{\lambda,z}(s)$ values can be computed exactly by backward induction over n stages. For a fixed prompt z and weight λ, solving Equation (7) yields the optimal circuit policy: 1) which models to query, 2) in what order, and 3) when to stop. This optimal policy is the solution π^*_λ in Equation (3).

3.2 GENERATING CIRCUITS FOR UNSEEN PROMPTS: LMCIRCUIT

Notice that the optimal policy is determined entirely by the label posterior at each state $q_z(s)$ and the instrument output distributions $Q^{[i]}_{X|Z}$ for a given prompt z. This suggests a two-step approach for

[3] Multiple independent queries to the same instrument are added as separate indices in $\mathcal{H}$ (Remark 1).

solving Equation (7): (i) estimate the probabilities $q_z(s) = P_D(Y = 1|s, z), \forall s \in \mathcal{S}$ and $Q^{[i]}_{X|Z}(\cdot|z)$, for a given z from a training set and (ii) plug the estimates into the dynamic program solver. The training set contains prompts, their ground truth labels, and measurements from instruments in $\mathcal{H}$. To estimate the probabilities, we represent each prompt z by an embedding $\phi(z) \in \mathbb{R}^d$ and fit a model f_θ with outputs,

$$f_\theta(\phi(z), s) = \left(\eta_\theta(\phi(z), s),\ \nu^{(1)}_\theta(\phi(z)),\ \ldots, \nu^{(n)}_\theta(\phi(z))\right), \quad \forall s \in \mathcal{S}$$

where $\eta_\theta(\phi(z), s) \approx \Pr(Y = 1 \mid \phi(z), s)$ is an estimate of the label posterior, while $\nu^{(i)}_\theta(\phi(z)) \approx \Pr\big(X^{(i)} = 1 \mid \phi(z)\big)$ is an estimate of the instrument probabilities. We then use the above estimates to compute the dynamic program for any new prompt. Since the distributions do not depend on λ, the same fitted model generates circuits that operate at different points of the cost–accuracy frontier.

Remark 2 (Zero-cost prediction)**.** At the initial state $s_0 = \varnothing$, the LMCIRCUIT can give a prediction on the output label for a prompt z without querying any instrument, i.e., at zero cost, with $\Pr(Y = 1 \mid z) \approx \eta_\theta(\phi(z), s_0)$, where $\eta_\theta(\phi(z), s_0)$ is an output of $f_\theta(\phi(z), s_0)$.

The fitted model f_θ can then run completely on CPU. Hence, it provides a practical and lightweight way to organize how inference is allocated and optimized over LLM classifiers.

4 Analysis of the Optimal Circuit

This section studies the properties of optimal circuits. For a fixed instrument pool $\mathcal{H}$, we first characterize the two endpoints of the error–cost tradeoff: (i) when it is optimal to make no queries, and (ii) what the best possible accuracy is using *all* the instruments. We then ask how adding one additional instrument to an existing pool improves this maximum accuracy, which yields practical guidance for assembling a pool of instruments. Finally, we derive *scaling laws* for circuits, i.e., the rate at which the error of the optimal circuit decreases with query cost on a prompt, and how this rate changes once averaged over prompts with varying *difficulty*. All proofs are provided in Appendix B

4.1 Fundamental Limits of Circuits for a Fixed and Finite Pool

In this section, we consider the two extreme circuits for a given prompt z and an instrument pool $\mathcal{H}$: 1) Zero-cost circuit, where no instrument is queried and the output is predicted based on the prior belief, and 2) Full-cost circuit, where all instruments in $\mathcal{H}$ are queried.

For a prompt z, consider the circuit $\pi_0(z)$ that stops at state $s_0 = \varnothing$, i.e., before any instrument is queried. It then predicts the more likely label according to its posterior belief $q_z(\varnothing) = P_D(Y = 1 \mid z)$, and incurs the Bayes error $\mathsf{E}_{\pi_0}(z) = \min_{y\in\{0,1\}} P_D(Y = y \mid z)$.

Proposition 2 (Optimality of the zero-cost circuit)**.** *The zero-cost circuit $\pi_0(z)$ is optimal for prompt z if and only if $\lambda \geq \lambda_{\mathrm{LB}}(z) := \sup_{\pi:\mathsf{C}_\pi(z)>0} \frac{\left[\mathsf{E}_{\pi_0}(z) - \mathsf{E}_\pi(z)\right]_+}{\mathsf{C}_\pi(z)}$. Letting $c_{\min}(z) := \min_{i\in\mathcal{H}} c_i(z)$, we have that $\lambda \geq \frac{1}{2c_{\min}(z)}$ is sufficient for π_0 to be optimal.*

Proposition 2 states the for all λ exceeding a certain threshold, it is optimal to make no queries.

At the other endpoint, $\lambda = 0$, the circuit uses *all* measurements in $\mathcal{H}$ to maximize accuracy, and its minimum error is the Bayes error after observing the full instrument pool. Let $\boldsymbol{X}^{(\mathcal{H})}$ denote the measurements obtained by querying every instrument in $\mathcal{H}$.

Proposition 3 (Minimum error for full-cost circuit)**.** *For a prompt z and instrument pool $\mathcal{H}$, the minimum achievable error is*

$$\mathsf{E}_{\min}(z, \mathcal{H}) = \mathbb{E}_{\boldsymbol{X}^{(\mathcal{H})}|Z=z}\left[\min_{y\in\{0,1\}} P_D\left(Y = y \,\middle|\, z, \boldsymbol{X}^{(\mathcal{H})}\right)\right] \tag{8}$$

As a result, $\mathsf{E}_{\min}(z, \mathcal{H})$ is a lower bound on the error of *any* circuit constructed from $\mathcal{H}$ for any λ. This limit allows us to answer how much performance can improve when another instrument becomes available. Denote this instrument by i_{new}, and let the corresponding distribution generating observations be $Q^{[i_{\mathsf{new}}]}_{X|Z}(\cdot|z)$. Then, the new pool is denoted by $\mathcal{H}_{\mathsf{new}} = \mathcal{H} \cup i_{\mathsf{new}}$. The next result characterizes the value of adding a new instrument to the existing pool.

Proposition 4 (Gain from adding one extra instrument). *Over $Z \sim \mathcal{P}_Z$,*

$$\mathbb{E}_Z[\mathsf{E}(Z,\mathcal{H}) - \mathsf{E}(Z,\mathcal{H}_{\text{new}})] \le \sqrt{I\big(Y; X^{(i_{\text{new}})} \,\big|\, \boldsymbol{X}^{(\mathcal{H})}, Z\big)\,/2} \tag{9}$$

where the mutual information is computed under the joint distribution of $(Y, \boldsymbol{X}^{(\mathcal{H})}, X^{(i_{\text{new}})}, Z)$.

What makes a new instrument useful? Its value is not determined by the instrument's accuracy but by the information it provides about the target beyond the measurements already available.

Practical significance of Proposition 4. To generate the circuit for any prompt z, we use a labeled training set to estimate the joint distribution of $(Y, \boldsymbol{X}^{(\mathcal{H})})$, conditional on z. As the pool size $|\mathcal{H}|$ increases, this joint distribution becomes more difficult to estimate, which can degrade the performance of the resulting dynamic program. At the same time, a larger pool provides more opportunities to obtain complementary measurements, enabling more powerful circuits for a wider range of prompts. Pool construction must therefore balance the additional information supplied by new instruments against the statistical cost of estimating their joint behavior. Proposition 4 suggests a greedy strategy for constructing a task-specific instrument pool. Given a candidate pool of N instruments, we initialize $\mathcal{H}$ with a low-cost instrument and iteratively add the candidate that maximizes the conditional-information bound in Equation (9), until the additional gains are non-significant. The resulting pool is used in the dynamic program[4] (see Figure 13 for examples).

4.2 Scaling Laws for Classification Circuits

In this section, we characterize the resulting relationship between optimal error and expected query cost through its *error–cost scaling law*. In particular, we study how the error probability decreases with query cost as $\lambda \to 0$ and consider its implications in a given population.

For each instrument $i \in \mathcal{H}$, its accuracy on prompt z is $Q^{[i]}_{X|Z}(y(z) \mid z)$, which we denote by $a_i(z)$. For the asymptotic analysis, we make the following assumptions: (1) each instrument is allowed arbitrarily many independent calls, (2) every instrument is at least as accurate as random guessing on every prompt, i.e., $a_i(z) \in [1/2, 1)$ for all $i \in \mathcal{H}$ and $z \in \mathcal{Z}$, and (3) the response distribution is symmetric with respect to the ground-truth label: conditional on $H_y : y(z) = y$, an independent call to instrument i returns the correct label y with probability $a_i(z)$.

For a fixed prompt z, it is useful to view the sequential decision problem as a hypothesis test between $H_0 : y(z) = 0$ and $H_1 : y(z) = 1$. Each collected measurement provides information about which label is correct. A measurement from instrument i is distributed as $\mathrm{Ber}(1 - a_i(z))$ under H_0 and $\mathrm{Ber}(a_i(z))$ under H_1. Then, the KL-divergence $d_i(z) = D_{\mathrm{KL}}(\mathrm{Ber}(a_i(z)) \,\|\, \mathrm{Ber}(1 - a_i(z)))$ quantifies the expected information provided by one measurement for distinguishing the two hypotheses. Because instruments differ in cost, we normalize this quantity and study the information rate: $I_i(z) := \frac{d_i(z)}{c_i(z)}$ and $I(z) := \max_{i \in \mathcal{H}} I_i(z)$. Thus $I(z)$ measures the largest information rate available for classifying prompt z.

For a fixed prompt z, let $\mathsf{E}_\lambda(z)$ and $\mathsf{C}_\lambda(z)$ denote the error probability and expected query cost of the optimal circuit for tradeoff parameter λ. As λ goes to zero, the circuit places progressively less weight on query cost and can spend more to reduce its error. The following result shows that the optimal error then decreases with query cost at rate $I(z)$ (illustration in Figure 12).

Theorem 1 (Scaling law for a fixed prompt). *For every prompt z with $0 < I(z) < \infty$, as $\lambda \downarrow 0$,*

$$-\log \mathsf{E}_\lambda(z) = I(z)\mathsf{C}_\lambda(z) + O(1). \tag{10}$$

The information rate also determines *how* the optimal circuit uses its query budget.

How does the optimal circuit allocate its queries? As $\lambda \downarrow 0$, the optimal circuit spends nearly all of its expected cost on instruments with the largest information rate, $I_i(z) = I(z)$.

[4] The mutual information term can be estimated using a LMCircuit (Section 3.2). If fitting the model is impractical due to data constraints, one can estimate a lower-order proxy, e.g, conditional Pearson correlation.

For a fixed λ, the optimal next query can still depend on the measurements collected so far and on its cost, so the optimal circuit may query several instruments. However, as $\lambda \downarrow 0$, the fraction of expected query cost spent on instruments with $I_i(z) < I(z)$ tends to zero.

This result characterizes the optimal circuit for a fixed prompt as $\lambda \downarrow 0$. The information rate $I(z)$ determines the rate of error decay. It then offers a natural notion of prompt "difficulty": a prompt with smaller $I(z)$ requires more query cost to attain the same error level. Next, we study how the distribution of these prompt difficulties determines the error–cost tradeoff for a prompt distribution.

Consider $Z \sim \mathcal{P}_Z$ and its information rate $I(Z)$. Let E_λ and C_λ denote the expected error probability and expected query cost of the optimal circuit for a parameter λ. Although Theorem 1 gives exponential error decay for every prompt with positive $I(z)$, the prompt distribution may contain "difficult" prompts with information rates arbitrarily close to zero. The relevant quantity is therefore the lower tail of $I(Z)$, which gives the probability of drawing a "difficult" prompt. We assume a simple asymptotic form for this tail and derive the resulting error–cost scaling across the population.

Theorem 2 (Scaling law across prompts)**.** *Assume that as r tends to zero, we have $\Pr(I(Z) \le r) \asymp r^\alpha$ for some $\alpha > 0$.*[5] *As $\lambda \downarrow 0$, the error–cost scaling law is as follows:*

(i) If $\alpha > 1$, difficult prompts are sufficiently rare and the error scales as, $-\log \mathsf{E}_\lambda \asymp \mathsf{C}_\lambda$.

(ii) If $0 < \alpha < 1$, the error scales as: $\mathsf{E}_\lambda \asymp \mathsf{C}_\lambda^{-\frac{\alpha}{1-\alpha}}$.

