COST-OPTIMAL LLM ROUTING WITH LIMITED FEEDBACK UNDER USER SATISFACTION GUARANTEES

Anonymous authors
Paper under double-blind review

ABSTRACT

Inference costs for large language model (LLM) applications are rapidly growing, driven by surging demand and rising infrastructure cost. Users expect high-quality responses, and in commercial settings this is formally codified in Service Level Agreements (SLAs), creating a fundamental tension between cost and quality. Recent progress on cost-aware LLM request routing has shown potential to resolve this tension, but existing approaches rely on complete feedback signals, offline training, extensive per-workload tuning, and most lack SLA guarantees or inference-time adaptivity. We introduce SLARouter, an online routing algorithm that learns a cost-optimal policy from the sparse, one-sided user feedback available in production systems. SLARouter provides theoretical guarantees for both cost optimality and strict SLA compliance. Experiments across a wide range of LLM benchmarks using a mixed-family model zoo show that SLARouter satisfies SLA constraints without the need for per-benchmark tuning, reducing operating cost by an average of $\sim 11\%$ over existing baselines. We further introduce a data-driven method that optimizes the warm-up cost of SLARouter, further reducing the cost for budget-constrained workloads as well as an LLM-judge-based mechanism that ensures smooth predictor training over time.

1 INTRODUCTION

The rapid proliferation of open-weight large language models (LLMs) (GLM-5 Team et al., 2026; IBM Research, 2025; MiniMax et al., 2025; Yang et al., 2025), alongside a growing array of API-based offerings, has left both end users and inference service providers facing an increasingly intractable model selection problem. Compounding this challenge is the fragmented benchmarking landscape. Results are often measured on incomparable tasks under inconsistent conditions, making it difficult to determine which model is best suited for a given workload (Naveed et al., 2025). This leaves practitioners resorting to educated guessing or expensive trial-and-error.

Against this backdrop, *LLM routing*, i.e., the problem of dynamically selecting an appropriate model for each incoming request, has emerged as a popular technique to improve user satisfaction and reduce inference costs (Chen et al., 2024b; Huang et al., 2025; Ong et al., 2025; Stripelis et al., 2024). Instead of committing to a single model, routing frameworks maintain a *model zoo* and dispatch each request to whichever model best balances quality and cost for that particular query. As model families often span parameter counts from under a billion to over a hundred billion, the cost difference between the smallest and largest model in a zoo can easily exceed an order of magnitude (Samsi et al., 2023), making intelligent routing essential for sustainable operations.

Cost efficiency is especially pressing as LLMs are increasingly deployed in a wide range of applications, where sequences of model calls compound energy expenditure. Emerging AI regulation, such as Article 95 of the EU AI Act (European Parliament and Council of the European Union, 2024), further asks for energy monitoring and compliance with sustainable computing best practices. Although there are existing studies on cost-aware routing and first steps towards cost-optimal routing, the following core challenges still remain.

The Need for Formal Service Level Guarantees. In commercial deployments, cost savings alone are insufficient. Providers and their customers typically codify quality requirements in *Service Level Agreements* (SLAs). For LLM inference, an SLA might specify a certain task success rate for an AI model over time. Early routing work (Aggarwal et al., 2024; Chen et al., 2024a; Ding et al.,

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.

Table 1: Related work. *Cost-Aware* = supports cost control. *SLA Guar.* = provides formal SLA guarantees. *Online* = can learn at inference time. *Observ. Data* = can be trained from one-sided observational data.

Method	Zoo Size	Cost-Aware	SLA Guar.	Online	Observ. Data	Source
Cost-Aware without Optimality Guarantees						
RouteLLM	2	✓	×	×	×	Ong et al. (2025)
Hybrid LLM	2	✓	×	×	×	Ding et al. (2024)
AutoMix	2	✓	×	×	×	Aggarwal et al. (2024)
FrugalGPT	> 2	✓	×	×	×	Chen et al. (2024a)
xRouter	> 2	✓	×	∼[†]	×	Qian et al. (2025)
Causal LLM Routing	> 2	✓	×	×	✓	Tsiourvas et al. (2025)
Formal Optimality Guarantees Requiring Complete User Feedback						
CARROT	> 2	✓	∼[‡]	×	×	Somerstep et al. (2025)
PROTEUS	> 2	✓	✓	∼[†]	×	Bhatti et al. (2026)
MESS+	> 2	✓	✓	✓	×	Woisetschläger et al. (2025)
Formal Optimality Guarantees under One-Sided and Sparse User Feedback						
SLARouter	> 2	✓	✓	✓	✓	**This Paper**

[†] Online RL training with offline deployment (no adaptation at inference time). [‡] Minimax rate-optimality guarantee, not an SLA satisfaction rate.

2024; Ong et al., 2025) demonstrated that cost-aware routing between a strong and a weak model is practically effective, but does not provide formal guarantees that the routed system will meet a prescribed quality threshold. More recent efforts have begun to close this gap by introducing minimax rate-optimal guarantees in a two-model setting (Somerstep et al., 2025) or formal SLA compliance to larger model zoos (Bhatti et al., 2026; Woisetschläger et al., 2025), albeit most without adaptation at inference time. At this point in time, there are only very few works that update their routing policies as user requests arrive, especially in combination with SLA guarantees.

The Feedback Bottleneck. A practical obstacle in virtually all existing routing approaches is that they assume the availability of *reliable, timely, and complete* quality signals. In real-world systems, however, explicit user feedback is sparse. For example, requests that may receive any signal at all have been reported anecdotally low (Krauth et al., 2025; Zhao et al., 2026). When the feedback arrives, it is usually *one-sided*, i.e., it pertains to exactly *one* model response. So, the performance of all other models remains unknown. Without addressing this feedback bottleneck, even a theoretically sound routing policy will degrade in practice. We therefore need a routing framework that adapts to user preferences and workload shifts over time based on one-sided observational feedback while providing SLA guarantees.

Research Question. *Can we build an LLM router that provides formal SLA guarantees and operates cost-optimally, relying only on sparse one-sided observational feedback available in practice?*

Contributions. We introduce **SLARouter**, a practical and scalable LLM routing algorithm that synthesizes the lessons of prior work into a unified framework capable of operating under production-realistic conditions. SLARouter makes the following novel contributions:

- **Online routing from sparse one-sided observational feedback.** SLARouter is an adaptive routing algorithm that learns a cost-optimal routing policy from one-sided and sparse user feedback as it is typically available in production systems. We show that SLARouter is up to $\sim 11\%$ cheaper compared to existing cost-aware routing techniques while providing on par routing performance. While SLARouter proves to be effective in systems with low user feedback rates, we extend our system with an LLM-based judge mechanism (Zheng et al., 2023) to improve predictor updates.
- **Formal SLA guarantees under observational data.** We provide theoretical guarantees for cost-optimality under one-sided sparse user feedback while maintaining a required user satisfaction rate over time. Our approach is based on a novel extension to the Lyapunov drift-plus-penalty framework (Neely, 2022). With our performance analysis, we demonstrate that we can effectively learn from observational data and achieve virtual queue stability in settings with highly sparse feedback rates.

Related Work. Table 1 compares SLARouter to existing LLM routing algorithms. Cost-aware routers such as RouteLLM (Ong et al., 2025), Hybrid LLM (Ding et al., 2024), AutoMix (Aggarwal et al., 2024), and FrugalGPT (Chen et al., 2024a) reduce inference expenditure effectively but offer no formal quality guarantees. CARROT (Somerstep et al., 2025) introduces rate-optimal guarantees but does not adapt online. PROTEUS (Bhatti et al., 2026) scales SLA compliance to larger model zoos but requires offline training, limiting adaptation to distributional shifts. MESS+ (Woisetschläger et al., 2025) combines online adaptation with formal SLA guarantees, but assumes complete, balanced, and timely feedback for all candidates that is rarely satisfied in production. Causal LLM Routing is

the only prior method that learns from observational data, but does not support online adaptation or SLA guarantees (Tsiourvas et al., 2025). SLARouter is the first to satisfy all four criteria, specifically, cost-awareness, formal SLA guarantees, fully online operation, and learning from the one-sided sparse feedback available in real systems.

2 SLAROUTER: PRACTICAL AND SCALABLE LLM REQUEST ROUTING WITH SERVICE LEVEL GUARANTEES BASED ON OBSERVATIONAL DATA

2.1 PROBLEM FORMULATION

We consider a model zoo of M LLMs, indexed by $m \in \{1, \dots, M\}$, that serve incoming user requests $t \in \{1, \dots, T\}$. For each request, our goal is to select a model that minimizes inference cost while guaranteeing that a minimum fraction α of requests receive a satisfactory response over time.

Inference Cost (Objective). Each model m incurs an inference cost $C_{m,t}$ per request t, e.g., API cost as \$/1M tokens. Model families routinely span orders of magnitude in cost, so routing intelligently instead of defaulting to the largest model can yield substantial savings.

Service Level Agreement (Constraint). In our SLA, we define $\alpha \in (0, 1)$ as the minimum required user satisfaction rate over time. Violating this threshold is typically associated with contractual penalties, making strict compliance non-negotiable. In practice, α should be set slightly above the contractually agreed threshold to absorb estimation noise and provide a compliance safety margin.

Control Problem. Combining objective and constraint, we aim to solve:

$$\min_{\{y_{m,t}:\forall t,m\}} \quad \frac{1}{T}\sum_{t=1}^{T}\sum_{m=1}^{M}\mathbb{E}\left[y_{m,t}C_{m,t}\right], \tag{1a}$$

$$\text{s.t.} \quad \frac{1}{T}\sum_{t=1}^{T}\sum_{m=1}^{M}\mathbb{E}\left[y_{m,t}s_{m,t}\right] \geq \alpha, \tag{1b}$$

$$\sum_{m=1}^{M} y_{m,t} = 1, \quad \forall t \in \{1, \dots, T\}, \tag{1c}$$

$$y_{m,t} \in \{0, 1\}, \quad \forall t \in \{1, \dots, T\},\ m \in \{1, \dots, M\}, \tag{1d}$$

where $y_{m,t} = 1$ if model m serves request t and 0 otherwise, and $s_{m,t} \in \{0, 1\}$ denotes whether that response was satisfactory.

Challenges. Our work is primarily motivated by the observation of how users provide feedback in systems like ChatGPT, Claude, or Microsoft Copilot. Users have the option to give a thumbs up or down, whenever they would like to provide feedback. Overall feedback signals are extremely sparse (Krauth et al., 2025; Zhao et al., 2026). Since we usually only query one model for a user request, this typically leaves us with *one-sided* and often *imbalanced data*, i.e., we do not know the true value of $s_{m,t}$ for most requests and models. Consequently, we have to deal with *unknown performance statistics* for all other models that were not queried and did not receive feedback. Furthermore, chat-like *requests arrive sequentially* in a streaming fashion without the knowledge of *statistics of future requests.* Hence, we cannot solve problem (1) directly in one shot over the entire time horizon. An *online* solution that routes the model *per request* is needed.