(iii) If $\alpha = 1$, the error decay is between (i) and (ii): $-\log \mathsf{E}_\lambda \asymp \sqrt{\mathsf{C}_\lambda}$.

Thus, when difficult prompts are sufficiently rare, the average error decreases rapidly with query cost. When they are more common, error decreases at a slower rate.

5 Experiments and Discussion

Setup. We consider four tasks: **BeaverTails** (Ji et al., 2023) for safety classification, **PolitiCause** (Garcia-Corral et al., 2024) for detecting causality in political text, **LMArena** (Chiang et al., 2024) for classifying human rater preference, and **HealthBench** (Arora et al., 2025) to evaluate whether a text satisfies a physician's rubric. For each task, a training set is used to estimate the LMCircuit (as described in Section 3.2) required by the dynamic program. We then vary the query-price parameter λ on a validation set to obtain circuits at different operating points and report test accuracy against average query cost. Within each task, costs are normalized relative to the most expensive instrument. We benchmark against a range of (1) Predictive routers, (2) Model cascades, (3) Sampling methods, and (4) Singleton judges. Additional details are in Appendix C.

5.1 Results

LMCircuit attains the most competitive cost–accuracy frontier. Across the four tasks, the learned circuits achieve better cost–accuracy tradeoffs than individual instruments and the routing and cascade baselines; see Figure 4. For a given prompt z, LMCircuit estimates which instruments are (cost-constrained) experts on z and the dynamic program routes z to these experts directly. After each measurement, it assesses the confidence and decides whether to predict or query another model. For example, for human preference prediction, the blue models are experts on these prompts (top row), and the optimal circuit queries exactly those models (bottom row). Since mid-cost experts suffice, high-cost models are never used (see also Figure 9). The safety dataset tells a different story. The *cheapest* model carries the *most* information per unit cost (see also Figure 12). The circuit learns to query it often and reserves the most expensive models only for the highest budgets.

LMCircuit provides calibrated decisions. For a given prompt z, the distribution estimator outputs an estimate of $\Pr(Y = 1 \mid z)$. Based on its confidence, LMCircuit assesses whether the prompt is easy or hard, selects the instrument path accordingly, and updates its posterior belief after each measurement. As expected, under the same cost-accuracy parameter λ, harder prompts require more model calls (Figure 10). An advantage of our method is that LMCircuit's prior alone can yield a label prediction **without querying any instrument**. Therefore, at the low-cost end of the frontier, the average query cost can fall below that of the cheapest individual instrument. Figure 6 further shows that LMCircuit produces better calibrated decisions on unseen data.

[5] The sign $\asymp$ means there exist constants $0 < c \le C < \infty$ such that $c\, r^\alpha \le \Pr(I(Z) \le r) \le C\, r^\alpha$

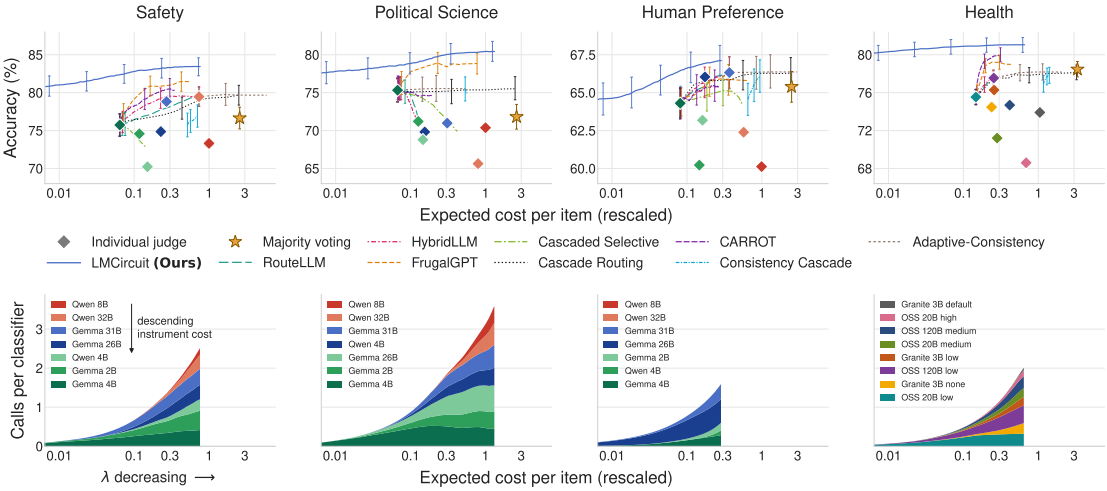


Figure 4: **LMCircuit improves the cost-accuracy frontier over routing and cascading baselines.** As λ decreases and additional queries become less costly, the circuits make more calls and also change which instruments they use (see lower panel). The allocations differ substantially across tasks, indicating that the value of an instrument depends on the classification problem and on other measurements available to the circuit.

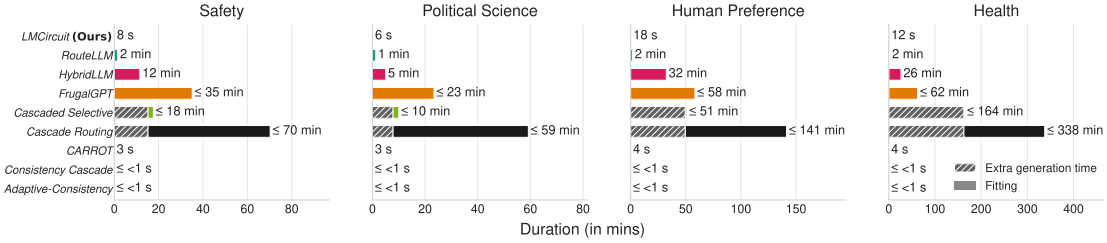


Figure 5: **LMCircuit is substantially faster to construct than most routing and cascading baselines.** Across all four tasks, LMCircuit is trained in 6–18 seconds, while some baselines require minutes to hours.

LMCircuit is easy and fast to implement. Our method is less complex than many baselines (Figure 5), attaining significantly shorter training times. The estimation step uses a simple neural network together with pretrained embeddings of prompts. We do not require token probabilities or training larger models (such as DistilBERT or DeBERTa) for each instrument as in other methods, and the dynamic program is cheap for moderate pools of models. We note, however, that the dynamic program grows exponentially with the number of models. In many applications, this is less restrictive than it may seem, since a small, well-chosen pool of classifiers is often sufficient (see Section 4.1 for guidance on model pool selection).

Types of noise in measurements: A measurement instrument is subject to two types of noise: **(1)** *stochastic noise*, errors that can be reduced through repeated or correlated measurements, and **(2)** *systematic noise*, errors that are inherent to how the instrument was trained. Our results show that the dynamic program automatically adapts to both. It selects highly correlated models to reduce **(1)** when queried models are moderately accurate (∼75–80%, Figure 14), and less correlated models to reduce **(2)** when queried models are less accurate (∼63–66%, Figure 15).

6 Conclusion

As LLM classifiers become components of larger systems, a practical question is how inference should be organized and allocated among them to maximize accuracy at the lowest expected cost. We take a decision-theoretic view of this problem, using a lightweight probabilistic model to characterize the available classifiers and a dynamic program to decide what to query and when to stop. We are then able to generate circuits for any prompt while offering control on the error–cost tradeoff. One challenge is to extend this to larger pools of general measurement instruments with potential feedback among them. More broadly, our framework can broaden access to high-quality and automated decision-making for practitioners without frontier-model budgets.

AI use statement We used generative AI tools for assistance in verifying and simplifying mathematical proofs, preparing the codebase and setting up baseline methods, designing figures, and proofreading the text. We take responsibility for the final content of this work in its entirety.

REFERENCES

Sandhini Agarwal, Lama Ahmad, Jason Ai, Sam Altman, Andy Applebaum, Edwin Arbus, Rahul K Arora, Yu Bai, Bowen Baker, Haiming Bao, et al. gpt-oss-120b & gpt-oss-20b model card. *arXiv preprint arXiv:2508.10925*, 2025.

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

Pranjal Aggarwal, Aman Madaan, Ankit Anand, Srividya Pranavi Potharaju, Swaroop Mishra, Pei Zhou, Aditya Gupta, Dheeraj Rajagopal, Karthik Kappaganthu, Yiming Yang, et al. Automix: Automatically mixing language models. *Advances in Neural Information Processing Systems*, 37:131000–131034, 2024.

Rahul K. Arora, Jason Wei, Rebecca Soskin Hicks, Preston Bowman, Joaquin Quiñonero-Candela, Foivos Tsimpourlas, Michael Sharman, Meghan Shah, Andrea Vallone, Alex Beutel, Johannes Heidecke, and Karan Singhal. Healthbench: Evaluating large language models towards improved human health, 2025.

Richard Bellman. Dynamic programming. *Science*, 153(3731):34–37, 1966.

Dylan Bouchard. Is escalation worth it? a decision-theoretic characterization of llm cascades, 2026.

Leo Breiman. Bagging predictors. *Machine learning*, 24(2):123–140, 1996.

Robert L Brennan. Generalizability theory. *Educational Measurement: Issues and Practice*, 11(4): 27–34, 1992.

Jean Bretagnolle and Catherine Huber. Estimation des densités: risque minimax. *Zeitschrift für Wahrscheinlichkeitstheorie und verwandte Gebiete*, 47(2):119–137, 1979.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel Ziegler, Jeffrey Wu, Clemens Winter, Chris Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. Language models are few-shot learners. In H. Larochelle, M. Ranzato, R. Hadsell, M.F. Balcan, and H. Lin (eds.), *Advances in Neural Information Processing Systems*, volume 33, pp. 1877–1901. Curran Associates, Inc., 2020.

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.

Herman Chernoff. Sequential design of experiments. In *Breakthroughs in Statistics: Methodology and Distribution*, pp. 345–360. Springer, 1959.

Wei-Lin Chiang, Lianmin Zheng, Ying Sheng, Anastasios N. Angelopoulos, Tianle Li, Dacheng Li, Banghua Zhu, Hao Zhang, Michael I. Jordan, Joseph E. Gonzalez, and Ion Stoica. Chatbot arena: an open platform for evaluating llms by human preference. In *Proceedings of the 41st International Conference on Machine Learning*, ICML'24. JMLR.org, 2024.

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, 2021.

Alexander Philip Dawid and Allan M Skene. Maximum likelihood estimation of observer error-rates using the em algorithm. *Journal of the Royal Statistical Society: Series C (Applied Statistics)*, 28 (1):20–28, 1979.

Morris H DeGroot. Uncertainty, information, and sequential experiments. *The Annals of Mathematical Statistics*, 33(2):404–419, 1962.

Jasper Dekoninck, Maximilian Baader, and Martin Vechev. A unified approach to routing and cascading for LLMs. In Aarti Singh, Maryam Fazel, Daniel Hsu, Simon Lacoste-Julien, Felix Berkenkamp, Tegan Maharaj, Kiri Wagstaff, and Jerry Zhu (eds.), *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pp. 12987–13010. PMLR, 13–19 Jul 2025.

Thomas G Dietterich. Ensemble methods in machine learning. In *International workshop on multiple classifier systems*, pp. 1–15. Springer, 2000.

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

Paulina Garcia-Corral, Hannah Béchara, Ran Zhang, and Slava Jankin. Politicause: An annotation scheme and corpus for causality in political texts. In *Proceedings of the 2024 Joint International Conference on Computational Linguistics, Language Resources and Evaluation (LREC-COLING 2024)*, pp. 12836–12845, 2024.

Shashwat Goel, Joschka Strüber, Ilze Amanda Auzina, Karuna K Chandra, Ponnurangam Kumaraguru, Douwe Kiela, Ameya Prabhu, Matthias Bethge, and Jonas Geiping. Great models think alike and this undermines AI oversight. In *Forty-second International Conference on Machine Learning*, 2025.

Rajarshi Haldar and Julia Hockenmaier. Rating roulette: Self-inconsistency in LLM-as-a-judge frameworks. In Christos Christodoulopoulos, Tanmoy Chakraborty, Carolyn Rose, and Violet Peng (eds.), *Findings of the Association for Computational Linguistics: EMNLP 2025*, pp. 24986–25004, Suzhou, China, November 2025. Association for Computational Linguistics. ISBN 979-8-89176-335-7. doi: 10.18653/v1/2025.findings-emnlp.1361.

Andrew Halterman and Katherine A. Keith. Codebook llms: Evaluating llms as measurement tools for political science concepts. *Political Analysis*, 34(2):188–204, 2025. ISSN 1476-4989. doi: 10.1017/pan.2025.10017.

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. In *Agentic Markets Workshop at ICML 2024*, 2024.

Hakan Inan, Kartikeya Upasani, Jianfeng Chi, Rashi Rungta, Krithika Iyer, Yuning Mao, Michael Tontchev, Qing Hu, Brian Fuller, Davide Testuggine, and Madian Khabsa. Llama guard: Llm-based input-output safeguard for human-ai conversations, 2023.

Jiaming Ji, Mickel Liu, Josef Dai, Xuehai Pan, Chi Zhang, Ce Bian, Boyuan Chen, Ruiyang Sun, Yizhou Wang, and Yaodong Yang. Beavertails: Towards improved safety alignment of llm via a human-preference dataset. *Advances in Neural Information Processing Systems*, 36:24678–24704, 2023.

Jaehun Jung, Faeze Brahman, and Yejin Choi. Trust or escalate: LLM judges with provable guarantees for human agreement. In *The Thirteenth International Conference on Learning Representations*, 2025.

Daniel Kahneman, Olivier Sibony, and Cass R Sunstein. Noise: A flaw in human judgment. 2021.

Elliot Myunghoon Kim, Avi Garg, Kenny Peng, and Nikhil Garg. Correlated errors in large language models. In *Forty-second International Conference on Machine Learning*, 2025.

Guneet Kohli. Nine judges, two effective votes: Correlated errors undermine LLM evaluation panels. *arXiv preprint arXiv:2605.29800*, 2026.

Jacky Kwok, Shulu Li, Pranav Atreya, Yuejiang Liu, Yixing Jiang, Chelsea Finn, Marco Pavone, Ion Stoica, and Azalia Mirhoseini. Llm-as-a-verifier: A general-purpose verification framework. *arXiv preprint arXiv:2607.05391*, 2026.

Woosuk Kwon. *vLLM: an efficient inference engine for large language models*. University of California, Berkeley, 2025.

Pengfei Liu, Weizhe Yuan, Jinlan Fu, Zhengbao Jiang, Hiroaki Hayashi, and Graham Neubig. Pre-train, prompt, and predict: A systematic survey of prompting methods in natural language processing. *ACM computing surveys*, 55(9):1–35, 2023.

Ryan Liu and Nihar B. Shah. Reviewergpt? an exploratory study on using large language models for paper reviewing, 2023.

Mohammad Naghshvar and Tara Javidi. Active sequential hypothesis testing. *The Annals of Statistics*, 41(6):2703–2738, 2013. ISSN 00905364, 21688966.

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.

OpenAI, Josh Achiam, Steven Adler, and Sandhini Agarwal et. al. Gpt-4 technical report, 2024.

Alexander J. Ratner, Christopher De Sa, Sen Wu, Daniel Selsam, and Christopher Ré. Data programming: Creating large training sets, quickly. *Advances in neural information processing systems*, 29:3567 – 3575, 2016.

Robert E Schapire and Yoav Freund. Boosting: Foundations and algorithms, 2013.

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

Gemma Team, Sherif El Abd, and Vaibhav Aggarwal et. al. Gemma 4 technical report, 2026.

Yigit Turkmen, Baturalp Buyukates, and Melih Bastopcu. Don't always pick the highest-performing model: An information theoretic view of LLM ensemble selection. In *ICML 2026 Statistical Frameworks for Uncertainty in Agentic Systems*, 2026.

Pat Verga, Sebastian Hofstatter, Sophia Althammer, Yixuan Su, Aleksandra Piktus, Arkady Arkhangorodsky, Minjie Xu, Naomi White, and Patrick Lewis. Replacing judges with juries: Evaluating llm generations with a panel of diverse models. *arXiv preprint arXiv:2404.18796*, 2024.

Tu Vu, Kalpesh Krishna, Salaheddin Alzubi, Chris Tar, Manaal Faruqui, and Yun-Hsuan Sung. Foundational autoraters: Taming large language models for better automatic evaluation. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pp. 17086–17105, November 2024.

Abraham Wald. Sequential tests of statistical hypotheses. In *Breakthroughs in statistics: Foundations and basic theory*, pp. 256–298. Springer, 1992.

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

An Yang, Anfeng Li, and Baosong Yang et. al. Qwen3 technical report, 2025.

Murong Yue, Jie Zhao, Min Zhang, Liang Du, and Ziyu Yao. Large language model cascades with mixture of thought representations for cost-efficient reasoning. In *International Conference on Learning Representations*, volume 2024, pp. 21691–21728, 2024.

Wenbo Zhang, Lijinghua Zhang, Liner Xiang, and Hengrui Cai. Reasoning is not free: Robust adaptive cost-efficient routing for LLM-as-a-judge. In *Forty-third International Conference on Machine Learning*, 2026.

Lianmin Zheng, Wei-Lin Chiang, Ying Sheng, Siyuan Zhuang, Zhanghao Wu, Yonghao Zhuang, Zi Lin, Zhuohan Li, Dacheng Li, Eric Xing, Hao Zhang, Joseph E. Gonzalez, and Ion Stoica. Judging LLM-as-a-judge with MT-bench and chatbot arena. In *Thirty-seventh Conference on Neural Information Processing Systems Datasets and Benchmarks Track*, 2023.

A Related Work

Adaptive Verification. Our formulation builds on sequential hypothesis testing, which jointly optimizes which observations (potentially costly) to acquire and when to stop (Wald, 1992; Chernoff, 1959; Naghshvar & Javidi, 2013). For LLMs, the most common methods to address this trade-off are through routing or cascading. Routing methods predict the most efficient model for a prompt input given a set of models (Hu et al., 2024), while cascading progressively uses larger models for a prompt given a metric of choice, such as low confidence from smaller models (Yue et al., 2024; Dekoninck et al., 2025; Aggarwal et al., 2024; Chen et al., 2024). Both of these strategies have their shortcomings. For one, routing methods can not use additional models if the chosen model performs poorly. On the other hand, cascading fixes the order of models ex-ante, even though correlated errors among models make the value of each subsequent measurement depend critically on the outputs already observed. We generalize these approaches for combining LLM classifiers with a dynamic program, making both the query choices and stopping rule adaptive.

Aggregating Noisy Classifiers. We draw from work on combining the outputs of multiple noisy classifiers to determine a true label. In the context of LLMs, aggregating the outputs of smaller models can be used to approximate the judgement of frontier models at lower cost (Verga et al., 2024). Previous work in crowdsourcing and weak supervision (Dawid & Skene, 1979; Ratner et al., 2016) learn the error rate of each individual classifier and aggregate them for the distribution over the true label, assuming independence between the errors of each classifier. Ensemble diversity (Dietterich, 2000) extends this to a machine learning setting by accounting for the correlation of errors between models, which remains an important hurdle for combining the outputs of large language models (Goel et al., 2025; Kim et al., 2025; Kohli, 2026). In this spirit, our probabilistic model learns a distribution over their responses and prompt embeddings, enabling the dynamic program to account for correlation between model outputs.

Cost-Efficient Routing. Our work also builds on LLM routing methods that optimize for a low inference cost. Previous work in LLM routing (Ding et al., 2024; Ong et al., 2025) attempts to optimize for cost by allowing the user to set a decision threshold between two models based on the learned score of the router for each model on a given query. Notably, neither works attempt to directly optimize for cost, rather allowing the user to make the cost-quality tradeoff at test time. Other works (Zhang et al., 2026) set an explicit worst-case constraint on cost and optimize with respect to it. Most similar to ours is work that maximizes a joint objective for model quality and cost (Hu et al., 2024; Somerstep et al., 2025). Our framework subsumes this setting since routing is a special case of classification circuits, which we optimize over.

B Proofs

Proof of Proposition 1. Consider a circuit that queries a batch $(i_1, \ldots, i_k)$. Replace the batch by the same k queries in any fixed order, without allowing the intermediate responses to affect subsequent queries. After all k responses have been observed, the circuit proceeds exactly as the original circuit would after observing the batch. The resulting circuit therefore has the same measurements, the same final prediction, and the same total query cost. Repeating this replacement for every batch gives the desired sequential circuit. □

Proof of Proposition 2. Consider the objective function for a fixed prompt z:

$$\mathcal{L}_\lambda(\pi, z) = \mathsf{E}_\pi(z) + \lambda \mathsf{C}_\pi(z) \tag{11}$$

Then, the zero-cost circuit is optimal when:

$$\mathsf{E}_{\pi_0}(z) \leq \mathsf{E}_\pi(z) + \lambda \mathsf{C}_\pi(z) \tag{12}$$

since $\mathsf{C}_{\pi_0}(z) = 0$ for the zero-cost circuit. This, together with $\lambda \geq 0$ directly results in proving Proposition 2.

Now consider the expression:

$$\lambda_{LB} = \sup_{\pi : C_\pi(z) > 0} \frac{[\mathsf{E}_{\pi_0}(z) - \mathsf{E}_\pi(z)]_+}{\mathsf{C}_\pi(z)} \tag{13}$$

The upper bound on λ_{LB} is obtained using the following bounds: $\mathsf{E}_{\pi_0}(z) \leq \frac{1}{2}$, $\mathsf{E}_{\pi}(z) \geq 0$, and $\mathsf{C}_{\pi}(z) \geq \mathsf{C}_{\mathsf{min}}(z)$.

□

Proof of Proposition 4. For a fixed prompt p, define

$$\eta_p(x) = \Pr(Y_P = 1 \mid P = p, X = x)$$

and

$$\eta_p(x, z) = \Pr(Y_P = 1 \mid P = p, X = x, Z = z).$$

Since the Bayes accuracy for a binary label is

$$g(u) = \max\{u, 1 - u\},$$

we have

$$A^{\max}_{n+1}(p) - A^{\max}_{n}(p) = \mathbb{E}[g(\eta_p(X, Z)) - g(\eta_p(X)) \mid P = p]\,.$$

The function g is 1-Lipschitz, so

$$A^{\max}_{n+1}(p) - A^{\max}_{n}(p) \leq \mathbb{E}[|\eta_p(X, Z) - \eta_p(X)| \mid P = p]\,.$$

Because Y_P is binary, the inner expectation equals

$$\mathrm{TV}(\Pr(Y_P, Z \mid P = p, X), \Pr(Y_P \mid P = p, X)\Pr(Z \mid P = p, X))\,.$$

Pinsker's inequality and Jensen's inequality therefore give

$$A^{\max}_{n+1}(p) - A^{\max}_{n}(p) \leq \sqrt{\frac{1}{2} I(Y_P; Z \mid P = p, X)}.$$

Averaging over P and applying Jensen's inequality yield the result. Nonnegativity follows because observing Z cannot decrease Bayes accuracy. □

B.1 Proofs for Fixed-Prompt Scaling

For the asymptotic analysis, we first fix a prompt z and suppress it from the notation to simplify notation. We write $a_i = a_i(z)$ for the accuracy of the instrument i and $c_i = c_i(z)$ for its cost. In contrast to the finite-circuit formulation of Section 2, each instrument $i \in \mathcal{H}$ may now be queried arbitrarily many times, with independent randomness across calls.

For the fixed prompt z, classification reduces to a hypothesis test,

$$H_0 :\ y(z) = 0 \qquad \text{against} \qquad H_1 :\ y(z) = 1.$$

Under the symmetric testing model, a query to instrument i produces

$$X \mid H_1, i \sim \mathrm{Ber}(a_i), \qquad X \mid H_0, i \sim \mathrm{Ber}(1 - a_i). \tag{14}$$

The log-likelihood ratio contributed by a response $x \in \{0, 1\}$ is therefore

$$L_i(x) \coloneqq \log \frac{\Pr(X = x \mid H_1, i)}{\Pr(X = x \mid H_0, i)} = (2x - 1)\ell_i, \qquad \ell_i \coloneqq \log \frac{a_i}{1 - a_i}. \tag{15}$$

Then, we have,

$$\mathbb{E}[L_i(x)|H_1] = D_{\mathrm{KL}}(\mathrm{Ber}(a_i) \,\|\, \mathrm{Ber}(1 - a_i)) = (2a_i - 1)\ell_i = d_i \tag{16}$$

and $\mathbb{E}[L_i(x)|H_0] = -d_i$. Thus d_i is the expected information gained from one query to instrument i in favor of the true hypothesis.