Algorithm 1: SLARouter – Cost-optimal & SLA compliant routing with sparse and one-sided user feedback

Initialize $\mathcal{D} \leftarrow \emptyset$, $Q_1 \leftarrow 0$, predictor parameter $\mathbf{z}$;
for $t \leftarrow 1$ **to** T **do**
  Compute $p_t \leftarrow \min\left(1, c/\sqrt[4]{t}\right)$;
  Sample $X_t \sim \text{Bernoulli}(p_t)$;
  if $X_t = 1$ *or* $t = 1$ **then**
    `// Only sample one model`
    Sample $\tilde{m} \sim \text{Uniform}(\{1, 2, \dots, M\})$;
    $y_{\tilde{m},t} \leftarrow 1, y_{m',t} \leftarrow 0, \forall m' \neq \tilde{m}$;
    if $s_{\tilde{m},t} \neq \emptyset$ **then**
      `// User provides feedback`
      $\mathcal{D} \leftarrow \mathcal{D} \cup (\mathbf{a}_t, s_{\tilde{m},t}, \tilde{m})$;
      Sample mini-batch $\mathcal{B}$ from $\mathcal{D}$;
      $\mathbf{z}_{t+1} \leftarrow \mathbf{z}_t - \eta_t \nabla F(\mathbf{z}_t, \mathcal{B})$;
    else
      `// User does not provide feedback`
      $\mathbf{z}_{t+1} \leftarrow \mathbf{z}_t$;
  else
    Predict request satisfaction probability $\hat{s}_{m,t}, \forall m$;
    $\tilde{m} \leftarrow \arg\min_m \{VC_{m,t} + Q_t(\alpha - \hat{s}_{m,t})\}$;
    $y_{\tilde{m},t} \leftarrow 1, y_{m',t} \leftarrow 0, \forall m' \neq \tilde{m}$;
    $\mathbf{z}_{t+1} \leftarrow \mathbf{z}_t$;
  `// Virtual queue update`
  if $s_{\tilde{m},t} \neq \emptyset$ **then**
    `// Use user feedback for` Q `update`
    $Q_{t+1} \leftarrow \max\{0, Q_t + \alpha - s_{\tilde{m},t}\}$;
  else
    `// Use predictor estimate for` Q `update`
    $Q_{t+1} \leftarrow \max\{0, Q_t + \alpha - \hat{s}_{\tilde{m},t}\}$;

2.2 PROPOSED SLAROUTER ALGORITHM

The Lyapunov drift-plus-penalty framework (Neely, 2022) gives a useful starting point for our problem. It operates in an online fashion without requiring knowledge of the underlying request

or feedback distribution. It can provide provable guarantees on time-average performance despite unknown future statistics. This stability condition emerges organically from the Lyapunov structure instead of being imposed as an ad hoc heuristic. However, the standard framework in (Neely, 2022) as well as its use in (Woisetschläger et al., 2025) still assumes full feedback signals for all the requests. In this work, we present a *novel extension* to support sparse and imbalanced observations, ensuring that models with insufficient feedback are not perpetually neglected. The overall procedure is in Algorithm 1. We explain the details as follows.

Methodology. SLARouter operates in two phases, *exploration* and *exploitation*, controlled by a random process $X_t \sim \text{Bernoulli}(p_t)$, where $p_t = \min\left(1, {}^{c}/\sqrt[4]{t}\right)$ and $c \in \mathbb{R}^+$, which is a hyperparameter to control the exploration probability. The exploration probability decays over time, gradually shifting weight toward exploitation as the predictor matures. Central to both phases is a *satisfaction predictor*, which is a multi-label classifier with one output head per candidate model that independently estimates the probability (denoted by $\hat{s}_{m,t}$) that a given request t would satisfy a user if served by model m.

Note that the predictor informs the routing decision but is not responsible for it. The decision is determined by our per-request optimization described below. Once a response is sent to the user, they can provide feedback to indicate whether it was satisfactory or not. As we aim to ensure SLA compliance, we use a *virtual queue* to capture all violations (negative feedback) over time. The queue length Q_t affects the routing decision as more violations will pressure the algorithm to choose a more capable and expensive model, and vice versa.

Feedback Availability. Whenever a request is sent to an LLM, the system may or may not receive user feedback about whether the response is satisfactory. This same principle applies to both the exploration and exploitation phases. We do not distinguish between real user and synthetic judge feedback (ablation provided in Section 5.2). If feedback is unavailable (from either the user or the LLM judge) for request t and model m, we write $s_{m,t} = \emptyset$.

Predictor. The predictor (with trainable parameter vector $\mathbf{z}$) must handle the key structural difficulty of *missing labels* across heads, which arises from the feedback stream since only the explored model receives a label at each step. The overall objective of predictor training is a standard binary cross-entropy loss:

$$F(\mathbf{z}) := -\mathbb{E}_{\mathbf{a}_t \sim \mathcal{A}}\left[\frac{1}{M}\sum_{m=1}^{M}\left(s_{m,t}\log\hat{s}(\mathbf{z},\mathbf{a}_t)_m + (1-s_{m,t})\log(1-\hat{s}(\mathbf{z},\mathbf{a}_t)_m)\right)\right] + \frac{\mu}{2}\|\mathbf{z}\|^2, \quad (2)$$

where $\mu > 0$ is a weight-decay coefficient, $\mathbf{a}_t$ is the input of the t-th request, and $\hat{s}(\mathbf{z},\mathbf{a}_t)_m = \hat{s}_{m,t}$ for predictor parameter $\mathbf{z}$ and request input $\mathbf{a}_t$.

The *uniqueness* of SLARouter's predictor is that it is trained only using one-sided feedback, i.e., feedback from only a single model that is chosen to serve the user. To facilitate this, all the requests with feedback collected from exploration steps are saved in a *predictor training dataset* $\mathcal{D}$. In each exploration step, if feedback is collected, this new sample $(\mathbf{a}_t, s_{\tilde{m},t}, \tilde{m})$ is added to $\mathcal{D}$, where $\tilde{m}$ is the randomly chosen model during exploration; then, a step of stochastic gradient descent (SGD) with a mini-batch $\mathcal{B}$, sampled from $\mathcal{D}$, is performed to update the predictor's parameter $\mathbf{z}$. The mini-batch size $|\mathcal{B}| := \min\{|\mathcal{D}|; B\}$, where $|\cdot|$ denotes the cardinality of the set and B is the default mini-batch size once $|\mathcal{D}| \geq B$. We denote the mini-batch gradient by $\nabla F(\mathbf{z},\mathcal{B})$, which is computed by

$$\nabla F(\mathbf{z},\mathcal{B}) = -\nabla_{\mathbf{z}}\left[\frac{1}{|\mathcal{B}|}\sum_{(\mathbf{a}_t, s_{\tilde{m},t}, \tilde{m})\in\mathcal{B}}\left(s_{\tilde{m},t}\log\hat{s}(\mathbf{z},\mathbf{a}_t)_{\tilde{m}} + (1-s_{\tilde{m},t})\log(1-\hat{s}(\mathbf{z},\mathbf{a}_t)_{\tilde{m}})\right)\right] + \mu\mathbf{z}. \quad (3)$$

Due to the one-sided feedback, it is possible that some LLMs do not have samples in a mini-batch. In that case, the predictor parameters directly connected to the outputs corresponding to those models are not updated for that specific mini-batch SGD step, because the gradient with respect to those parameters is zero as there is no loss term corresponding to those models in (3).

When the feedback availability follows an IID distribution across requests and models, we can easily see that $\nabla F(\mathbf{z},\mathcal{B})$ is an unbiased estimate of the full gradient $\nabla F(\mathbf{z})$. Thus, we have addressed the limited feedback problem by *drawing an equivalence between random sampling for SGD, random model selection for exploration, and (random) feedback availability*.

Routing Decision. As the predictor improves, routing decisions rely increasingly on its predictions. The predictor scores $\hat{s}_{m,t}$ and the virtual queue Q_t jointly determine the *per-request decision problem*:

$$\min_{\{y_{m,t}:\forall m\}} \quad V\cdot\sum_{m=1}^{M} y_{m,t}C_{m,t} + Q_t\left(\alpha - \sum_{m=1}^{M} y_{m,t}\hat{s}_{m,t}\right), \quad (4a)$$

$$\text{s.t.} \quad \text{Constraints (1c), (1d)}. \tag{4b}$$

The update rule for Q_t is defined in (5) below. Parameter $V > 0$ controls the trade-off between cost efficiency and convergence speed. A smaller V yields faster SLA compliance at higher initial operating cost, while a larger V prioritizes cost savings at the expense of slower convergence.

Handling Feedback Sparsity. Updating Q_t in the standard way requires a feedback signal at every step, but explicit user feedback often arrives at low rates in production systems (Krauth et al., 2025; Zhao et al., 2026). When $s_{\tilde{m},t}$ is unavailable, we substitute it with the predictor score $\hat{s}_{\tilde{m},t}$ as a proxy. Q_t evolves as follows:

$$Q_{t+1} = \begin{cases} \max\{0, Q_t + \alpha - s_{\tilde{m},t}\}, & \text{if } s_{\tilde{m},t} \neq \emptyset \\ \max\{0, Q_t + \alpha - \hat{s}_{\tilde{m},t}\}, & \text{if } s_{\tilde{m},t} = \emptyset \end{cases}. \tag{5}$$

This substitution keeps our Lyapunov analysis intact, provided that the predictor estimate is sufficiently accurate. This is a condition enforced by the exploration schedule, which guarantees continued predictor improvement and adaptation to the user throughout deployment.

3 Performance Analysis

We present the theoretical performance analysis in this section. The full proofs are in Appendix A.

3.1 Predictor Training

Assumption 1. The predictor training objective $F(\mathbf{z})$ is L-smooth and satisfies the Polyak–Łojasiewicz (PL) condition with parameter $\mu > 0$ (and $\mu \leq L$), i.e., $\frac{1}{2}\|\nabla F(\mathbf{z})\|^2 \geq \mu\big(F(\mathbf{z}) - F_{\min}\big), \forall \mathbf{z}$, where $F_{\min} := \min_{\mathbf{z}} F(\mathbf{z})$. Its single-sample stochastic gradient, denoted by $\nabla f(\mathbf{z}, \mathbf{a}, m)$, is unbiased with bounded variance σ^2, i.e., $\mathbb{E}\big[\nabla f(\mathbf{z}, \mathbf{a}, m) \mid \mathbf{z}\big] = \nabla F(\mathbf{z})$ and $\mathbb{E}\big[\|\nabla f(\mathbf{z}, \mathbf{a}, m) - \nabla F(\mathbf{z})\|^2 \mid \mathbf{z}\big] \leq \sigma^2, \forall \mathbf{z}, m$. The request $\mathbf{a}_t \sim \mathcal{A}$ is IID across t.

Assumption 1 is standard in SGD convergence analysis. Let k denote the SGD steps until request t. Note that both k and the dataset size $|\mathcal{D}|$ are random variables due to the randomness of exploration. A unique feature of our setup is that there exist two distinct sources of gradient error:

$$\underbrace{\nabla F(\mathbf{z})}_{\substack{\text{all } M \text{ models} \\ \text{over full dataset space}}} \xrightarrow{|\mathcal{D}| \text{ samples available}} \underbrace{\nabla \hat{F}(\mathbf{z})}_{\substack{\text{variance induced by} \\ \text{limited feedback: } \mathcal{O}(\sigma^2/|\mathcal{D}|)}} \xrightarrow{B \text{ batch size}} \underbrace{\nabla F(\mathbf{z}, \mathcal{B})}_{\substack{\text{variance induced by} \\ \text{stochastic gradient sampling: } \mathcal{O}(\sigma^2/B)}}.$$

The *empirical* "full" gradient on the currently collected dataset $\mathcal{D}$, denoted by $\nabla \hat{F}(\mathbf{z}_k)$, is computed over the $|\mathcal{D}|$ available feedback samples, while the update gradient $\nabla F(\mathbf{z}, \mathcal{B})$ further subsamples B samples per SGD step from $\mathcal{D}$. Larger B and $|\mathcal{D}|$ reduce both error sources, as shown in Lemma 1.

Lemma 1. *Choosing* $\eta_k = \min\left\{\frac{4}{\mu(k+1)}, \frac{1}{2L}\right\}$, *after* $k \geq 1$ *steps of predictor training using SGD with the loss function defined in* (2)*, we have*

$$\mathbb{E}\left[F(\mathbf{z}_k) \middle| k, \mathcal{D}\right] - F_{\min} \leq \mathcal{O}\left(\frac{1}{k^2} + \frac{\sigma^2}{|\mathcal{D}|} + \frac{\sigma^2}{kB}\right). \tag{6}$$

The proof involves a unique way of decomposing and bounding the two error sources. Lemma 1 captures the two distinct sources of gradient error that jointly govern the rate at which $F(\mathbf{z}_k)$ approaches its true minimum $F_{\min}$. The first and the last terms are standard for SGD. The second term captures the variance induced by the finite size of dataset $\mathcal{D}$. The diminishing step size $\eta_k = \Theta(1/k)$ is the standard theoretical technique for exact convergence to $F_{\min}$. In practice, we found that a constant rate can already be sufficient for good performance (see Section B.3).

Next, we focus on the total expectation of $F(\mathbf{z}_k)$, starting with an assumption on feedback availability.

Assumption 2. For every request t, user feedback is available IID with a fixed probability $\phi \in (0, 1]$.