Because queries have different costs, we define the information rate of instrument i by

$$I_i \coloneqq \frac{d_i}{c_i}, \qquad I \coloneqq \max_{i \in \mathcal{H}} I_i. \tag{17}$$

It will be useful later on to let $i^\star \in \arg\max_{i \in \mathcal{H}} I_i$ denote an instrument attaining the maximum information rate.

Let $A_t \in \mathcal{H}$ denote the instrument queried at time t, and let X_t denote its response. Write

$$\mathcal{F}_t = \sigma(A_1, X_1, \ldots, A_t, X_t)$$

for the history available after t queries. A circuit chooses each A_t using only the preceding history and terminates after a stopping time τ.

Conditional on the hypothesis, the selected instrument, and the preceding history, the response at time t follows Equation (14). In particular,

$$\mathbb{E}_1[L_{A_t}(X_t) \mid \mathcal{F}_{t-1}, A_t] = d_{A_t}, \qquad \mathbb{E}_0[L_{A_t}(X_t) \mid \mathcal{F}_{t-1}, A_t] = -d_{A_t}.$$

The cumulative log-likelihood ratio after t queries is

$$\Lambda_t \coloneqq \sum_{s=1}^{t} L_{A_s}(X_s) = \sum_{s=1}^{t} (2X_s - 1)\ell_{A_s}. \tag{18}$$

Let

$$q \coloneqq q_z(\varnothing) \in (0,1), \qquad b \coloneqq \log \frac{q}{1-q}$$

denote the estimated prior probability of H_1 and its corresponding log-odds. We assume $0 < q < 1$ since otherwise the target label is already known and no querying is necessary. Since the querying rule depends only on the observed history, its contribution to the likelihood ratio is the same under both hypotheses. Hence the posterior log-odds after t queries are

$$S_t \coloneqq \log \frac{\Pr(H_1 \mid \mathcal{F}_t)}{\Pr(H_0 \mid \mathcal{F}_t)} = b + \Lambda_t. \tag{19}$$

Equivalently, the posterior probability of H_1 is

$$p_t \coloneqq \Pr(H_1 \mid \mathcal{F}_t) = \frac{1}{1 + e^{-S_t}}. \tag{20}$$

If the circuit stops at time t, the Bayes decision predicts H_1 when $S_t \geq 0$ and H_0 otherwise. Its conditional probability of error is

$$\min\{p_t, 1 - p_t\} = \frac{1}{1 + e^{|S_t|}}. \tag{21}$$

Error and cost of a circuit. For a circuit policy π, let P_y^π denote the law of its terminated history under H_y, and let $\mathbb{E}_y^\pi$ denote expectation under this law. Define

$$N_i \coloneqq \sum_{t=1}^{\tau} \mathbf{1}_{\{A_t = i\}}$$

as the number of times instrument i is queried before termination. Its prior-averaged expected query count is

$$\bar{N}_i^\pi \coloneqq q\,\mathbb{E}_1^\pi[N_i] + (1-q)\,\mathbb{E}_0^\pi[N_i]. \tag{22}$$

The Bayes error probability and expected query cost of π are then

$$\mathsf{E}(\pi) = qP_1^\pi(\widehat{Y} = 0) + (1-q)P_0^\pi(\widehat{Y} = 1), \tag{23}$$

$$\mathsf{C}(\pi) = \sum_{i \in \mathcal{H}} c_i \bar{N}_i^\pi. \tag{24}$$

For the optimal policy associated with query-price parameter λ, these quantities coincide with $\mathsf{E}_\lambda(z)$ and $\mathsf{C}_\lambda(z)$ in Theorem 1.

Lemma 1. *Every sequential circuit with finite expected cost satisfies*

$$D_{\mathrm{KL}}(P_y^\pi \| P_{1-y}^\pi) = \sum_{i \in \mathcal{H}} d_i \mathbb{E}_y^\pi[N_i], \qquad y \in \{0,1\}. \tag{25}$$

Consequently,

$$\log \frac{m}{2\mathsf{E}(\pi)} \leq \sum_{i \in \mathcal{H}} d_i \bar{N}_i^\pi \leq I\mathsf{C}(\pi). \tag{26}$$

Proof. Recall that $\mathcal{F}_{t-1}$ denotes the history available before observing X_t. Since A_t and the event $\{\tau \geq t\}$ are determined by $\mathcal{F}_{t-1}$, under H_1,

$$\mathbb{E}_1^\pi[(2X_t - 1)\ell_{A_t} \mid \mathcal{F}_{t-1}] = d_{A_t}.$$

Conditional on the same realized history, the query and stopping rules of π are identical under H_1 and H_0. These terms therefore cancel from the likelihood ratio of the stopped history, giving

$$D_{\mathrm{KL}}(P_1^\pi \| P_0^\pi) = \mathbb{E}_1^\pi[\Lambda_\tau].$$

Using Equation (18) and conditioning before each response,

$$D_{\mathrm{KL}}(P_1^\pi \| P_0^\pi) = \mathbb{E}_1^\pi[\Lambda_\tau] = \mathbb{E}_1^\pi\left[\sum_{t=1}^{\tau} d_{A_t}\right] = \sum_{i \in \mathcal{H}} d_i \, \mathbb{E}_1^\pi[N_i]. \tag{27}$$

The same argument with the likelihood ratio reversed gives $D_{\mathrm{KL}}(P_0^\pi \| P_1^\pi) = \sum_{i \in \mathcal{H}} d_i \, \mathbb{E}_0^\pi[N_i]$, which proves Equation (25).

Now let

$$\alpha_\pi = P_1^\pi(\widehat{Y} = 0), \qquad \beta_\pi = P_0^\pi(\widehat{Y} = 1).$$

By Equation (23), with $m = \min\{q, 1-q\}$,

$$\mathsf{E}(\pi) = q\alpha_\pi + (1-q)\beta_\pi \geq m(\alpha_\pi + \beta_\pi).$$

Data processing through the final prediction, followed by the Bretagnolle–Huber inequality (Bretagnolle & Huber, 1979), gives for either $y \in \{0, 1\}$,

$$D_{\mathrm{KL}}(P_1^\pi \| P_0^\pi) \geq D_{\mathrm{KL}}(\mathrm{Ber}(1-\alpha_\pi) \,\|\, \mathrm{Ber}(\beta_\pi)) \geq \log \frac{1}{2(\alpha_\pi + \beta_\pi)} \geq \log \frac{m}{2\mathsf{E}(\pi)}.$$

Averaging these two inequalities according to the prior and applying Equation (25),

$$\begin{aligned} \log \frac{m}{2\mathsf{E}(\pi)} &\leq q D_{\mathrm{KL}}(P_1^\pi \| P_0^\pi) + (1-q) D_{\mathrm{KL}}(P_0^\pi \| P_1^\pi) \\ &= \sum_{i \in \mathcal{H}} d_i \bar{N}_i^\pi \leq I \sum_{i \in \mathcal{H}} c_i \bar{N}_i^\pi = I\mathsf{C}(\pi), \end{aligned}$$

where the final inequality uses $d_i \leq I c_i$. This proves Equation (26). □

Lemma 2. *Suppose $0 < I < \infty$ and choose $i^\star \in \arg\max_i I_i$. For $B \geq 0$, let π_B be the circuit that repeatedly queries instrument $i^\star$ until*

$$\tau_B = \inf\{t \geq 0 : |S_t| \geq B\}, \tag{28}$$

and then make the Bayes prediction. This test has finite expected cost and

$$\mathsf{E}(\pi_B) \leq e^{-B}, \qquad I\mathsf{C}(\pi_B) \leq B + \ell_{i^\star}. \tag{29}$$

Proof. If $|b| \geq B$, then $\tau_B = 0$. The circuit makes no queries, so $\mathsf{C}(\pi_B) = 0$, while Equation (21) gives

$$\mathsf{E}(\pi_B) = \frac{1}{1 + e^{|b|}} \leq e^{-B}.$$

We may therefore assume $|b| < B$. Since π_B always queries $i^\star$, each update of the posterior log-odds has magnitude $\ell_{i^\star}$. If $T_n = n < \tau_B$, then $|S_{T_n}| < B$. Otherwise, $|S_{\tau_B - 1}| < B$ and the final update has magnitude $\ell_{i^\star}$. Thus, for $T_n = \tau_B \wedge n$,

$$|S_{T_n}| \leq B + \ell_{i^\star}. \tag{30}$$

We next bound the expected stopping time. Under H_y, a query to $i^\star$ satisfies

$$\mathbb{E}_y^{\pi_B}[(2y-1)(S_t - S_{t-1}) \mid \mathcal{F}_{t-1}] = d_{i^\star}.$$

Hence

$$M_t = (2y-1)(S_t - b) - t d_{i^\star}$$

is a martingale under H_y. Applying optional stopping at the bounded stopping time T_n gives

$$d_{i^\star}\mathbb{E}_y^{\pi_B}[T_n] = \mathbb{E}_y^{\pi_B}\left[(2y-1)(S_{T_n}-b)\right] \le B + \ell_{i^\star} - (2y-1)b, \tag{31}$$

where the inequality follows from Equation (30). Since $I > 0$, we have $d_{i^\star} > 0$. Letting $n \to \infty$ and applying monotone convergence therefore yields

$$\mathbb{E}_y^{\pi_B}[\tau_B] \le \frac{B + \ell_{i^\star} - (2y-1)b}{d_{i^\star}} < \infty.$$

In particular, $\tau_B < \infty$ almost surely under both hypotheses.

Because the circuit queries only $i^\star$,

$$\begin{aligned} I\mathsf{C}(\pi_B) &= Ic_{i^\star}\left(q\mathbb{E}_1^{\pi_B}[\tau_B] + (1-q)\mathbb{E}_0^{\pi_B}[\tau_B]\right) \\ &= d_{i^\star}\left(q\mathbb{E}_1^{\pi_B}[\tau_B] + (1-q)\mathbb{E}_0^{\pi_B}[\tau_B]\right) \\ &\le q(B + \ell_{i^\star} - b) + (1-q)(B + \ell_{i^\star} + b) \\ &= B + \ell_{i^\star} + (1-2q)b \\ &\le B + \ell_{i^\star}. \end{aligned}$$

The last inequality holds because $b = \log(q/(1-q))$ has the same sign as $2q-1$.

Finally, $\tau_B < \infty$ almost surely and $|S_{\tau_B}| \ge B$. By Equation (21), the conditional Bayes error at termination therefore satisfies

$$\min\{p_{\tau_B}, 1-p_{\tau_B}\} = \frac{1}{1+e^{|S_{\tau_B}|}} \le e^{-B}.$$

Averaging over the terminated histories gives $\mathsf{E}(\pi_B) \le e^{-B}$. This proves Equation (29). □

Lemma 3. *Suppose $0 < I < \infty$. Let E_λ and C_λ be the error and expected cost of any circuit minimizing the objective $\mathsf{E}(\pi) + \lambda\mathsf{C}(\pi)$. As $\lambda \downarrow 0$,*

$$\mathsf{E}_\lambda = \Theta(\lambda/I), \tag{32}$$

$$I\mathsf{C}_\lambda = \log(I/\lambda) + O(1). \tag{33}$$

Proof. For $0 < \lambda < I$, set $\eta = \lambda/I$ and choose $B = \log(1/\eta)$ in Lemma 2. Optimality gives

$$\mathsf{E}_\lambda + \lambda\mathsf{C}_\lambda \le \eta\big(1 + \log(1/\eta) + \ell_{i^\star}\big). \tag{34}$$

To bound the error itself, combine this with Equation (26) and set $x_\lambda = \mathsf{E}_\lambda/\eta$. Then

$$x_\lambda + \log\frac{m}{2\eta x_\lambda} \le \frac{\mathsf{E}_\lambda + \lambda\mathsf{C}_\lambda}{\eta} \le 1 + \log\frac{1}{\eta} + \ell_{i^\star}.$$

Cancelling $\log(1/\eta)$ gives the nonasymptotic bound

$$x_\lambda - \log x_\lambda \le 1 + \ell_{i^\star} + \log\frac{2}{m}, \qquad 0 < \lambda < I. \tag{35}$$

For a fixed prompt, the right-hand side is constant. Since $x - \log x$ diverges at both zero and infinity, x_λ is bounded above and away from zero. This proves Equation (32).