In Algorithm 1, we always have $k = |\mathcal{D}|$, as we only train the predictor when new feedback is available. We note that this assumption can be relaxed to support alternative schedules. With the exploration probability of $p_t = \min\left(1, c/\sqrt[4]{t}\right)$, we then have $\mathbb{E}[k] = \mathbb{E}[|\mathcal{D}|] = \Theta(t^{3/4}\phi)$. Taking total expectation of the bound in Lemma 1 and noting $k = |\mathcal{D}|$, we then have $\mathbb{E}\left[F(\mathbf{z}_k)\right] - F_{\min} \leq \mathcal{O}\left((1+\sigma^2)/t^{3/4}\phi\right)$.

Considering characteristics of the cross-entropy loss and applying Markov's inequality, we have the following lemma on satisfaction estimation error that is used as a basis for subsequent results.

Lemma 2. *Under the same condition as for Lemma 1, there exists some positive $\beta \leq M$, so that*

$$\mathbb{E}\left[\max_m |\hat{s}_{m,t} - s_{m,t}|\right] \leq \beta F_{\min} + \mathcal{O}\left(\frac{\beta(1+\sigma^2)}{t^{3/4}\phi}\right); \tag{7}$$

$$\max_m |\hat{s}_{m,t} - s_{m,t}| \leq \beta F_{\min} + \mathcal{O}\left(\frac{\beta(1+\sigma^2)}{t^{3/4}\phi\delta}\right), \textit{with probability } 1-\delta. \tag{8}$$

This shows the estimation error of user satisfaction is upper-bounded at high probability with an error floor of $\beta F_{\min}$, where the last term vanishes as t gets large. The theory uses $\beta = M$ in the proof, while in practice β can be much smaller than M.

3.2 SLA Satisfaction

Assumption 3. We assume that $\beta F_{\min} < \frac{1}{3}$, and there exists at least one model m such that $\Pr\{s_{m,t} = 1\} \geq q > \alpha$ and $(1-\phi)\beta F_{\min} < q - \alpha$, for some $q \in (\alpha, 1]$.

This assumption states that the predictor architecture is capable enough so that an optimally trained predictor has a small loss. It also states that there is at least one model for which the satisfaction probability is greater than α, since otherwise the problem becomes infeasible. The last condition weakens as feedback becomes denser and vanishes entirely in the full-feedback case $\phi = 1$. We also assume that the conditions in Lemmas 1 and 2 hold in the following.

Theorem 1. *Let* $\rho := \frac{q-\alpha-(1-\phi)\beta F_{\min}}{2} > 0$, $\psi := \Theta\left(\left(\frac{\beta(1+\sigma^2)}{\phi\,\rho\left(\frac{1}{3}-\beta F_{\min}\right)}\right)^{\frac{4}{3}}\right)$, $\gamma := \max\{\alpha\psi;\, 3V\Delta_C\}$. *For any $t \geq 1$, we have these upper bounds on the virtual queue length:*

$$\mathbb{E}[Q_t] \leq \gamma + \mathcal{O}\left(\sqrt{t}\right) \quad \textit{and} \quad \tfrac{1}{t}\textstyle\sum_{\tau=1}^{t}\mathbb{E}[Q_\tau] \leq \mathcal{O}\left(\gamma + 1/\rho\right), \tag{9}$$

where $\Delta_C := C_{\max} - C_{\min}$, and $C_{\max}$ and $C_{\min}$ are the maximum and minimum costs, respectively.

The proof of Theorem 1 is based on a unique way of analyzing the terms involving feedback availability and the two variants of queue updates in (5). The feedback probability ϕ appears in the denominator in the definition of ψ (and hence γ), which verifies that the queue tends to be longer when the feedback rate is low. Based on (5) and Lemma 2, we obtain the following corollary.

Corollary 1. *We have the following upper bound on the time-averaged constraint violation:*

$$\begin{aligned}\alpha - \tfrac{1}{T}\textstyle\sum_{t=1}^{T}\sum_{m=1}^{M}\mathbb{E}[y_{m,t}s_{m,t}] &\leq \tfrac{\mathbb{E}[Q_{T+1}]}{T} + (1-\phi)\beta F_{\min} + \mathcal{O}\Big(\tfrac{(1-\phi)\beta(1+\sigma^2)}{\phi\, T^{3/4}}\Big) \\ &\leq \mathcal{O}\Big(\tfrac{\gamma}{T} + \tfrac{1}{\sqrt{T}}\Big) + (1-\phi)\beta F_{\min}. \end{aligned} \tag{10}$$

As $T \to \infty$, the constraint violation approaches the error floor $(1-\phi)\beta F_{\min}$. When the predictor is more accurate or the feedback rate is high, the constraint violation becomes less severe. For full feedback with $\phi = 1$, we recover the result in Woisetschläger et al. (2025). If full SLA compliance at level α' is required after a finite number of requests T_0, one may set α slightly above α' such that the upper bound is less than $\alpha - \alpha'$ after T_0 steps.

3.3 Cost Optimality

Theorem 2. *For $\{y_{m,t} : \forall m, t\}$ obtained from the SLARouter algorithm (Algorithm 1), there exists a learning rate schedule $\{\eta_k : \forall k\}$ such that*

$$\mathbb{E}\left[\tfrac{1}{T}\textstyle\sum_{t=1}^{T}\sum_{m=1}^{M} y_{m,t}C_{m,t}\right] \leq C^{\mathrm{OPT}} + \mathcal{O}\Big(\tfrac{\beta}{\sqrt[4]{T}} + \beta F_{\min} + \tfrac{1}{V}\Big), \tag{11}$$

where C^{OPT} is the optimal solution to (1) *obtained from an optimal stationary policy that has full statistical knowledge of requests $t \in \{1, \ldots, T\}$.*

The proof bounds the drift-plus-penalty expression and incorporates Lemma 2 and Theorem 1. The optimality gap gets smaller as T is larger, up to the irreducible predictor error $\beta F_{\min}$ and the $1/V$ term. The parameter V governs the trade-off between cost optimality and SLA satisfaction, where a larger V tightens the optimality gap but slows constraint satisfaction, and vice versa. The residual term $\beta F_{\min}$ reflects the irreducible predictor error. When the predictor is expressive, $F_{\min}$ is small.

4 PRACTICAL CONSIDERATIONS

Positive-Class Reweighting. In practice, the distribution of user feedback is rarely balanced. Users tend to provide predominantly negative feedback to indicate whenever the LLM does not serve their requests sufficiently well. Without accounting for this potential imbalance, the predictor might converge to a trivial constant predictor in the worst case. To counteract this, we use per-model positive-class weights to upweight sparse positive feedback. For each model m, we maintain running counts of positive and negative observations across all collected feedback and compute $p_m = {}^{N_m^-}/_{N_m^+}$, where N_m^+ and N_m^- denote the number of positive and negative feedback samples for model m in $\mathcal{D}$. At each SGD step, p_m is applied as the positive-class weight for the corresponding output in the masked BCE loss over the mini-batch B. This rescales the contribution of the minority class proportionally, ensuring the predictor remains discriminative regardless of label skew.[1]

Dynamic Adaptation of V. The parameter V controls the trade-off between cost efficiency and convergence speed towards the satisfaction target α. A small V prioritizes rapid SLA compliance by routing to expensive models early, and vice versa. From our theoretical analysis, we see that V depends on the cost spread between largest and smallest model in a zoo, making manual tuning of V fragile. This dependency directly follows from Theorem 1. Instead, we adapt V with a data-driven mechanism $V = (Q_{\max} \cdot \varepsilon)/\Delta C$, where $Q_{\max}$ is the maximum tolerated queue length, $\varepsilon > 0$ is a cost-sensitivity parameter, and ΔC is the observed mean cost difference between the cheapest and most expensive model. Intuitively, a wider cost spread yields a smaller V, lowering the pressure toward cheaper models and allowing for timely SLA convergence. This removes a hard-to-tune hyperparameter and improves the adaptivity of SLARouter. We demonstrate the practical benefit in Section 5.2.

5 EXPERIMENTS

We underpin our theoretical results with an extensive empirical evaluation across numerous LLM benchmarks. Our code is available on GitHub[2]. Details on reproducibility are in the appendix.

5.1 DESIGN

Model Zoo & Benchmarks. We construct our model zoo from the Qwen 3.5 family (Qwen Team, 2026; Yang et al., 2025), specifically two dense models (2B, 9B) and two mixture-of-experts models (35B, 122B) as well as the 70B Llama 3.3 model (Dubey et al., 2024; Meta AI, 2024). Benchmarks span reasoning (GSM8K, MMLU, SocialIQA, WinoGrande) (Cobbe et al., 2021; Hendrycks et al., 2021; Sakaguchi et al., 2020; Sap et al., 2019), scientific knowledge (ARC Challenge, ARC Easy, SciQ, GPQA) (Clark et al., 2018; Rein et al., 2024; Welbl et al., 2017), reading comprehension (BoolQ, LAMBADA) (Clark et al., 2019; Paperno et al., 2016), and agentic tasks (ACPBench) (Kokel et al., 2025).

Predictor. We use a ModernBERT-based encoder (Warner et al., 2025) with a multi-label classification head along with SGD, a learning rate of 0.006, a weight decay of 0.01, a minibatch size of $B = 16$, and set the exploration probability parameter $c = 1.0$, if not specified otherwise.

Service Level Agreement. We define α per benchmark within the capabilities of the model zoo (75^{th} performance percentile) and set $V = 0.00001$ to ensure timely SLA compliance. We explore data-driven V (Section 4) with $Q_{\max} = 30$ and $\varepsilon = 0.001$, which is desirable for settings with high user satisfaction requirements.

Objective. We minimize the operating cost (in USD) per task under constraint α, and report the model call ratio per benchmark to characterize routing behavior.

Observational Data. We set the user feedback rate to 0.2 for all experiments. This is primarily to allow for sufficient training data for the CARROT and Causal Router classifiers. Below this rate a

[1] We note that in our benchmark evaluation we do not observe notable imbalance. However, in production deployments imbalance can be significant, making this correction an important safeguard.

[2] For the review period our code is available via an anonymous Git repository: `https://anonymous.4open.science/r/slarouter-F044`

Table 2: Main results. SLARouter is the cheapest routing technique on average that always guarantees SLA compliance (α). Green indicates the router or model satisfies the SLA (α), underline indicates the cheapest compliant single model and **bold** the cheapest compliant router. We report the standard error across 3 seeds. CARROT and Causal Router results include offline classifier training costs.

Method	ACPBench (α=69%)			ARC Challenge (α=56%)			ARC Easy (α=82%)		
	Operating Cost ($)	Request. Satisf. (%)	Model Call Ratio (2B/9B/35B/70B/122B)	Operating Cost ($)	Request. Satisf. (%)	Model Call Ratio (2B/9B/35B/70B/122B)	Operating Cost ($)	Request. Satisf. (%)	Model Call Ratio (2B/9B/35B/70B/122B)
2B only	\$0.077	44.7	100%/0%/0%/0%/0%	\$0.006	38.2	100%/0%/0%/0%/0%	\$0.010	70.6	100%/0%/0%/0%/0%
9B only	\$0.284	59.2	0%/100%/0%/0%/0%	\$0.019	53.3	0%/100%/0%/0%/0%	\$0.033	81.3	0%/100%/0%/0%/0%
35B only	\$0.679	66.2	0%/0%/100%/0%/0%	\$0.049	59.5	0%/0%/100%/0%/0%	\$0.079	84.2	0%/0%/100%/0%/0%
70B only	\$0.304	76.7	0%/0%/0%/100%/0%	\$0.024	59.9	0%/0%/0%/100%/0%	\$0.040	85.3	0%/0%/0%/100%/0%
122B only	\$1.217	73.3	0%/0%/0%/0%/100%	\$0.097	61.6	0%/0%/0%/0%/100%	\$0.157	84.9	0%/0%/0%/0%/100%
CARROT	\$0.606$_{\pm 0.010}$	70.0$_{\pm 1.0}$	9%/9%/11%/40%/31%	\$0.042$_{\pm 0.005}$	57.6$_{\pm 0.5}$	6%/17%/30%/30%/17%	**\$0.054**$_{\pm 0.001}$	82.01$_{\pm 1.0}$	12%/34%/9%/31%/14%
Causal	\$0.707$_{\pm 0.015}$	72.4$_{\pm 0.8}$	4%/7%/3%/42%/44%	\$0.044$_{\pm 0.004}$	56.9$_{\pm 0.6}$	11%/13%/28%/27%/21%	\$0.055$_{\pm 0.001}$	81.64$_{\pm 0.7}$	15%/32%/9%/28%/16%
MESS+	\$0.583$_{\pm 0.017}$	69.8$_{\pm 0.7}$	10%/7%/15%/41%/27%	\$0.038$_{\pm 0.003}$	57.6$_{\pm 2.0}$	7%/13%/33%/37%/10%	\$0.055$_{\pm 0.003}$	81.89$_{\pm 1.4}$	15%/26%/8%/35%/16%
SLARouter (ours)	**\$0.566**$_{\pm 0.021}$	70.3$_{\pm 0.8}$	9%/5%/17%/45%/24%	**\$0.036**$_{\pm 0.010}$	57.2$_{\pm 0.5}$	9%/11%/32%/39%/9%	\$0.057$_{\pm 0.001}$	82.55$_{\pm 0.6}$	12%/17%/22%/38%/11%

Method	BoolQ (α=85%)			GPQA (α=47%)			GSM8K (α=80%)		
	Operating Cost ($)	Request. Satisf. (%)	Model Call Ratio (2B/9B/35B/70B/122B)	Operating Cost ($)	Request. Satisf. (%)	Model Call Ratio (2B/9B/35B/70B/122B)	Operating Cost ($)	Request. Satisf. (%)	Model Call Ratio (2B/9B/35B/70B/122B)
2B only	\$0.020	72.0	100%/0%/0%/0%/0%	\$0.009	32.4	100%/0%/0%/0%/0%	\$0.043	56.7	100%/0%/0%/0%/0%
9B only	\$0.098	88.6	0%/100%/0%/0%/0%	\$0.043	40.0	0%/100%/0%/0%/0%	\$0.158	85.7	0%/100%/0%/0%/0%
35B only	\$0.142	89.0	0%/0%/100%/0%/0%	\$0.065	45.3	0%/0%/100%/0%/0%	\$0.391	87.6	0%/0%/100%/0%/0%
70B only	\$0.095	88.8	0%/0%/0%/100%/0%	\$0.043	49.1	0%/0%/0%/100%/0%	\$0.160	61.8	0%/0%/0%/100%/0%
122B only	\$0.265	87.2	0%/0%/0%/0%/100%	\$0.121	51.3	0%/0%/0%/0%/100%	\$0.816	83.7	0%/0%/0%/0%/100%
CARROT	**\$0.115**$_{\pm 0.002}$	85.7$_{\pm 0.5}$	12%/15%/13%/46%/14%	\$0.063$_{\pm 0.008}$	47.0$_{\pm 0.5}$	9%/10%/8%/46%/27%	\$0.315$_{\pm 0.008}$	80.2$_{\pm 1.1}$	6%/39%/22%/16%/17%
Causal	\$0.139$_{\pm 0.003}$	86.7$_{\pm 1.2}$	10%/7%/12%/44%/27%	\$0.061$_{\pm 0.005}$	47.1$_{\pm 1.0}$	7%/10%/11%/48%/24%	\$0.442$_{\pm 0.017}$	80.4$_{\pm 1.4}$	7%/25%/18%/12%/38%
MESS+	\$0.117$_{\pm 0.021}$	86.2$_{\pm 0.9}$	14%/13%/13%/45%/15%	\$0.067$_{\pm 0.003}$	47.9$_{\pm 0.7}$	6%/3%/15%/47%/29%	**\$0.301**$_{\pm 0.025}$	81.3$_{\pm 2.1}$	8%/47%/20%/9%/16%
SLARouter (ours)	\$0.116$_{\pm 0.025}$	86.0$_{\pm 0.9}$	15%/11%/9%/49%/16%	**\$0.060**$_{\pm 0.003}$	47.2$_{\pm 0.6}$	7%/8%/12%/51%/22%	\$0.303$_{\pm 0.016}$	81.3$_{\pm 0.8}$	6%/49%/17%/11%/17%

Method	LAMBADA (α=66%)			MMLU (α=80%)			SciQ (α=97%)		
	Operating Cost ($)	Request. Satisf. (%)	Model Call Ratio (2B/9B/35B/70B/122B)	Operating Cost ($)	Request. Satisf. (%)	Model Call Ratio (2B/9B/35B/70B/122B)	Operating Cost ($)	Request. Satisf. (%)	Model Call Ratio (2B/9B/35B/70B/122B)
2B only	\$0.009	48.1	100%/0%/0%/0%/0%	\$0.084	59.5	100%/0%/0%/0%/0%	\$0.010	94.3	100%/0%/0%/0%/0%
9B only	\$0.042	63.4	0%/100%/0%/0%/0%	\$0.410	75.5	0%/100%/0%/0%/0%	\$0.045	97.1	0%/100%/0%/0%/0%
35B only	\$0.064	69.3	0%/0%/100%/0%/0%	\$0.601	81.2	0%/0%/100%/0%/0%	\$0.069	96.8	0%/0%/100%/0%/0%
70B only	\$0.044	72.5	0%/0%/0%/100%/0%	\$0.401	82.0	0%/0%/0%/100%/0%	\$0.046	97.5	0%/0%/0%/100%/0%
122B only	\$0.121	69.4	0%/0%/0%/0%/100%	\$1.122	86.9	0%/0%/0%/0%/100%	\$0.131	97.4	0%/0%/0%/0%/100%
CARROT	**\$0.055**$_{\pm 0.005}$	66.1$_{\pm 0.7}$	17%/11%/22%/33%/17%	\$0.636$_{\pm 0.027}$	80.5$_{\pm 0.5}$	10%/11%/14%/32%/33%	**\$0.049**$_{\pm 0.008}$	97.0$_{\pm 0.3}$	12%/8%/8%/65%/7%
Causal	\$0.061$_{\pm 0.008}$	66.5$_{\pm 0.5}$	15%/9%/25%/29%/22%	\$0.662$_{\pm 0.018}$	80.8$_{\pm 1.2}$	9%/14%/11%/29%/37%	\$0.059$_{\pm 0.002}$	97.1$_{\pm 0.8}$	7%/12%/10%/55%/16%
MESS+	**\$0.055**$_{\pm 0.008}$	66.5$_{\pm 1.5}$	16%/10%/19%/38%/17%	\$0.600$_{\pm 0.017}$	80.3$_{\pm 1.5}$	11%/9%/12%/39%/29%	\$0.058$_{\pm 0.007}$	97.0$_{\pm 0.9}$	11%/13%/11%/49%/16%
SLARouter (ours)	**\$0.055**$_{\pm 0.003}$	67.0$_{\pm 0.8}$	16%/3%/26%/40%/15%	**\$0.556**$_{\pm 0.011}$	80.7$_{\pm 0.9}$	7%/9%/16%/48%/20%	\$0.051$_{\pm 0.004}$	97.1$_{\pm 0.2}$	9%/9%/11%/64%/7%

Method	SocialIQa (α=51%)			WinoGrande (α=78%)			*Avg. across 11 benchmarks* (α=72%)		
	Operating Cost ($)	Request. Satisf. (%)	Model Call Ratio (2B/9B/35B/70B/122B)	Operating Cost ($)	Request. Satisf. (%)	Model Call Ratio (2B/9B/35B/70B/122B)	Operating Cost ($)	Request. Satisf. (%)	Model Call Ratio (2B/9B/35B/70B/122B)
2B only	\$0.006	40.9	100%/0%/0%/0%/0%	\$0.002	63.1	100%/0%/0%/0%/0%	\$0.038	50.2	100%/0%/0%/0%/0%
9B only	\$0.020	48.1	0%/100%/0%/0%/0%	\$0.006	72.1	0%/100%/0%/0%/0%	\$0.176	63.1	0%/100%/0%/0%/0%
35B only	\$0.046	51.6	0%/0%/100%/0%/0%	\$0.020	74.7	0%/0%/100%/0%/0%	\$0.283	67.0	0%/0%/100%/0%/0%
70B only	\$0.024	54.4	0%/0%/0%/100%/0%	\$0.009	82.7	0%/0%/0%/100%/0%	\$0.175	67.9	0%/0%/0%/100%/0%
122B only	\$0.091	53.2	0%/0%/0%/0%/100%	\$0.040	76.2	0%/0%/0%/0%/100%	\$0.530	69.1	0%/0%/0%/0%/100%
CARROT	\$0.045$_{\pm 0.006}$	51.4$_{\pm 0.3}$	11%/9%/22%/30%/28%	\$0.020$_{\pm 0.003}$	77.4$_{\pm 0.3}$	7%/9%/8%/40%/36%	\$0.287$_{\pm 0.024}$	72.0$_{\pm 1.1}$	11%/14%/16%/35%/24%
Causal	\$0.062$_{\pm 0.006}$	51.0$_{\pm 0.8}$	17%/3%/7%/14%/59%	\$0.021$_{\pm 0.002}$	77.8$_{\pm 0.5}$	6%/9%/4%/43%/38%	\$0.313$_{\pm 0.019}$	72.3$_{\pm 0.9}$	11%/13%/13%/31%/32%
MESS+	\$0.042$_{\pm 0.002}$	50.2$_{\pm 0.3}$	17%/15%/26%/18%/24%	\$0.021$_{\pm 0.003}$	78.7$_{\pm 1.0}$	4%/4%/6%/50%/36%	\$0.274$_{\pm 0.017}$	72.0$_{\pm 1.7}$	13%/13%/15%/37%/22%
SLARouter (ours)	**\$0.041**$_{\pm 0.008}$	51.8$_{\pm 0.7}$	6%/16%/20%/37%/21%	**\$0.018**$_{\pm 0.003}$	78.4$_{\pm 0.8}$	8%/4%/6%/54%/28%	**\$0.258**$_{\pm 0.014}$	72.4$_{\pm 0.8}$	10%/11%/17%/44%/18%

fair comparison is not possible. However, we do conduct an ablation study on SLARouter to show its performance under lower feedback rates than 0.2.

Baselines. We compare SLARouter against CARROT (Somerstep et al., 2025), Causal LLM Routing (Tsiourvas et al., 2025), and MESS+ (Woisetschläger et al., 2025). In our comparison, we include the offline classifier training costs in the total operating cost. We align the V and c setting between MESS+ and SLARouter to ensure comparability.

5.2 Results

SLA Guarantees (α). No single model is able to achieve α across all benchmarks (Table 2). Hence, routing is required to solve the overall problem. *SLARouter is the only method that satisfies α without careful per-benchmark hyperparameter tuning*, by maintaining accurate constraint tracking through predictor-estimated queue updates under sparse feedback. MESS+ satisfies α on 9 out of 11 benchmarks, with failures attributable to feedback sparsity stalling Q updates under its assumption of complete, balanced signals. We have tuned CARROT's and Causal Router's cost sensitivity hyperparameter to satisfy α where possible. However, both require careful per-benchmark tuning of μ and λ. Further details are available in the appendix.

Cost Efficiency. *SLARouter achieves the best cost-quality trade-off on average.* It simultaneously satisfies α and minimizes operating cost. Among methods that meet α, SLARouter is $\sim 11\%$ and $\sim 18\%$ cheaper than CARROT and the Causal router, respectively. This is due to the interplay between the predictor and the adaptation of Q at runtime. MESS+ matches SLARouter on cost but undershoots α on several benchmarks, confirming that low operating cost alone is insufficient without reliable constraint tracking. The cost advantage is most pronounced on demanding workloads. On

Table 3: Introducing data-driven selection of V improves operational efficiency while still complying with our SLA α. It also removes a hyperparameter and makes SLARouter more practical to use.