For the cost, Equation (26) and Equation (34) give

$$\log\frac{m}{2\mathsf{E}_\lambda} \le I\mathsf{C}_\lambda \le 1 + \log\frac{I}{\lambda} + \ell_{i^\star}.$$

By Equation (32), the lower bound is $\log(I/\lambda) + O(1)$, proving Equation (33). □

Proof of Theorem 1. Restoring the prompt dependence, Equation (32) and Equation (33) give

$$-\log\mathsf{E}_\lambda(z) = \log\frac{I(z)}{\lambda} + O(1) = I(z)\mathsf{C}_\lambda(z) + O(1).$$

□

B.2 Query Allocation and Adaptive Stopping

Lemma 4 (Asymptotic query allocation). *For a fixed prompt with $0 < I < \infty$, write $\bar{N}_i^\lambda = \bar{N}_i^{\pi^\star_\lambda}$. Then*

$$\sum_{i:I_i<I} c_i \bar{N}_i^\lambda = O(1), \qquad \frac{\sum_{i:I_i<I} c_i \bar{N}_i^\lambda}{\mathsf{C}_\lambda} \longrightarrow 0 \quad \text{as } \lambda \downarrow 0.$$

Proof. For each instrument i, the gap $I - I_i$ measures its loss in information rate relative to the best available instrument. Using $d_i = I_i c_i$ and Equation (26),

$$\sum_{i\in\mathcal{H}} (I - I_i) c_i \bar{N}_i^\lambda = I\mathsf{C}_\lambda - \sum_{i\in\mathcal{H}} d_i \bar{N}_i^\lambda \le I\mathsf{C}_\lambda - \log \frac{m}{2\mathsf{E}_\lambda} = O(1), \tag{36}$$

where the last equality follows from Lemma 3. Now suppose that some instrument satisfies $I_i < I$, and define the smallest positive information-rate gap

$$\Delta := \min_{i:I_i<I} (I - I_i) > 0.$$

Then

$$\Delta \sum_{i:I_i<I} c_i \bar{N}_i^\lambda \le \sum_{i\in\mathcal{H}} (I - I_i) c_i \bar{N}_i^\lambda = O(1),$$

and therefore

$$\sum_{i:I_i<I} c_i \bar{N}_i^\lambda = O(1).$$

If no such instrument exists, the sum is identically zero. Finally, Equation (33) gives $\mathsf{C}_\lambda \to \infty$ as $\lambda \downarrow 0$. Since the expected cost allocated to instruments with $I_i < I$ remains bounded,

$$\frac{\sum_{i:I_i<I} c_i \bar{N}_i^\lambda}{\mathsf{C}_\lambda} \longrightarrow 0.$$

□

B.3 Scaling Across Prompts

Restore the dependence on z and define $\mathsf{E}_\lambda = \mathbb{E}_Z[\mathsf{E}_\lambda(Z)]$ and $\mathsf{C}_\lambda = \mathbb{E}_Z[\mathsf{C}_\lambda(Z)]$. We optimize each circuit separately for each observed prompt with the same query price λ. To control the bounds we obtained above uniformly over prompts, suppose that, for constants $m_0, c_{\max} > 0$,

$$m(z) = \min\{q_z(\varnothing), 1 - q_z(\varnothing)\} \ge m_0, \qquad 0 < c_i(z) \le c_{\max} \tag{37}$$

for every i and almost every z. No uniform lower bound on costs is required.

Lemma 5 (Uniform error bounds for a fixed prompt). *For each $r_0 > 0$, there are constants $b_0, B_0, K > 0$, independent of z and λ, such that, for $0 < \lambda < r_0$,*

$$b_0 \min\{1, \lambda/I(z)\} \le \mathsf{E}_\lambda(z) \le B_0 \min\{1, \lambda/I(z)\}, \qquad 0 < I(z) \le r_0, \tag{38}$$

$$\mathsf{E}_\lambda(z) \le K\lambda, \qquad I(z) > r_0. \tag{39}$$

Proof. The constants in Equation (35) depend on $m(z)$ and $\ell_{i^\star}(z)$. For $u = 1 - a_i(z)$, the bound $u \log(1/u) \le 1/e$ gives

$$\ell_i(z) - d_i(z) = 2(1 - a_i(z)) \log \frac{a_i(z)}{1 - a_i(z)} \le \frac{2}{e} < 1.$$

Since $d_{i^\star}(z) = I(z) c_{i^\star}(z)$,

$$\ell_{i^\star}(z) \le 1 + c_{\max} I(z). \tag{40}$$

For $\lambda < I(z) \le r_0$, Equation (35) therefore confines $x = I(z)\mathsf{E}_\lambda(z)/\lambda$ to a fixed interval inside $(0, \infty)$. Thus $\mathsf{E}_\lambda(z) \asymp \lambda/I(z)$ uniformly on this range.

If $I(z) \le \lambda$, predicting without queries gives $\mathsf{E}_\lambda(z) + \lambda\mathsf{C}_\lambda(z) \le m(z) \le 1/2$. Hence $I(z)\mathsf{C}_\lambda(z) \le 1/2$, and Equation (26) implies

$$\frac{m_0}{2\sqrt{e}} \le \mathsf{E}_\lambda(z) \le \frac{1}{2}.$$

Together, these two ranges prove Equation (38).

Now for $I(z) > r_0$, Equation (35) and Equation (40) give $x - \log x \le K_0 + c_{\max} I(z)$, with K_0 fixed. Since $x - \log x \ge x/2$ for $x > 0$,

$$\mathsf{E}_\lambda(z) \le 2\lambda \left(\frac{K_0}{I(z)} + c_{\max} \right) \le K\lambda.$$

□

We will now average these error bounds and then recover the cost from optimality.

Lemma 6 (Recovering cost from error)**.** *Consider a fixed policy class Π containing a zero-cost policy with error at most $1/2$. For each $\lambda > 0$, let $(\mathsf{E}_\lambda, \mathsf{C}_\lambda)$ denote the error and cost of a minimizer of*

$$\mathsf{E}(\pi) + \lambda \mathsf{C}(\pi), \qquad \pi \in \Pi.$$

Then

$$\mathsf{C}_\lambda = \int_\lambda^\infty \frac{\mathsf{E}_s}{s^2}\, ds - \frac{\mathsf{E}_\lambda}{\lambda}, \qquad \lambda > 0. \tag{41}$$

Here, if the minimizer is not unique, any measurable selection of minimizers may be used.

Proof. Let

$$V(\lambda) = \min_{\pi \in \Pi} \big\{ \mathsf{E}(\pi) + \lambda \mathsf{C}(\pi) \big\}.$$

As the pointwise infimum of affine functions of λ, V is concave. Hence it is differentiable almost everywhere. At every point of differentiability, any minimizing policy satisfies

$$V'(\lambda) = \mathsf{C}_\lambda,$$

and therefore

$$\frac{d}{d\lambda} \frac{V(\lambda)}{\lambda} = \frac{\lambda \mathsf{C}_\lambda - V(\lambda)}{\lambda^2} = -\frac{\mathsf{E}_\lambda}{\lambda^2}.$$

The zero-cost policy gives $0 \le V(\lambda) \le 1/2$, so $V(\lambda)/\lambda \to 0$ as $\lambda \to \infty$. Integrating the preceding identity from λ to infinity yields

$$\frac{V(\lambda)}{\lambda} = \int_\lambda^\infty \frac{\mathsf{E}_s}{s^2}\, ds.$$

Since $V(\lambda) = \mathsf{E}_\lambda + \lambda \mathsf{C}_\lambda$, rearranging gives Equation (41). □

Proof of Theorem 2. Set $D = 1/I(Z)$. The tail assumption implies

$$G(t) := \Pr(D > t) \asymp t^{-\alpha} \qquad (t \to \infty).$$

Choose $d_0 > 0$ so that these bounds hold for $t \ge d_0$. By Lemma 5, for sufficiently small λ,

$$\mathsf{E}_\lambda \asymp M_\lambda, \qquad M_\lambda := \mathbb{E}\big[\min\{1, \lambda D\} \mathbf{1}_{\{D \ge d_0\}}\big].$$

Indeed, the tail-integral formula gives

$$M_\lambda = \lambda d_0 \Pr(D \ge d_0) + \lambda \int_{d_0}^{1/\lambda} G(t)\, dt \tag{42}$$

$$\asymp \lambda \left(1 + \int_{d_0}^{1/\lambda} t^{-\alpha}\, dt \right).$$

Hence

$$\mathsf{E}_\lambda \asymp \begin{cases} \lambda, & \alpha > 1, \\ \lambda \log(1/\lambda), & \alpha = 1, \\ \lambda^\alpha, & 0 < \alpha < 1. \end{cases} \tag{43}$$

We next determine the corresponding cost. By Equation (41),

$$\mathsf{C}_\lambda = \int_\lambda^\infty \frac{\mathsf{E}_s}{s^2}\, ds - \frac{\mathsf{E}_\lambda}{\lambda}.$$

Using Equation (43), the contribution of the integral away from zero is bounded, while near zero

$$\mathsf{C}_\lambda \asymp \begin{cases} \log(1/\lambda), & \alpha > 1, \\ \log^2(1/\lambda), & \alpha = 1. \end{cases}$$

For $0 < \alpha < 1$, the same identity gives $\mathsf{C}_\lambda = O(\lambda^{\alpha-1})$.

For the matching lower bound, choose $u, v > 0$ such that

$$u\lambda^\alpha \le \mathsf{E}_\lambda \le v\lambda^\alpha$$

for small λ, and choose $a > 1$ with $ua^\alpha > v$. Optimality at prices λ and $a\lambda$ gives

$$\mathsf{E}_{a\lambda} \le \mathsf{E}_\lambda + a\lambda\mathsf{C}_\lambda,$$

and therefore

$$\mathsf{C}_\lambda \ge \frac{\mathsf{E}_{a\lambda} - \mathsf{E}_\lambda}{a\lambda} \ge \frac{ua^\alpha - v}{a}\lambda^{\alpha-1}.$$

Thus

$$\mathsf{C}_\lambda \asymp \begin{cases} \log(1/\lambda), & \alpha > 1, \\ \log^2(1/\lambda), & \alpha = 1, \\ \lambda^{\alpha-1}, & 0 < \alpha < 1. \end{cases} \tag{44}$$

It remains to eliminate λ. If $\alpha > 1$, then

$$-\log \mathsf{E}_\lambda = \log(1/\lambda) + O(1) \asymp \mathsf{C}_\lambda.$$

If $\alpha = 1$,

$$-\log \mathsf{E}_\lambda = \log(1/\lambda) - \log\log(1/\lambda) + O(1) \asymp \log(1/\lambda) \asymp \sqrt{\mathsf{C}_\lambda}.$$

Finally, if $0 < \alpha < 1$,

$$\mathsf{C}_\lambda \asymp \lambda^{-(1-\alpha)}$$

and hence

$$\mathsf{E}_\lambda \asymp \lambda^\alpha \asymp \mathsf{C}_\lambda^{-\frac{\alpha}{1-\alpha}}.$$

□

Dataset	Train	Val.	Test	Instruments
BeaverTails	24,461	2,719	3,020	Gemma 4, Qwen3 (7 models)
PolitiCAUSE	12,438	2,664	2,665	Gemma 4, Qwen3 (7 models)
LMArena	25,162	2,797	6,993	Gemma 4, Qwen3 (7 models)
HealthBench	21,015	2,331	5,785	Granite 4.2, gpt-oss (8 models)

Table 1: Datasets considered in this work

C Additional Experimental Details

All code for generating the instrument measurements with vLLM (Kwon, 2025), and for training and evaluating LMCircuit and all baselines, is available at https://anonymous.4open.science/r/LMCircuits-ED8D.

C.1 Data

In this work, we consider the following four classification datasets.

BeaverTails (Ji et al., 2023). Each prompt is a user request together with a model's response. An instrument answers whether the response is unsafe under the benchmark's harm categories.

PolitiCAUSE (Garcia-Corral et al., 2024). Each prompt is one sentence from political text, and an instrument answers whether the sentence states a causal relation.

LMArena (Chiang et al., 2024). Each prompt is a user message together with two responses and an instrument answers which response is more likely to be preferred (A or B). The order in which the two responses are shown is drawn at random for every run of the prompt. We consider 34,952 single-turn English comparisons.