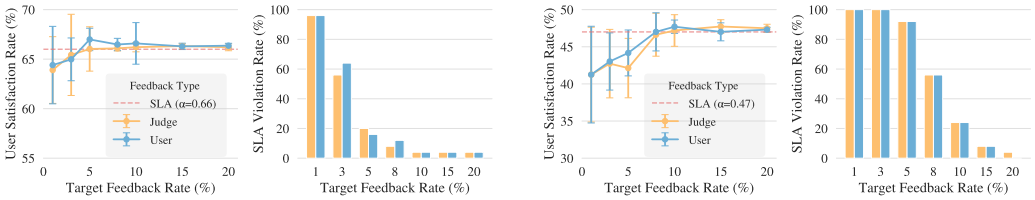
SLARouter variant	ACPBench (α=69%)			GPQA (α=47%)			WinoGrande (α=78%)		
	Operating Cost ($)	Post-Conv. Accuracy	Model Call Ratio (2B/9B/35B/70B/122B)	Operating Cost ($)	Post-Conv. Accuracy	Model Call Ratio (2B/9B/35B/70B/122B)	Operating Cost ($)	Post-Conv. Accuracy	Model Call Ratio (2B/9B/35B/70B/122B)
Best fixed V	$\$0.5655_{\pm 0.0208}$	$70.3_{\pm 0.8}$	9%/5%/17%/45%/24%	$\$0.0602_{\pm 0.0029}$	$47.2_{\pm 0.5}$	7%/8%/12%/51%/22%	$\$0.0177_{\pm 0.0029}$	$78.4_{\pm 0.8}$	8%/4%/6%/54%/28%
Data-driven	$\$0.4910_{\pm 0.0075}$	$69.3_{\pm 0.2}$	13%/4%/19%/48%/16%	$\$0.0557_{\pm 0.0012}$	$47.0_{\pm 0.3}$	11%/6%/5%/58%/20%	$\$0.0160_{\pm 0.0029}$	$78.1_{\pm 0.8}$	12%/4%/4%/56%/24%
Δ (rel.)	-13%	-1.4%	—	-7%	-0.4%	—	-10%	-0.4%	—

ACPBench, SLARouter reduces cost by $\sim 7\%$ relative to CARROT and by about 18% compared to the Causal router.

Satisfaction Accuracy and Margin. *SLARouter exhibits the lowest gap between average performance across 11 benchmarks and our SLA requirement α among adaptive methods*, confirming that its gains stem from more effective use of available signals rather than conservative routing. Unlike MESS+, whose queue underestimates satisfaction shortfalls under sparse feedback, SLARouter's virtual queue substitutes predictor estimates to maintain uninterrupted constraint tracking throughout deployment. CARROT and the Causal router are competitive, but their margins are sensitive to the benchmark-specific choice of their cost sensitivity parameter.

Data-Driven Adaptation of Cost-Sensitivity V. The results for SLARouter in Table 2 are all based on a fixed value for V to ensure comparability with MESS+. Yet, we discover that *a data-driven rule for setting V can lead to improved efficiencies* while remaining SLA compliant (Table 3). On challenging tasks like ACPBench, GPQA, and WinoGrande, the data-driven approach reduces operating cost by 13%, 7%, and 10%, respectively.

Predictor Quality under Sparse Feedback. *Synthetic feedback collected from an LLM-based judge is a well-suited supplement for missing user feedback.* When users do not provide feedback, SLARouter falls back to synthetic signals from GPT-5.4 with reasoning enabled, which closely mirrors real user feedback with 94.4% and 87.9% agreement on ACPBench and GPQA, respectively. We show that overall the predictor needs relatively few samples (< 180) to be able to estimate user satisfaction probabilities (Figure 1). The judge comes with no additional burden on end users. Details on the judge and alignment with ground truth data is in the appendix.



(a) **ACPBench** (1800 user requests) (b) **GPQA** (448 user requests)

Figure 1: Deployment of an LLM-based judge (GPT-5.4 with reasoning) to supplement predictor training under sparse user feedback. Judge feedback closely aligns with user feedback and helps reduce SLA violations, enabling reliable routing without the need for dense user engagement.

6 Conclusion

We present SLARouter, an online LLM router that extends the Lyapunov drift-plus-penalty framework, making it the first routing method to provide theoretical guarantees for cost optimality at a given user satisfaction rate under sparse, one-sided observational feedback. On average, SLARouter reduces operating costs by $\sim 20\%$ over existing baselines while satisfying SLA constraints without per-benchmark tuning. Those results are produced over a model zoo with multiple model families and non-linear price/parameter relationships (Llama 3.3 70B costs the about same as Qwen 3.5 9B), showing that SLARouter adapts to heterogeneous zoos regardless of their shape. When user feedback is sparse, an LLM-based judge provides a reliable synthetic supplement, enabling robust predictor updates without additional burden on end users. Our approach assumes constant feedback rates and IID request arrivals, which may not hold in production systems where feedback can arrive in bursts and where requests by individual users may be correlated. Hence, we see it as a priority to address non-constant, bursty feedback arriving in temporal waves and the challenge of learning routing policies from negativity-biased feedback in future work.

AI USE STATEMENT

In this work, we used generative AI tools for implementing methods and assistance in writing the proofs. We have not used generative AI tools to help develop theoretical models or conceptual frameworks, formulate mathematical claims, provide critical ingredients for proving mathematical claims, propose or refine hypotheses, design or provide feedback on research methodology or experiments. The following tasks are not applicable to our work: assist with translation, clean and reformat datasets, support qualitative and thematic data analysis, interpret results. In addition to the required disclosure, we acknowledge the use of generative AI for the following: create or modify scientific figures or images, create or edit software code, brainstorming, edit a research paper to improve readability. All outputs generated by AI have been verified by the authors and manually checked against their source. We take responsibility for the final content of this work, including text, claims, or artifacts produced with the aid of generative AI.

REPRODUCIBILITY STATEMENT

The code to fully reproduce the results reported in our paper is available here: `https://anonymous.4open.science/r/slarouter-F044`. This includes the pipeline to generate all relevant benchmark data, run the paper experiments (incl. related work baselines), and produce tables/figures. We provide detailed instructions on reproducing the results in the appendix. The code repository supports running additional experiments and is built in a way for people to easily extend the evaluation harness with their own routing techniques and use their own performance criteria. This is, we emphasize usability and extensibility with our code base.

REFERENCES

P. Aggarwal, A. Madaan, A. Anand, S. P. Potharaju, S. Mishra, P. Zhou, A. Gupta, D. Rajagopal, K. Kappaganthu, Y. Yang, S. Upadhyay, M. Faruqui, and Mausam. AutoMix: Automatically mixing language models. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2024.

A. S. Bhatti, V. Vaddina, and D. Birru. PROTEUS: SLA-aware routing via Lagrangian RL for multi-LLM serving systems. In *Proceedings of the 6th European Workshop on Machine Learning and Systems (EuroMLSys '26)*, pages 60–71, New York, NY, USA, 2026. Association for Computing Machinery. doi: 10.1145/3805621.3807607.

L. Chen, M. Zaharia, and J. Zou. FrugalGPT: How to use large language models while reducing cost and improving performance. *Transactions on Machine Learning Research (TMLR)*, 2024a.

S. Chen, W. Jiang, B. Lin, J. T. Kwok, and Y. Zhang. RouterDC: Query-based router by dual contrastive learning for assembling large language models. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2024b.

C. Clark, K. Lee, M.-W. Chang, T. Kwiatkowski, M. Collins, and K. Toutanova. BoolQ: Exploring the surprising difficulty of natural yes/no questions. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 2924–2936, 2019.

P. Clark, I. Cowhey, O. Etzioni, T. Khot, A. Sabharwal, C. Schoenick, and O. Tafjord. Think you have solved question answering? Try ARC, the AI2 reasoning challenge. *ArXiv*, abs/1803.05457, 2018.

K. Cobbe, V. Kosaraju, M. Bavarian, M. Chen, H. Jun, L. Kaiser, M. Plappert, J. Tworek, J. Hilton, R. Nakano, C. Hesse, and J. Schulman. Training verifiers to solve math word problems. *ArXiv*, abs/2110.14168, 2021.

D. Ding, A. Mallick, C. Wang, R. Sim, S. Mukherjee, V. Rühle, L. V. S. Lakshmanan, and A. H. Awadallah. Hybrid LLM: Cost-efficient and quality-aware query routing. In *International Conference on Learning Representations (ICLR)*, 2024. URL `https://openreview.net/forum?id=02f3mUtqnM`.

A. Dubey et al. The Llama 3 herd of models, 2024.

European Parliament and Council of the European Union. Regulation (EU) 2024/1689 of the European Parliament and of the Council of 13 June 2024 laying down harmonised rules on artificial intelligence (Artificial Intelligence Act). Technical report, 2024. URL http://data.europa.eu/eli/reg/2024/1689/oj.

L. Gao, J. Tow, S. Biderman, et al. A framework for few-shot language model evaluation, Dec. 2023. URL https://github.com/EleutherAI/lm-evaluation-harness. Version v0.4.0.

GLM-5 Team, A. Zeng, et al. GLM-5: from vibe coding to agentic engineering, 2026.

D. Hendrycks, C. Burns, S. Basart, A. Zou, M. Mazeika, D. Song, and J. Steinhardt. Measuring massive multitask language understanding. *Proceedings of the International Conference on Learning Representations (ICLR)*, 2021.

C. Huang, T. Shi, Y. Zhu, R. Chen, and X. Quan. Lookahead routing for large language models. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2025.

IBM Research. Granite 4.0 language models. https://github.com/ibm-granite/granite-4.0-language-models, 2025.

H. Kokel, M. Katz, K. Srinivas, and S. Sohrabi. Acpbench: Reasoning about action, change, and planning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 39, pages 26559–26568, 2025.

K. Krauth, Y. Wang, and M. Jordan. Breaking feedback loops in recommender systems with causal inference. *ACM Trans. Recomm. Syst.*, 4(1), July 2025.

Meta AI. Llama-3.3-70B-Instruct model card. Hugging Face, Dec. 2024. URL https://huggingface.co/meta-llama/Llama-3.3-70B-Instruct.

MiniMax, A. Li, B. Gong, et al. MiniMax-01: Scaling foundation models with lightning attention, 2025.

H. Naveed, A. U. Khan, S. Qiu, M. Saqib, S. Anwar, M. Usman, N. Akhtar, N. Barnes, and A. Mian. A comprehensive overview of large language models. *ACM Transactions on Intelligent Systems and Technology*, 16(5):1–72, Aug. 2025. ISSN 2157-6912.

M. Neely. *Stochastic network optimization with application to communication and queueing systems*. Springer Nature, 2022.

I. Ong, A. Almahairi, V. Wu, W.-L. Chiang, T. Wu, J. E. Gonzalez, M. W. Kadous, and I. Stoica. RouteLLM: Learning to route LLMs from preference data. In *International Conference on Learning Representations (ICLR)*, 2025. URL https://openreview.net/forum?id=8sSqNntaMr.

OpenRouter. OpenRouter: Model pricing. https://openrouter.ai/models, 2026. Accessed August 2026.

D. Paperno, G. Kruszewski, A. Lazaridou, N. Q. Pham, R. Bernardi, S. Pezzelle, M. Baroni, G. Boleda, and R. Fernández. The LAMBADA dataset: Word prediction requiring a broad discourse context. In K. Erk and N. A. Smith, editors, *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1525–1534, Berlin, Germany, Aug. 2016. Association for Computational Linguistics.

C. Qian, Z. Liu, S. Kokane, A. Prabhakar, J. Qiu, H. Chen, Z. Liu, H. Ji, W. Yao, S. Heinecke, S. Savarese, C. Xiong, and H. Wang. xRouter: Training cost-aware LLMs orchestration system via reinforcement learning. *arXiv preprint arXiv:2510.08439*, 2025.

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

D. Rein, B. L. Hou, A. C. Stickland, J. Petty, R. Y. Pang, J. Dirani, J. Michael, and S. R. Bowman. GPQA: A graduate-level google-proof q&a benchmark. In *First Conference on Language Modeling*, 2024.

K. Sakaguchi, R. Le Bras, C. Bhagavatula, and Y. Choi. WinoGrande: An adversarial winograd schema challenge at scale. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 8732–8740, 2020.

S. Samsi, D. Zhao, J. McDonald, B. Li, A. Michaleas, M. Jones, W. Bergeron, J. Kepner, D. Tiwari, and V. Gadepally. From words to watts: Benchmarking the energy costs of large language model inference. In *2023 IEEE High Performance Extreme Computing Conference (HPEC)*, pages 1–9, 2023.