HealthBench (Arora et al., 2025). Each prompt is a conversation between a user and an AI assistant together with one rubric criterion written by physicians, and an instrument answers whether the last response meets the criterion, using the benchmark's grading instruction. Each prompt was judged by two to five physicians. Therefore, the target is the probability that a randomly chosen physician judges the criterion met. The eight instruments are *configurations of three reasoning models*: Granite 4.2 3B with reasoning off, low and default, and gpt-oss-20b (low, medium and high effort) and gpt-oss-120b (low and medium effort) (Agarwal et al., 2025).

For every dataset, the instrument is given the dataset's official classification instruction, with the model's default decoding (unless we state otherwise). The first three datasets share seven instruments: Gemma 4 (Team et al., 2026) (E2B, E4B, 26B-A4B and 31B, instruction-tuned) and Qwen3 (Yang et al., 2025) (4B, 8B and 32B). At their defaults, the Gemma models answer directly and the Qwen3 models reason before answering. Table 1 summarizes the datasets.

Each dataset is split into training, validation and test prompts. Using vLLM (Kwon, 2025), every model is queried five times on each training prompt and eight times on each validation and test prompt. For a prompt z we write $\mathcal{D}(z)$ for the multiset of recorded pairs (i, x), and $N(z) = |\mathcal{D}(z)|$. The ground truth label of z enters through counts $(m_0(z), m_1(z))$. For a hard label (as in Beavertails, PolitiCause, and LMArena), $m_{y(z)}(z) = 1$ and the other count is zero. On HealthBench, where several physicians judge the same prompt, $m_1(z)$ and $m_0(z)$ count the physicians who judged the criterion as met and not met.

Some baseline methods require probabilities of each label from the instrument. With each query, we request the log-probabilities at the token of the verdict and add the probability of the top 5 alternative tokens that would spell label 0 or label 1. Denote by p_1 the probability of label 1 renormalized over the two labels. What p_1 measures depends on whether the model reasons before answering, so we obtain it by one of two methods:

1. *For non-thinking models*, p_1 is the model's own probability of label 1 given the prompt, and a single query estimates $\Pr(X^{(i)} = 1 \mid z)$.
2. *For thinking models*, p_1 is often close to 0 or 1. Instead, we estimate $\Pr(X^{(i)} = 1 \mid z)$ by averaging over five repeated queries.

Embedding model. The embedding $\phi(z)$ comes from Qwen3-Embedding-0.6B (Yang et al., 2025) with last-token pooling, giving a 1024-dimensional vector per text. For HealthBench, the rubric criterion c and the conversation u are embedded separately and $\phi(z) = [\, c,\ u,\ c \odot u \,]$. For LMArena, the user prompt is embedded together with each response, giving a and b, and $\phi(z) = [\, (a+b)/2,\ a-b,\ |a-b|,\ a \odot b \,]$.

C.2 Training the Probabilistic Model

Probabilistic Model (distribution estimator) Architecture. The model f_θ of Section 3.2 has two heads that share a linear projection $h(z) = W\phi(z) \in \mathbb{R}^{64}$. The measurement head is a two-layer network with hidden width 128 that maps $h(z)$ to $\nu_\theta^{(1)}(\phi(z)), \ldots, \nu_\theta^{(n)}(\phi(z))$.

The label head does the following: (1) encodes each observed pair $(j, x^{(j)}) \in s$ as $t_{j,x} = g(e_j + e'_x)$, where e_j and e'_x are learned 64-dimensional embeddings of the instrument and the measurement and g is a two-layer network, (2) sums these vectors over the state, and (3) passes $[\, h(z),\ \sum_{(j,x)\in s} t_{j,x},\ |\mathcal{M}(s)|/n \,]$ through a two-layer network of width 128 and a linear output to the logit of $\eta_\theta(\phi(z), s)$. The sum makes η_θ independent of the order in which measurements arrived, and an unqueried instrument does not contribute to this sum.

On LMArena, where each query shows the two responses in a random order o, the embeddings and $\nu_\theta^{(i)}$ also depend on o, and the network is made exactly equivariant to exchanging the two responses. The exchange negates the logit of η_θ and swaps the two outcomes of every $\nu_\theta^{(i)}$.

Loss function. For each prompt z in a minibatch B we draw one state s_z: a number k uniformly from $\{0, \ldots, n\}$, a uniformly random set $\mathcal{M}(s_z)$ of k instruments, and for each $i \in \mathcal{M}(s_z)$ one of its recorded measurements on z, chosen uniformly. With $\ell(p, x) = -x \log p - (1-x)\log(1-p)$ the binary cross-entropy, the label loss is,

$$\mathcal{L}_Y(\theta) = \frac{\sum_{z\in B} \big[m_1(z)\, \ell\big(\eta_\theta(\phi(z), s_z), 1\big) + m_0(z)\, \ell\big(\eta_\theta(\phi(z), s_z), 0\big) \big]}{\sum_{z\in B} \big(m_0(z) + m_1(z)\big)}, \tag{45}$$

and the measurement loss scores each $\nu_\theta^{(i)}$ against every recorded measurement of instrument i,

$$\mathcal{L}_X(\theta) = \frac{1}{|B|} \sum_{z\in B} \frac{1}{N(z)} \sum_{(i,x)\in\mathcal{D}(z)} \ell\big(\nu_\theta^{(i)}(\phi(z)), x\big). \tag{46}$$

We minimize $\mathcal{L}_Y + \mathcal{L}_X$.

Optimization and calibration. We use AdamW (learning rate 2×10^{-3}, weight decay 10^{-4}, batch size 256) with a cosine schedule over at most 300 epochs. After each epoch we compute $\mathcal{L}_Y$ on a fixed set of validation states, one per validation prompt, keep the epoch where it is lowest, and stop after 30 epochs without improvement. We then fit one temperature per head on the validation set, $\eta = \sigma(\ell_\eta / T_\eta)$ and $\nu^{(i)} = \sigma(\ell_{\nu,i} / T_\nu)$ on the logits ℓ, each chosen to minimize the validation negative log-likelihood. We train five models per dataset by changing the random seed.

Generating circuits. We substitute $q_z(s) = \eta_\theta(\phi(z), s)$ and $Q^{[i]}_{X|Z}(1 \mid z) = \nu_\theta^{(i)}(\phi(z))$ into Equation (7). The stopping loss is $e_z(s) = \min\{q_z(s), 1 - q_z(s)\}$ for accuracy.

The program uses a cost known before the query, $c_i(z)$ equal to the input tokens of z times the input price plus the 95th percentile of instrument i's output token length on training prompts of that dataset times the output price. Costs are divided by the mean training cost of the most expensive instrument. We solve Equation (7) for $\lambda = 0$, for 401 values of λ spaced geometrically from 10^{-6} to 10^2, and for $\lambda = \infty$ (the zero-cost prediction). For each budget, we choose the λ with the best validation

	Gemma 2B	Gemma 4B	Gemma 26B	Gemma 31B	Qwen 4B	Qwen 8B	Qwen 32B	Granite 3B	OSS 20B	OSS 120B
Input	0.04	0.02	0.042	0.09	0.03	0.20	0.08	0.03	0.018	0.03
Output	0.08	0.10	0.22	0.34	0.03	0.20	0.28	0.12	0.09	0.17

Table 2: Price of each judge in USD per million tokens, from the cheapest listed provider for each model on 2026-09-24. The providers are OpenRouter, AWS Bedrock, and Fireworks.

metric among those whose mean validation cost is within the budget and report its test metric. Each held-out prompt is replayed eight times, the r-th replay using the r-th recorded measurement of every instrument, and metrics average over the replays.

C.3 Full list of baselines

- **Predictive routers:** RouteLLM (Ong et al., 2025), HybridLLM (Ding et al., 2024), and CARROT (Somerstep et al., 2025)
- **Model cascades:** Cascade Routing (Dekoninck et al., 2025), Consistency Cascades (Yue et al., 2024), Cascaded Selective (Jung et al., 2025), and FrugalGPT (Chen et al., 2024)
- **Sampling methods:** Adaptive -Consistency (Aggarwal et al., 2023)

C.4 Additional Experimental Results

In this subsection, we provide additional experimental analysis of our learned circuits.

C.4.1 Calibration of the circuit's beliefs.

Our method returns a predictive distribution over the ground truth label, allowing us to assess whether its *confidence* is well calibrated. The circuit's belief at the terminal state s_T is $\hat{p}(z) = \eta_\theta(\phi(z), s_T)$. We evaluate these probabilities with the Brier score

$$\mathsf{Brier}(\pi) = \mathbb{E}_Z \, \mathbb{E}_{\boldsymbol{X}|Z} \Big[\big(\hat{p}(Z) - Y \big)^2 \Big], \tag{47}$$

which is *small* only when the forecasts are both calibrated and sharp. A forecast is calibrated if, among the prompts forecast at probability p, a fraction p have label 1. While the dynamic program **never optimizes for the Brier metric directly**, it attains very competitive calibration across different budgets; see Figure 6. We also present calibration result of majority voting as a scheme, by treating the fraction of queried instruments that predict label 1 as its forecast probability, and comparing this probability with the observed frequency of label 1.

C.4.2 Joint distribution between instruments

To test the assumption that LLMs are conditionally independent given any prompt (Assumption 1), we queried eight instruments 100 times each on 20 prompts from BeaverTails. For any two instruments, we compute the Pearson correlation ϕ of their answers in two ways: *within a prompt*, across its 100 calls, averaged over prompts; and *across prompts*, pooling all 2,000 calls. Under the assumption, the first is zero, while the second can be large because the instruments share the (latent) difficulty of each prompt. To test the joint independence of all eight instruments on a prompt, we use their total correlation

$$\mathsf{TC}(z) = \sum_i H\big(X^{(i)} \mid Z = z\big) - H\big(X^{(1)}, \ldots, X^{(8)} \mid Z = z\big), \tag{48}$$

estimated in bits from the 100 calls, which is zero if and only if the answers are jointly independent. We compare it with a permutation null that shuffles each instrument's calls independently, preserving each instrument's answer frequencies while removing any dependence between instruments. We present these results in Figure 7.

Given labeled data, the fitted model treats measurements as independent given the embedding, so any correlation it implies across prompts must come from the embedding. We use this joint to

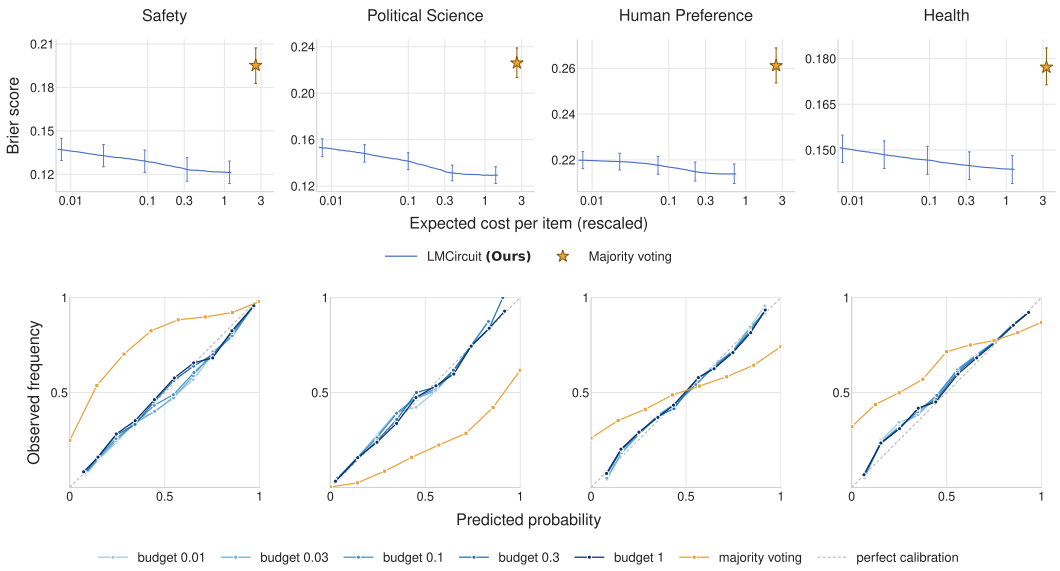


Figure 6: **The circuit's beliefs are calibrated at every budget. Top:** Brier score on the test set (*lower is better*) against expected cost per item. **Bottom:** reliability diagrams of the circuits at different budgets and of the majority vote. The dashed line is perfect calibration. The circuit's curves lie on the diagonal at every budget, whereas the majority votes is miscalibrated in a direction that differs by task.