M. Sap, H. Rashkin, D. Chen, R. Le Bras, and Y. Choi. Social IQa: Commonsense reasoning about social interactions. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 4463–4473, 2019.

S. Somerstep, F. Maia Polo, A. F. Melo de Oliveira, P. Mangal, M. Silva, O. Bhardwaj, M. Yurochkin, and S. Maity. CARROT: A cost aware rate optimal router. In *ICLR 2025 Workshop on Foundation Models in the Wild*, 2025. URL https://openreview.net/forum?id=xEBOy2ze1U.

D. Stripelis, Z. Xu, Z. Hu, A. D. Shah, H. Jin, Y. Yao, J. Zhang, T. Zhang, S. Avestimehr, and C. He. TensorOpera router: A multi-model router for efficient LLM inference. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing: Industry Track*, pages 452–462, Miami, Florida, US, 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.emnlp-industry.34. URL https://aclanthology.org/2024.emnlp-industry.34/.

A. Tsiourvas, W. Sun, and G. Perakis. Causal LLM routing: End-to-end regret minimization from observational data. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2025.

B. Warner, A. Chaffin, et al. Smarter, better, faster, longer: A modern bidirectional encoder for fast, memory efficient, and long context finetuning and inference. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2526–2547. Association for Computational Linguistics, 2025. doi: 10.18653/v1/2025.acl-long.127. URL https://aclanthology.org/2025.acl-long.127/.

J. Welbl, N. F. Liu, and M. Gardner. Crowdsourcing multiple choice science questions. In *Proceedings of the 3rd Workshop on Noisy User-generated Text*, pages 94–106, 2017.

H. Woisetschläger, R. Zhang, S. Wang, and H.-A. Jacobsen. MESS+: Dynamically learned inference-time LLM routing in model zoos with service level guarantees. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2025.

A. Yang et al. Qwen3 technical report, 2025.

J. You, J.-W. Chung, and M. Chowdhury. Zeus: Understanding and optimizing GPU energy consumption of DNN training. In *20th USENIX Symposium on Networked Systems Design and Implementation (NSDI 23)*. USENIX Association, 2023. URL https://www.usenix.org/conference/nsdi23/presentation/you.

W. Zhao, A. Patterson, C. C. Senthilkumar, Y. Wang, A. Gupta, S. Sadighi, and N. Nayyar. From metrics to meaning: Estimating user feedback using LLM-based evaluation. 2026. URL https://www.amazon.science/publications/from-metrics-to-meaning-estimating-user-feedback-using-llm-based-evaluation.

L. Zheng, W.-L. Chiang, Y. Sheng, S. Zhuang, Z. Wu, Y. Zhuang, Z. Lin, Z. Li, D. Li, E. P. Xing, H. Zhang, J. E. Gonzalez, and I. Stoica. Judging LLM-as-a-Judge with MT-Bench and chatbot arena. In *Advances in Neural Information Processing Systems (NeurIPS), Datasets and Benchmarks Track*, 2023.

Appendix

A Proofs **14**

A.1 Proof of Lemma 1 14

A.2 Total Expectation Result of Lemma 1 18

A.3 Proof of Lemma 2 19

A.4 Proof of Theorem 1 20

A.5 Proof of Corollary 1 23

A.6 Proof of Theorem 2 23

B Experimental Details & Additional Results **25**

B.1 Environment and Hardware 25

B.2 Dataset 25

B.3 Predictor: Architecture and Training Details 26

B.4 Performance Matching Between Baselines and SLARouter 27

B.5 Non-stationary benchmark scenarios 28

B.6 Judge Feedback Alignment 28

A PROOFS

A.1 PROOF OF LEMMA 1

For the loss gradient on a single sample, the variance term (stochastic noise) can be defined as

$$\mathbb{E}\left[\left\|\nabla f(\mathbf{z},\mathbf{a},m)-\nabla F(\mathbf{z})\right\|^2\middle|\mathbf{z}\right]\leq\sigma^2,\ \forall\mathbf{z},m,$$

where the inequality is due to Assumption 1. Subsequently, we can define the variance term for the stochastic gradient (updated on mini-batch size B, where we use the short-hand notation $\mathbf{g}:=\nabla F(\mathbf{z},\mathcal{B})$) and the empirical gradient $\nabla\hat{F}$ computed on $|\mathcal{D}|$ available samples as

$$\mathbb{E}\left[\left\|\mathbf{g}-\nabla F(\mathbf{z})\right\|^2\middle|\mathbf{z}\right]\leq\frac{\sigma^2}{B},\ \forall\mathbf{z},m,\tag{A.1}$$

$$\mathbb{E}\left[\left\|\nabla\hat{F}(\mathbf{z})-\nabla F(\mathbf{z})\right\|^2\middle|\mathbf{z}\right]\leq\frac{\sigma^2}{|\mathcal{D}|},\ \forall\mathbf{z},m,\tag{A.2}$$

respectively.

We then have

$$\begin{aligned}
&\mathbb{E}\left[\left\|\mathbf{g}-\nabla\hat{F}(\mathbf{z})\right\|^2\middle|\mathbf{z}\right]\\
&=\mathbb{E}\left[\left\|\mathbf{g}-\nabla\hat{F}(\mathbf{z})+\nabla F(\mathbf{z})-\nabla F(\mathbf{z})\right\|^2\middle|\mathbf{z}\right]\\
&\leq\mathbb{E}\left[2\left\|\mathbf{g}-\nabla F(\mathbf{z})\right\|^2+2\left\|\nabla\hat{F}(\mathbf{z})-\nabla F(\mathbf{z})\right\|^2\middle|\mathbf{z}\right]\\
&\leq\frac{2\sigma^2}{B}+\frac{2\sigma^2}{|\mathcal{D}|},\ \forall\mathbf{z},m
\end{aligned}\tag{A.3}$$

We note that $\mathbb{E}\left[\mathbf{g}_k|\,\mathbf{z}_k\right]:=\nabla\hat{F}(\mathbf{z}_k)$, which is generally not equal to $\nabla F(\mathbf{z}_k)$ under partial feedback.

Proof. Let k denote the index of exploration steps. The SGD update of predictor training is

$$\mathbf{z}_{k+1}\leftarrow\mathbf{z}_k-\eta_k\mathbf{g}_\mathbf{k}\tag{A.4}$$

By L-smoothness, we have

$$\begin{aligned}
&\mathbb{E}\left[F(\mathbf{z}_{k+1})|\,\mathbf{z}_k\right]\\
&\leq F(\mathbf{z}_k)+\left\langle\nabla F(\mathbf{z}_k),\mathbb{E}\left[\mathbf{z}_{k+1}-\mathbf{z}_k|\,\mathbf{z}_k\right]\right\rangle+\frac{L}{2}\left\|\mathbf{z}_{k+1}-\mathbf{z}_k\right\|^2\\
&=F(\mathbf{z}_k)-\eta_k\left\langle\nabla F(\mathbf{z}_k),\mathbb{E}\left[\mathbf{g}_k|\,\mathbf{z}_k\right]\right\rangle+\frac{L\eta_k^2}{2}\mathbb{E}\left[\left\|\mathbf{g}_k\right\|^2\middle|\,\mathbf{z}_k\right]\\
&\leq F(\mathbf{z}_k)-\eta_k\left\langle\nabla F(\mathbf{z}_k),\nabla\hat{F}(\mathbf{z}_k)-\nabla F(\mathbf{z}_k)+\nabla F(\mathbf{z}_k)\right\rangle+\frac{L\eta_k^2}{2}\mathbb{E}\left[\left\|\mathbf{g}_k\right\|^2\middle|\,\mathbf{z}_k\right]\\
&\leq F(\mathbf{z}_k)-\eta_k\left\langle\nabla F(\mathbf{z}_k),\nabla\hat{F}(\mathbf{z}_k)-\nabla F(\mathbf{z}_k)\right\rangle-\eta_k\left\|\nabla F(\mathbf{z}_k)\right\|^2+\frac{L\eta_k^2}{2}\mathbb{E}\left[\left\|\mathbf{g}_k\right\|^2\middle|\,\mathbf{z}_k\right]\\
&\leq F(\mathbf{z}_k)+\frac{\eta_k\left\|\nabla F(\mathbf{z}_k)\right\|^2}{4}+\eta_k\left\|\nabla\hat{F}(\mathbf{z}_k)-\nabla F(\mathbf{z}_k)\right\|^2-\eta_k\left\|\nabla F(\mathbf{z}_k)\right\|^2+\frac{L\eta_k^2}{2}\mathbb{E}\left[\left\|\mathbf{g}_k\right\|^2\middle|\,\mathbf{z}_k\right]\\
&\leq F(\mathbf{z}_k)-\frac{3}{4}\eta_k\left\|\nabla F(\mathbf{z}_k)\right\|^2+\eta_k\left\|\nabla\hat{F}(\mathbf{z}_k)-\nabla F(\mathbf{z}_k)\right\|^2+\frac{L\eta_k^2}{2}\left(\left\|\nabla\hat{F}(\mathbf{z}_k)\right\|^2+\frac{2\sigma^2}{B}+\frac{2\sigma^2}{|\mathcal{D}|}\right)\\
&\leq F(\mathbf{z}_k)-\frac{3}{4}\eta_k\left\|\nabla F(\mathbf{z}_k)\right\|^2+\eta_k\left\|\nabla\hat{F}(\mathbf{z}_k)-\nabla F(\mathbf{z}_k)\right\|^2\\
&\quad+\frac{L\eta_k^2}{2}\left(\left\|\nabla\hat{F}(\mathbf{z}_k)-\nabla F(\mathbf{z}_k)+\nabla F(\mathbf{z}_k)\right\|^2+\frac{2\sigma^2}{B}+\frac{2\sigma^2}{|\mathcal{D}|}\right)
\end{aligned}$$

$$
\begin{aligned}
&\leq F(\mathbf{z}_k) - \frac{3}{4}\eta_k \left\|\nabla F(\mathbf{z}_k)\right\|^2 + \eta_k \left\|\nabla \hat{F}(\mathbf{z}_k) - \nabla F(\mathbf{z}_k)\right\|^2 \\
&\quad + L\eta_k^2 \left(\left\|\nabla \hat{F}(\mathbf{z}_k) - \nabla F(\mathbf{z}_k)\right\|^2 + \left\|\nabla F(\mathbf{z}_k)\right\|^2 + \frac{\sigma^2}{B} + \frac{\sigma^2}{|\mathcal{D}|} \right) \\
&= F(\mathbf{z}_k) - \left(\frac{3}{4}\eta_k - L\eta_k^2\right) \left\|\nabla F(\mathbf{z}_k)\right\|^2 + (\eta_k + L\eta_k^2) \left\|\nabla \hat{F}(\mathbf{z}_k) - \nabla F(\mathbf{z}_k)\right\|^2 + L\eta_k^2 \left(\frac{\sigma^2}{B} + \frac{\sigma^2}{|\mathcal{D}|}\right) \\
&= F(\mathbf{z}_k) - \left(\frac{3}{4}\eta_k - L\eta_k^2\right) \left\|\nabla F(\mathbf{z}_k)\right\|^2 + (\eta_k + 2L\eta_k^2)\frac{\sigma^2}{|\mathcal{D}|} + L\eta_k^2 \frac{\sigma^2}{B}. \qquad \text{(A.5)}
\end{aligned}
$$

From the PL condition, we have $\left\|\nabla F(\mathbf{z})\right\|^2 \geq 2\mu \left(F(\mathbf{z}) - F_{\min}\right), \forall \mathbf{z}$. Subtracting $F_{\min}$ on both sides of (A.5) and plugging in the PL inequality gives

$$
\begin{aligned}
&\mathbb{E}\left[F(\mathbf{z}_{k+1}) - F_{\min} \middle| \mathbf{z}_k\right] \\
&\leq (F(\mathbf{z}_k) - F_{\min}) - 2\mu \left(\frac{3}{4}\eta_k - L\eta_k^2\right)(F(\mathbf{z}_k) - F_{\min}) + (\eta_k + 2L\eta_k^2)\frac{\sigma^2}{|\mathcal{D}|} + L\eta_k^2\frac{\sigma^2}{B} \\
&= \left(1 - 2\mu\eta_k \left(\frac{3}{4} - L\eta_k\right)\right)(F(\mathbf{z}_k) - F_{\min}) + (\eta_k + 2L\eta_k^2)\frac{\sigma^2}{|\mathcal{D}|} + L\eta_k^2\frac{\sigma^2}{B}. \qquad \text{(A.6)}
\end{aligned}
$$

Let $\eta_k \leq \frac{1}{2L}$. We have $\frac{3}{4} - L\eta_k \geq \frac{1}{4}$, and thus

$$
\mathbb{E}\left[F(\mathbf{z}_{k+1}) - F_{\min} \middle| \mathbf{z}_k\right] \leq \left(1 - \frac{\mu\eta_k}{2}\right)(F(\mathbf{z}_k) - F_{\min}) + \frac{2\eta_k\sigma^2}{|\mathcal{D}|} + \frac{L\eta_k^2\sigma^2}{B}, \qquad \text{(A.7)}
$$

where we have absorbed the $2L\eta_k^2$ term in the $|\mathcal{D}|$-variance into the leading η_k term since $\eta_k \leq \frac{1}{2L}$ implies $2L\eta_k^2 \leq \eta_k$. The remaining mini-batch variance term $\frac{L\eta_k^2\sigma^2}{B}$ is kept unchanged.