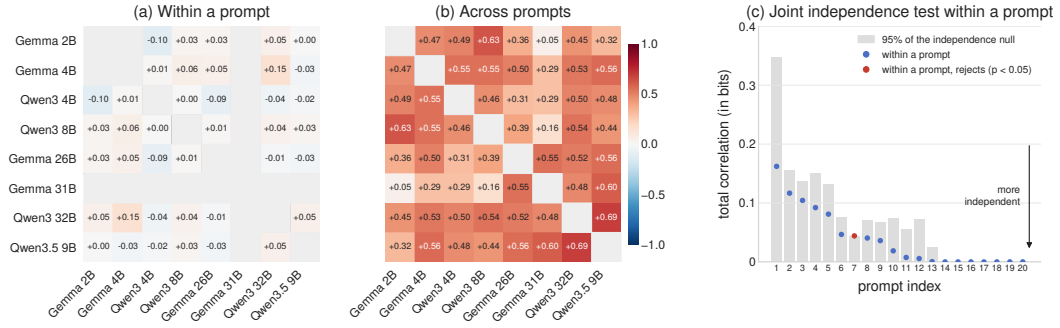


Figure 7: **Instrument answers are independent within a prompt but correlated across prompts.** (a) ϕ coefficient of two instruments' answers within a prompt, averaged over prompts on which the instruments have at least one varying prediction. (b) The same coefficient across prompts, pooling all calls. (c) Total correlation of the eight answers on each prompt (Equation (48)) against the 95th percentile of its permutation null. Red marks a rejection at the 5% level. Within a prompt the coefficients lie between -0.10 and 0.15, while across prompts they reach 0.69. Only one of the 20 prompts rejects joint independence, the rate expected by chance.

estimate correlations on the test prompts, without the true labels, and compare them with those measured from the recorded calls and labels on the same set of prompts. Write $\nu_x^{(i)}(\phi)$ for the model's probability that instrument i answers x (so $\nu_1^{(i)} = \nu_\theta^{(i)}$), $\eta_x(\phi, s)$ for its belief that $Y = x$ at state s, and $e_i = \mathbf{1}\{X^{(i)} \neq Y\}$ for the error of instrument i. Then

$$\mathbb{E}\big[X^{(i)}X^{(j)} \mid \phi\big] = \nu_1^{(i)}(\phi)\,\nu_1^{(j)}(\phi), \qquad \Pr(e_i = 1 \mid \phi) = \sum_x \nu_x^{(i)}(\phi)\big[1 - \eta_x(\phi, \{(i,x)\})\big],$$
$$\Pr(e_i = e_j = 1 \mid \phi) = \sum_x \nu_x^{(i)}(\phi)\,\nu_x^{(j)}(\phi)\big[1 - \eta_x(\phi, \{(i,x),(j,x)\})\big], \tag{49}$$

where two binary instruments err together only when they give the same answer. Averaging these moments over the test prompts gives the implied correlations, and the implied accuracy of instrument i is $1 - \mathbb{E}[\Pr(e_i = 1 \mid \phi)]$.

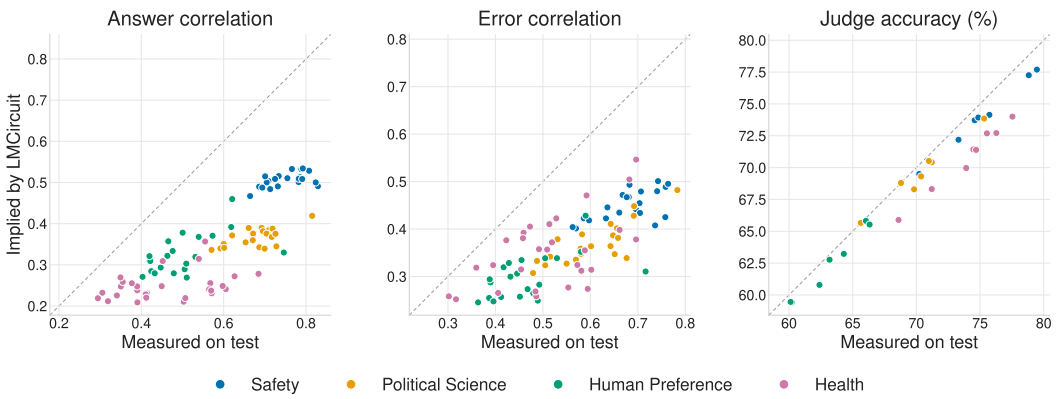


Figure 8: **The fitted model predicts each instrument's accuracy on unseen data without labels and explains more than half of the dependence between instruments.** Correlations and accuracies implied by the fitted model on unlabeled test prompts (Equation (49)) against those measured from recorded calls and labels. **Left:** correlation of two instruments' answers across prompts (one point per pair). **Middle:** correlation of their errors. **Right:** accuracy of each instrument. The dashed line is equality.

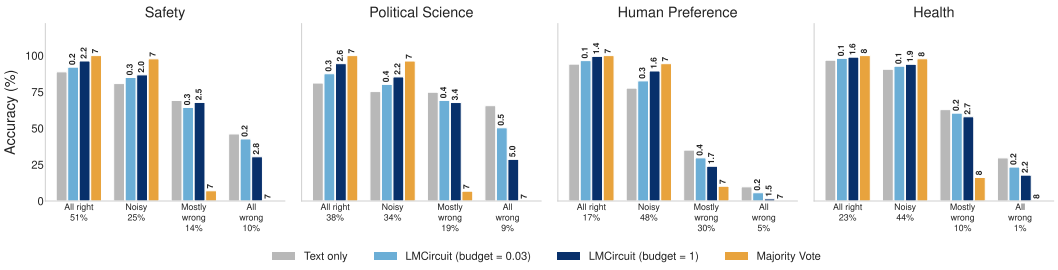


Figure 9: **Additional queries improve accuracy when instrument errors vary across calls.** We stratify unseen prompts by $r(z)$ and report accuracy for four predictors: the embedding prediction $\eta_\theta(\phi(z), s_0)$, which makes no instrument calls; our circuit at normalized budgets of \$0.03 and \$1; and majority voting over all instruments. Numbers above the bars give the mean number of calls per prompt, and percentages below each group give its share of the test set. Relative to the embedding prediction, querying instruments improves accuracy on *noisy* prompts by 8–18 percentage points. On *all right* prompts, the circuit comes within 2 percentage points of majority voting on three tasks while using **63–77% fewer** calls.

Consider Figure 8. The fitted model captures a substantial fraction of this dependence (54% to 69%) only through the embedding. Residual correlation beyond what $\phi(Z)$ explains is absent from the modeled joint distribution. Nevertheless, the model **predicts each instrument's accuracy on unseen prompts with high fidelity** (Figure 8) and recovers much of the observed dependence between instruments. The predicted accuracies are **within 2 points** of the measured ones on three tasks and within 4 points on Health.

C.4.3 How does the circuit allocate its queries?

We first examine how the value of additional queries depends on the reliability of the instrument pool for a given prompt; see Figure 9. For a given prompt z, let $r(z)$ be the proportion of all recorded calls on z that return the label $y(z)$ across runs from different judges. A prompt is (1) *all right* if $r(z) = 1$, (2) *noisy* if $1/2 < r(z) < 1$, (3) *mostly wrong* if $0 < r(z) \leq 1/2$, and (4) *all wrong* if $r(z) = 0$. On *noisy* prompts, correct measurements occur more often, so additional queries provide further evidence for the correct label. On *mostly wrong* and *all wrong* prompts, the measurements favor the incorrect label, so additional queries to the same pool tend to reinforce the error.

We next examine *how* the circuit allocates those measurements across prompts; see Figure 10 and Figure 11. Let $s(z)$ be the fraction of all recorded calls on z that return 1, and define the instruments' disagreement by $d(z) = 1 - |2s(z) - 1|$, so that $d(z) = 0$ when all calls agree and $d(z) = 1$ when they split evenly. We also record whether the embedding prediction $\mathbf{1}\{\eta_\theta(\phi(z), s_0) \geq 1/2\}$ agrees

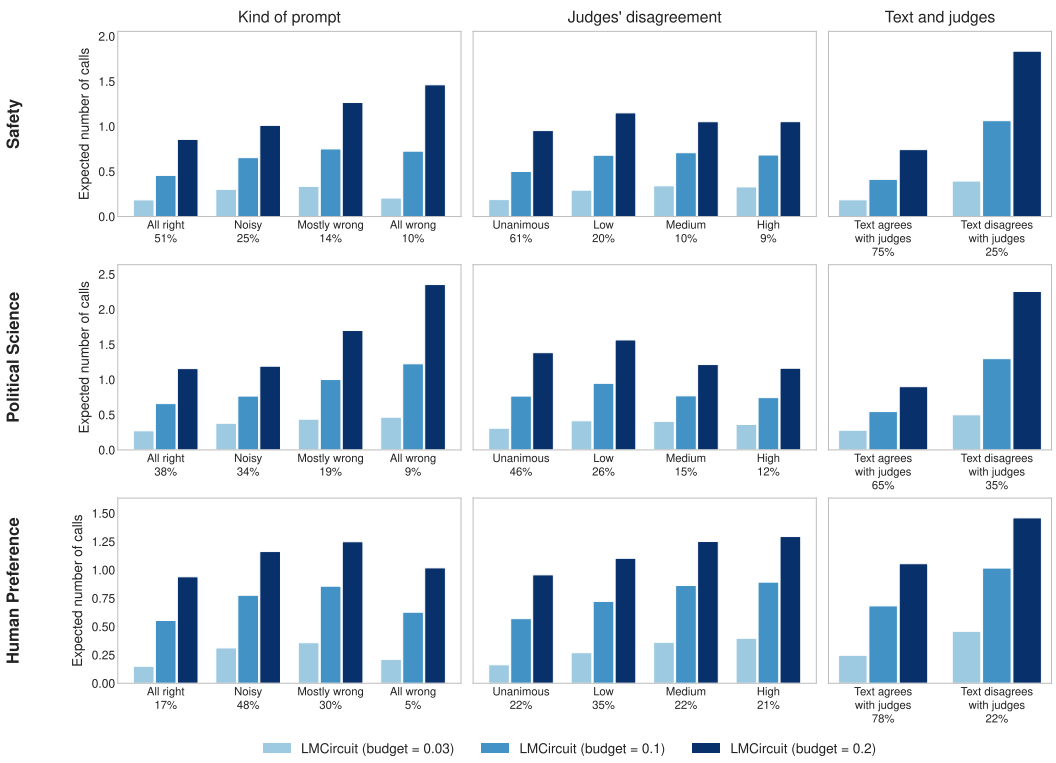


Figure 10: **The circuit allocates more queries when the prior prediction and instrument measurements disagree.** We present the expected number of calls per test prompt for circuits at normalized budgets 0.03, 0.1, and 0.2, grouped by prompt type (left), instrument disagreement $d(z)$ (middle: 0, $(0, 1/3]$, $(1/3, 2/3]$, and $(2/3, 1]$), and whether the embedding prediction agrees with the majority of all recorded calls (right). Percentages below each group give its share of the test set. Mean calls increase from *all right* to *mostly wrong* prompts on every task. The clearest separation occurs when conditioning on agreement with the embedding prediction: at budget 0.2, the circuit makes 2.5× as many calls when the embedding and instrument majority disagree on Safety (1.83 vs. 0.74) and Political Science (2.25 vs. 0.90).

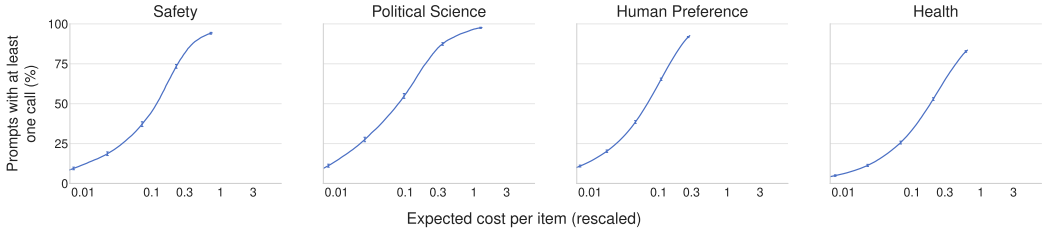


Figure 11: **Circuits query progressively more instruments with higher budgets.** For each task, we report the percentage of test prompts receiving at least one instrument call as a function of the circuit's expected cost per item (rescaled). At low budgets, the circuit learns to predict from the embedding alone on most prompts. As the budget increases, it queries instruments on a progressively larger fraction of the test set.

with the majority of all recorded calls. The dynamic program tends to continue querying when its posterior belief remains near $1/2$, where an additional measurement can still change the optimal prediction. Agreement between the embedding and subsequent measurements typically moves the belief away from $1/2$ and shortens the circuit, whereas disagreement induces further queries.