Taking total expectation gives

$$
\mathbb{E}\left[F(\mathbf{z}_{k+1})\right] - F_{\min} \leq \left(1 - \frac{\mu\eta_k}{2}\right)\left(\mathbb{E}\left[F(\mathbf{z}_k)\right] - F_{\min}\right) + \frac{2\eta_k\sigma^2}{|\mathcal{D}|} + \frac{L\eta_k^2\sigma^2}{B}. \qquad \text{(A.8)}
$$

Let $\mathcal{F}_k := \mathbb{E}\left[F(\mathbf{z}_k)\right] - F_{\min}$. We have

$$
\mathcal{F}_2 \leq \left(1 - \frac{\mu\eta_1}{2}\right)\mathcal{F}_1 + \frac{2\eta_1\sigma^2}{|\mathcal{D}|} + \frac{L\eta_1^2\sigma^2}{B}.
$$

Next, we show by induction that

$$
\mathcal{F}_k \leq \mathcal{F}_1 \prod_{\kappa=1}^{k-1}\left(1 - \frac{\mu\eta_\kappa}{2}\right) + \frac{2\sigma^2}{|\mathcal{D}|}\sum_{\kappa=1}^{k-1}\eta_\kappa \prod_{\kappa'=\kappa+1}^{k-1}\left(1 - \frac{\mu\eta_{\kappa'}}{2}\right) + \frac{L\sigma^2}{B}\sum_{\kappa=1}^{k-1}\eta_\kappa^2 \prod_{\kappa'=\kappa+1}^{k-1}\left(1 - \frac{\mu\eta_{\kappa'}}{2}\right). \qquad \text{(A.9)}
$$

To see this, we assume that (A.9) holds for $k \geq 2$, as we have shown above that it holds for $k = 2$. We then have according to (A.8) that

$$
\begin{aligned}
&\mathcal{F}_{k+1} \\
&\leq \left(1 - \frac{\mu\eta_k}{2}\right)\mathcal{F}_k + \frac{2\eta_k\sigma^2}{|\mathcal{D}|} + \frac{L\eta_k^2\sigma^2}{B} \\
&\leq \left(1 - \frac{\mu\eta_k}{2}\right)\left[\mathcal{F}_1 \prod_{\kappa=1}^{k-1}\left(1 - \frac{\mu\eta_\kappa}{2}\right) + \frac{2\sigma^2}{|\mathcal{D}|}\sum_{\kappa=1}^{k-1}\eta_\kappa \prod_{\kappa'=\kappa+1}^{k-1}\left(1 - \frac{\mu\eta_{\kappa'}}{2}\right) + \frac{L\sigma^2}{B}\sum_{\kappa=1}^{k-1}\eta_\kappa^2 \prod_{\kappa'=\kappa+1}^{k-1}\left(1 - \frac{\mu\eta_{\kappa'}}{2}\right)\right] \\
&\quad + \frac{2\eta_k\sigma^2}{|\mathcal{D}|} + \frac{L\eta_k^2\sigma^2}{B} \\
&= \mathcal{F}_1 \prod_{\kappa=1}^{k}\left(1 - \frac{\mu\eta_\kappa}{2}\right) + \frac{2\sigma^2}{|\mathcal{D}|}\sum_{\kappa=1}^{k-1}\eta_\kappa \prod_{\kappa'=\kappa+1}^{k}\left(1 - \frac{\mu\eta_{\kappa'}}{2}\right) + \frac{L\sigma^2}{B}\sum_{\kappa=1}^{k-1}\eta_\kappa^2 \prod_{\kappa'=\kappa+1}^{k}\left(1 - \frac{\mu\eta_{\kappa'}}{2}\right)
\end{aligned}
$$

$$+\frac{2\eta_k\sigma^2}{|\mathcal{D}|}+\frac{L\eta_k^2\sigma^2}{B}$$

$$=\mathcal{F}_1\prod_{\kappa=1}^{k}\left(1-\frac{\mu\eta_\kappa}{2}\right)+\frac{2\sigma^2}{|\mathcal{D}|}\sum_{\kappa=1}^{k}\eta_\kappa\prod_{\kappa'=\kappa+1}^{k}\left(1-\frac{\mu\eta_{\kappa'}}{2}\right)+\frac{L\sigma^2}{B}\sum_{\kappa=1}^{k}\eta_\kappa^2\prod_{\kappa'=\kappa+1}^{k}\left(1-\frac{\mu\eta_{\kappa'}}{2}\right),$$

where the last equality is because $\prod_{\kappa'=k+1}^{k}\left(1-\frac{\mu\eta_{\kappa'}}{2}\right)=1$. This proves (A.9).

Recall that $\eta_k=\min\left\{\frac{4}{\mu(k+1)},\frac{1}{2L}\right\}$. We note that, due to L-smoothness, we have $\mu\le L$, because otherwise the L-smoothness contradicts the PL condition. Therefore, $\frac{8L}{\mu}\ge 8$. We have $\eta_k=\frac{1}{2L}$ when $k\le\tilde{k}:=\left\lfloor\frac{8L}{\mu}-1\right\rfloor$. For $k>\tilde{k}$, we have $\eta_k=\frac{4}{\mu(k+1)}$, and in this case, $1-\frac{\mu\eta_k}{2}=1-\frac{2}{k+1}=\frac{k-1}{k+1}$.

We now bound the last two terms of (A.9). Define

$$G_1(k):=\sum_{\kappa=1}^{k-1}\eta_\kappa\prod_{\kappa'=\kappa+1}^{k-1}\left(1-\frac{\mu\eta_{\kappa'}}{2}\right),\tag{A.10}$$

$$G_2(k):=\sum_{\kappa=1}^{k-1}\eta_\kappa^2\prod_{\kappa'=\kappa+1}^{k-1}\left(1-\frac{\mu\eta_{\kappa'}}{2}\right),\tag{A.11}$$

so that the last two terms of (A.9) are equal to $\frac{2\sigma^2}{|\mathcal{D}|}G_1(k)+\frac{L\sigma^2}{B}G_2(k)$.

For ease of presentation, we also define $k_0:=\tilde{k}+1$. Then, for $k\le k_0$, we have $\kappa\le\tilde{k}$ and $\kappa'\le\tilde{k}$ in the sum and product terms, respectively.

Bounding $G_1(k)$. For $k\le k_0$, since $\eta_\kappa=\frac{1}{2L}$ and $1-\frac{\mu\eta_{\kappa'}}{2}=1-\frac{\mu}{4L}$ throughout,

$$G_1(k)=\sum_{\kappa=1}^{k-1}\frac{1}{2L}\left(1-\frac{\mu}{4L}\right)^{k-1-\kappa}\le\frac{1}{2L}\cdot\frac{1}{\frac{\mu}{4L}}=\frac{2}{\mu}.\tag{A.12}$$

For $k\ge k_0$, the recurrence form of (A.10) is

$$\begin{aligned}G_1(k+1)&=\left(1-\frac{\mu\eta_k}{2}\right)G_1(k)+\eta_k\\&=\frac{k-1}{k+1}\cdot G_1(k)+\frac{4}{\mu(k+1)}.\end{aligned}\tag{A.13}$$

We now show by induction that, for general $k>k_0$, we have

$$G_1(k)\le\frac{2\tilde{k}k_0}{\mu(k-1)k}+\frac{4(k-k_0)}{\mu k}\le\mathcal{O}(1),\tag{A.14}$$

where the last inequality is straightforward because $\tilde{k}$ and k_0 are constants depending only on L and μ, and the last term of $\frac{4(k-k_0)}{\mu k}$ does not vanish but is upper bounded by $\frac{4}{\mu}$ for arbitrarily large k. We therefore focus on proving the first inequality as follows.

To see this, we first note that (A.14) holds for $k=k_0$, because

$$\frac{2\tilde{k}k_0}{\mu(k_0-1)k_0}+\frac{4(k_0-k_0)}{\mu k_0}=\frac{2\tilde{k}k_0}{\mu\tilde{k}k_0}=\frac{2}{\mu}\ge G_1(k_0)$$

according to (A.12). Now, assume that (A.14) holds for some $k\ge k_0$. We then have from (A.13) that

$$\begin{aligned}G_1(k+1)&=\frac{k-1}{k+1}\cdot G_1(k)+\frac{4}{\mu(k+1)}\\&\le\frac{k-1}{k+1}\cdot\left[\frac{2\tilde{k}k_0}{\mu(k-1)k}+\frac{4(k-k_0)}{\mu k}\right]+\frac{4}{\mu(k+1)}\end{aligned}$$

$$
\begin{aligned}
&= \frac{2\tilde{k}k_0}{\mu k(k+1)} + \frac{4(k-k_0)(k-1)}{\mu k(k+1)} + \frac{4}{\mu(k+1)} \\
&\leq \frac{2\tilde{k}k_0}{\mu k(k+1)} + \frac{4(k-k_0)}{\mu(k+1)} + \frac{4}{\mu(k+1)} \\
&= \frac{2\tilde{k}k_0}{\mu k(k+1)} + \frac{4(k+1-k_0)}{\mu(k+1)},
\end{aligned}
$$

which proves (A.14).

Bounding $G_2(k)$. For $k \leq k_0$, since $\eta_\kappa = \frac{1}{2L}$ and $1 - \frac{\mu\eta_{\kappa'}}{2} = 1 - \frac{\mu}{4L}$ throughout,

$$
G_2(k) = \sum_{\kappa=1}^{k-1} \frac{1}{4L^2} \left(1 - \frac{\mu}{4L}\right)^{k-1-\kappa} \leq \frac{1}{4L^2} \cdot \frac{1}{\frac{\mu}{4L}} = \frac{1}{L\mu}. \tag{A.15}
$$

For $k \geq k_0$, the recurrence form of (A.11) is

$$
\begin{aligned}
G_2(k+1) &= \left(1 - \frac{\mu\eta_k}{2}\right) G_2(k) + \eta_k^2 \\
&= \frac{k-1}{k+1} \cdot G_2(k) + \frac{16}{\mu^2(k+1)^2}.
\end{aligned} \tag{A.16}
$$

We now show by induction that, for general $k > k_0$, we have

$$
G_2(k) \leq \frac{\tilde{k}k_0}{L\mu(k-1)k} + \frac{16}{\mu^2 k} = \mathcal{O}\left(\frac{1}{k}\right), \tag{A.17}
$$

where the last equality is straightforward because $\tilde{k}$ and k_0 are constants depending only on L and μ. We therefore focus on proving the first inequality as follows.