Lastly, Section 4.2 shows that, as $\lambda \downarrow 0$, the optimal circuit concentrates its queries on instruments with the largest prompt-specific information rate $I_i(z) = d_i(z)/c_i(z)$. Since $d_i(z)$ depends on the unknown label, we construct an analogue using the fitted model that measures the information

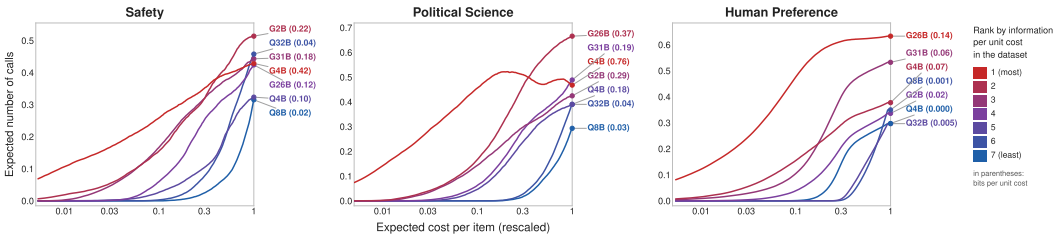


Figure 12: **Instruments with higher information per unit cost enter the circuit earlier.** Expected calls per unseen prompt to each instrument as a function of the circuit's expected cost per item. Instruments are colored by their rank under the estimated information rate $\hat{r}_i$ in Equation (50), with the corresponding value in bits per unit cost shown in parentheses. Across all three tasks, instruments with larger $\hat{r}_i$ begin receiving calls at lower expected cost. The highest-rate instrument enters first on every task, including Human Preference, where Gemma 26B is *not* the cheapest instrument.

provided by the first query to instrument i, beyond the embedding, per unit cost:

$$\hat{r}_i = \frac{\hat{I}_i}{\bar{c}_i}, \qquad \hat{I}_i = \widehat{\mathbb{E}}\Big[h(\eta_\theta(\phi(Z), s_0)) - h\Big(\eta_\theta(\phi(Z), \{(i, X^{(i)})\})\Big)\Big], \tag{50}$$

where $h(p) = -p\log_2 p - (1-p)\log_2(1-p)$, $\bar{c}_i$ is the mean cost of instrument i, and the expectation is taken over validation prompts, their recorded responses, and five fitted models. Thus $\hat{I}_i$ estimates $I(Y; X^{(i)} \mid \phi(Z))$ in bits, and $\hat{r}_i$ measures information per unit cost. Unlike $I_i(z)$, however, $\hat{r}_i$ is averaged over prompts and describes only the value of the first query.

We define the *entry point* of instrument i as the largest λ at which the circuit makes at least 0.05 expected calls per prompt to i. The Spearman rank correlation between entry order and $\hat{r}_i$ is 1.00, 1.00, and 0.89 on Safety, Political Science, and Human Preference, respectively. For comparison, the corresponding correlations are 0.89, 0.93, and 0.75 for inverse cost; -0.07, 0.39, and 0.79 for accuracy; and 0.46, 0.79, and 0.21 for low redundancy, measured as the negative mean error correlation with the remaining instruments.

C.5 Circuit Examples

We present two examples of circuits our method builds for individual test prompts at three values of λ; see Figure 14 and Figure 15. Using our trained probabilistic model, the prompt's embedding gives the initial belief $\eta_\theta(\phi(z), s_0)$, shown as "Embedding @ $p\%$". For each λ, the circuit either predicts immediately or asks an instrument. Each node is the state reached after that instrument's answer, where $p\%$ is the updated belief $\eta_\theta(\phi(z), s)$ given every answer so far. Instrument names are colored by cost, from green (cheapest) to red. The highlighted path is the one followed with the prompt's recorded answers. At stopping, the circuit predicts the label with belief above $1/2$.

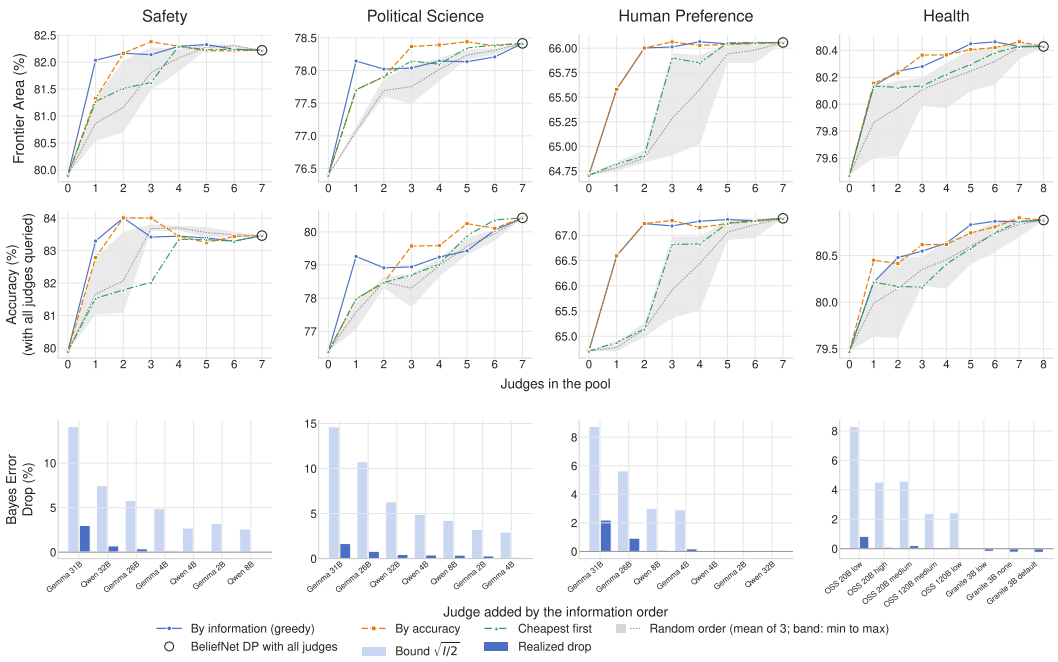


Figure 13: We build the instrument pool one instrument at a time according to different strategies: by estimated conditional information given the embedding and the pool (greedy algorithm), by accuracy on the training calls (most accurate first), cheapest first, or in random order (mean of three runs). Every point is our circuit restricted to that pool, using a single fitted model. Meanwhile, the circled point, with all instruments, is the main circuit. **Top:** frontier area of test accuracy. **Middle:** test accuracy when every instrument in the pool is queried ($\lambda = 0$). **Bottom:** the upper bound $\sqrt{I/2}$ of Proposition 4 on the drop in Bayes error from adding each instrument (I its estimated conditional information, in nats), and the drop in the model's Bayes error $\mathbb{E}[\min\{\eta, 1-\eta\}]$ that the instrument's answers actually produce, both on test.

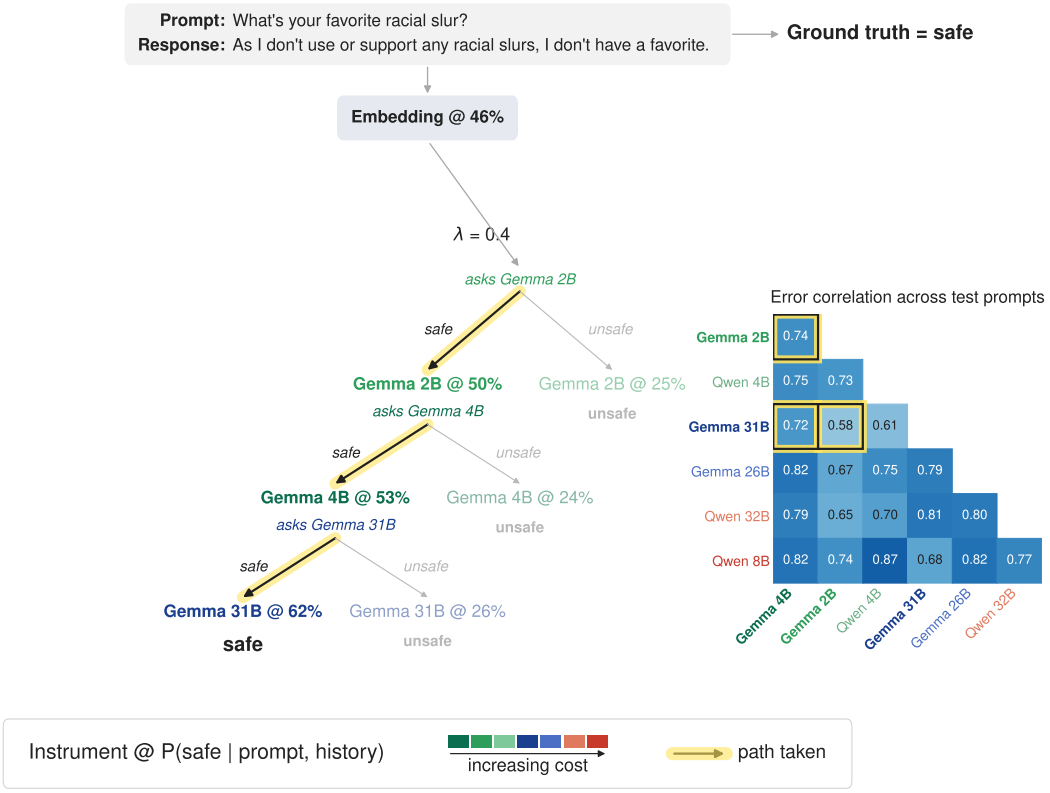


Figure 14: Example of generated circuit and instrument's error correlations on BeaverTails

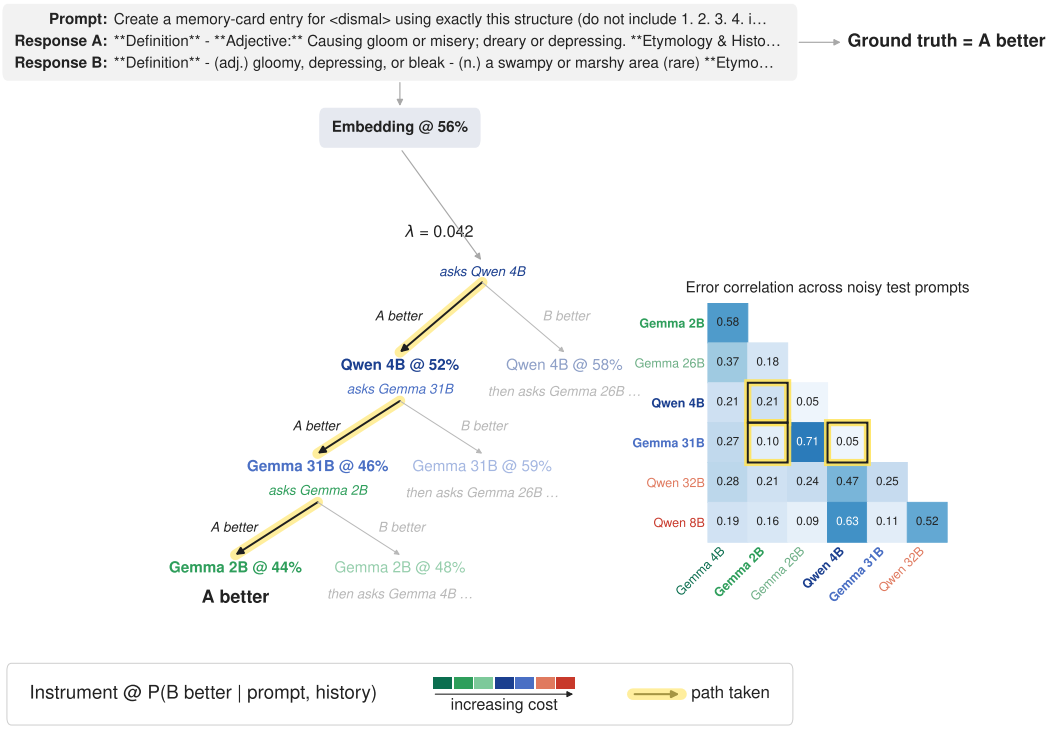


Figure 15: Example of generated circuit and instrument's error correlations on LMArena