To see this, we first note that (A.17) holds for $k = k_0$, because

$$
\frac{\tilde{k}k_0}{L\mu(k_0-1)k_0} + \frac{16}{\mu^2 k} = \frac{\tilde{k}k_0}{L\mu\tilde{k}k_0} + \frac{16}{\mu^2 k} = \frac{1}{L\mu} + \frac{16}{\mu^2 k} \geq \frac{1}{L\mu} \geq G_2(k_0),
$$

where the last inequality follows from (A.15). Now, assume that (A.17) holds for some $k \geq k_0$. We then have from (A.16) that

$$
\begin{aligned}
G_2(k+1) &= \frac{k-1}{k+1} \cdot G_2(k) + \frac{16}{\mu^2(k+1)^2} \\
&\leq \frac{k-1}{k+1} \cdot \left[\frac{\tilde{k}k_0}{L\mu(k-1)k} + \frac{16}{\mu^2 k}\right] + \frac{16}{\mu^2(k+1)^2} \\
&= \frac{\tilde{k}k_0}{L\mu k(k+1)} + \frac{16(k-1)}{\mu^2 k(k+1)} + \frac{16}{\mu^2(k+1)^2} \\
&\leq \frac{\tilde{k}k_0}{L\mu k(k+1)} + \frac{16(k-1)}{\mu^2 k(k+1)} + \frac{16}{\mu^2 k(k+1)} \\
&\leq \frac{\tilde{k}k_0}{L\mu k(k+1)} + \frac{16}{\mu^2(k+1)},
\end{aligned}
$$

which proves (A.17).

Total Noise Term. Combining (A.14) and (A.17) with (A.9), the total noise contribution satisfies

$$
\frac{2\sigma^2}{|\mathcal{D}|} G_1(k) + \frac{\sigma^2}{B} G_2(k) = \mathcal{O}\left(\frac{\sigma^2}{|\mathcal{D}|} + \frac{\sigma^2}{kB}\right). \tag{A.18}
$$

Bounding First Term of (A.9)**.** Finally, we consider the first term of (A.9). When $k \leq k_0$,

$$
H(k) := \prod_{\kappa=1}^{k-1} \left(1 - \frac{\mu\eta_\kappa}{2}\right) = \left(1 - \frac{\mu}{4L}\right)^{k-1}. \tag{A.19}
$$

For $k \geq k_0$, recall that $\eta_k = \frac{4}{\mu(k+1)}$ and thus $1 - \frac{\mu\eta_k}{2} = 1 - \frac{2}{k+1} = \frac{k-1}{k+1}$. We have the following recurrence relation:

$$H(k+1) = \left(1 - \frac{\mu\eta_k}{2}\right) H(k) = \frac{k-1}{k+1} \cdot H(k). \tag{A.20}$$

We now prove by induction that, for general $k > k_0$,

$$H(k) = \left(1 - \frac{\mu}{4L}\right)^{\tilde{k}} \cdot \frac{\tilde{k}k_0}{(k-1)k} = \mathcal{O}\left(\frac{1}{k^2}\right), \tag{A.21}$$

where the last equality is because $\tilde{k}$ and k_0 are constants as they only depend on L and μ. Thus, we only focus on proving the first equality.

We note that (A.21) holds for $k = k_0$, because

$$\left(1 - \frac{\mu}{4L}\right)^{\tilde{k}} \cdot \frac{\tilde{k}k_0}{\tilde{k}k_0} = \left(1 - \frac{\mu}{4L}\right)^{k_0 - 1} = H(k_0),$$

where the last equality is from (A.19). Now, assume that (A.21) holds for some $k \geq k_0$. We then have from (A.20) that

$$\begin{aligned} H(k+1) &= \frac{k-1}{k+1} \cdot H(k) \\ &= \frac{k-1}{k+1} \cdot \left(1 - \frac{\mu}{4L}\right)^{\tilde{k}} \cdot \frac{\tilde{k}k_0}{(k-1)k} \\ &= \left(1 - \frac{\mu}{4L}\right)^{\tilde{k}} \cdot \frac{\tilde{k}k_0}{k(k+1)}, \end{aligned}$$

which proves (A.21).

Final Bound. Combining (A.18) and (A.21) with (A.9), we obtain

$$\mathcal{F}_k = \mathbb{E}\left[F(\mathbf{z}_k)\right] - F_{\min} \leq \mathcal{O}\left(\frac{1}{k^2} + \frac{\sigma^2}{|\mathcal{D}|} + \frac{\sigma^2}{kB}\right). \tag{A.22}$$

□

A.2 Total Expectation Result of Lemma 1

We give a formal proof of the total expectation result of Lemma 1, starting with bounding the expectation of k.

Lemma A.1. *Let k denote the random variable of the number of predictor training steps after processing t requests, and ϕ denote the fixed feedback availability rate across all steps. We have*

$$\mathbb{E}\left[k\right] \leq \mathcal{O}\left(\phi t^{\frac{3}{4}}\right) \text{ and } \mathbb{E}\left[\frac{1}{k}\right] \leq \mathcal{O}\left(\frac{1}{\phi t^{\frac{3}{4}}}\right). \tag{A.23}$$

Proof. Recall that $X_t \sim \text{Bernoulli}(p_t)$ is an indicator denoting whether an exploration step occurs when processing request t. Furthermore, $Y_t \sim \text{Bernoulli}(\phi)$ is an indicator denoting whether the user provides feedback at a given exploration step, where ϕ is constant across all steps. Assuming k is the total number of SGD steps after processing t requests, and noting that $X_\tau \perp Y_\tau$, we have

$$\lambda := \mathbb{E}\left[k\right] = \mathbb{E}\left[\sum_{\tau=1}^{t} X_\tau Y_\tau\right] = \sum_{\tau=1}^{t} \mathbb{E}\left[X_\tau\right] \mathbb{E}\left[Y_\tau\right] = \sum_{\tau=1}^{t} p_\tau \phi \tag{A.24}$$

$$= \sum_{\tau=1}^{t} \phi \min\left(1, \frac{c}{\sqrt[4]{\tau}}\right) = \sum_{\tau=1}^{c^4} \phi + \sum_{\tau=c^4+1}^{t} \frac{\phi c}{\sqrt[4]{\tau}} \tag{A.25}$$

$$= c^4\phi + \frac{4\phi c}{3}(t^{\frac{3}{4}} - c^3) = \Theta\left(\phi t^{\frac{3}{4}}\right) \tag{A.26}$$

This proves the first result.

Considering $\frac{1}{k}$, we adopt the convention that $\frac{1}{k} := 0$ when $k = 0$, since no predictor update has occurred in that case and Lemma 1 (which is stated for $k \geq 1$) contributes no term. Equivalently, all expectations involving $\frac{1}{k}$ are taken on the event $\{k \geq 1\}$. We then note that

$$\mathbb{E}\left[\frac{1}{k}\right] = \mathbb{E}\left[\frac{1}{k} \cdot \mathbb{1}_{k \geq \lambda/2}\right] + \mathbb{E}\left[\frac{1}{k} \cdot \mathbb{1}_{k < \lambda/2}\right] \leq \frac{2}{\lambda} + \mathbb{E}\left[\frac{1}{k} \cdot \mathbb{1}_{k < \lambda/2}\right], \tag{A.27}$$

where $\mathbb{1}_{\mathcal{C}}$ is an indicator function of whether the condition $\mathcal{C}$ holds, and λ is as derived in (A.26) accounting for the feedback probability ϕ.

We now consider the last term in (A.27). The multiplicative Chernoff bound shows that

$$\Pr\{k \leq (1-\delta)\lambda\} \leq e^{-\frac{\delta^2 \lambda}{2}},$$

for $0 < \delta < 1$. Choosing $\delta = \frac{1}{2}$ gives

$$\Pr\left\{k \leq \frac{\lambda}{2}\right\} \leq e^{-\frac{\lambda}{8}}.$$

Under the above convention (so that $\frac{1}{k} \leq 1$ on $\{k \geq 1\}$ and $\frac{1}{k} = 0$ on $\{k = 0\}$),

$$\mathbb{E}\left[\frac{1}{k} \cdot \mathbb{1}_{k < \lambda/2}\right] \leq \mathbb{E}\left[\mathbb{1}_{k < \lambda/2}\right] \leq e^{-\frac{\lambda}{8}} = \frac{1}{e^{\frac{\lambda}{8}}} \leq \frac{1}{1 + \frac{\lambda}{8}}, \tag{A.28}$$

where the last inequality is due to the elementary relation that $e^x \geq 1 + x$ for any x. Combining (A.26), (A.27), and (A.28), we obtain

$$\mathbb{E}\left[\frac{1}{k}\right] \leq \mathcal{O}\left(\frac{1}{\phi t^{\frac{3}{4}}}\right). \tag{A.29}$$

□

We then have the following lemma.

Lemma A.2 (Formal total expectation bound of Lemma 1)**.** *Under the same conditions as for Lemma 1, we have*

$$\mathbb{E}\left[F(\mathbf{z}_k)\right] - F_{\min} \leq \mathcal{O}\left(\frac{1+\sigma^2}{t^{3/4}\phi}\right). \tag{A.30}$$

Proof. The result follows directly by noting $k = |\mathcal{D}|$ and taking total expectation to the bound in Lemma 1. □

A.3 Proof of Lemma 2

We first prove the expectation version of Lemma 2.

For ease of discussion, let ϵ denote the upper bound of $\mathbb{E}\left[F(\mathbf{z}_k)\right] - F_{\min}$ so that $\epsilon = \mathcal{O}\left(\frac{1+\sigma^2}{t^{3/4}\phi}\right)$ according to Lemma A.2. We consider $\beta = M$ in this proof.

When $\mathbb{E}\left[F(\mathbf{z}_k)\right] - F_{\min} \leq \epsilon$, from (2) and noting that the cross entropy is non-negative, we have

$$\begin{aligned}
&\mathbb{E}\left[\max_m \{-s_{m,t} \log \hat{s}_{m,t} - (1 - s_{m,t}) \log(1 - \hat{s}_{m,t})\}\right] \\
&\leq \mathbb{E}\left[\sum_{m=1}^{M} \left(-s_{m,t} \log \hat{s}_{m,t} - (1 - s_{m,t}) \log(1 - \hat{s}_{m,t})\right)\right] \\
&\leq M\mathbb{E}\left[F(\mathbf{z}_k)\right] \\
&\leq MF_{\min} + \epsilon M.
\end{aligned} \tag{A.31}$$

For arbitrary sample $\mathbf{a}_t$ and model m, let

$$\Gamma := -s_{m,t} \log \hat{s}_{m,t} - (1 - s_{m,t}) \log(1 - \hat{s}_{m,t}).$$

We consider two cases as follows. When $s_{m,t} = 0$, we have

$$\begin{aligned} &\Gamma = -\log(1 - \hat{s}_{m,t}) \\ &\Rightarrow \hat{s}_{m,t} = 1 - e^{-\Gamma} \\ &\Rightarrow |\hat{s}_{m,t} - s_{m,t}| = 1 - e^{-\Gamma}. \end{aligned}$$

When $s_{m,t} = 1$, we have

$$\begin{aligned} &\Gamma = -\log \hat{s}_{m,t} \\ &\Rightarrow 1 - \hat{s}_{m,t} = 1 - e^{-\Gamma} \\ &\Rightarrow |\hat{s}_{m,t} - s_{m,t}| = 1 - e^{-\Gamma}. \end{aligned}$$

Noting the elementary inequality $e^{-\Gamma} \geq 1 - \Gamma$, we obtain

$$|\hat{s}_{m,t} - s_{m,t}| \leq \Gamma.$$

Because this relation holds for any sample and the corresponding Γ defined on the sample, the expectation of the cross-entropy loss cannot be smaller than the expectation of the absolute difference. Combining with (A.31), we have

$$\begin{aligned} \mathbb{E}\left[\max_m |\hat{s}_{m,t} - s_{m,t}|\right] &\leq MF_{\min} + \epsilon M \\ &= MF_{\min} + \mathcal{O}\left(\frac{M(1+\sigma^2)}{t^{3/4}\phi}\right). \end{aligned} \tag{A.32}$$

The final result then follows by setting $\beta = M$.

Next, we proceed with the proof of the high probability version of Lemma 2. Applying Markov's inequality to $F(\mathbf{z}_k) - F_{\min} \geq 0$, we have

$$\Pr\left\{F(\mathbf{z}_k) - F_{\min} \geq \frac{\mathbb{E}\left[F(\mathbf{z}_k)\right] - F_{\min}}{\delta}\right\} \leq \delta, \tag{A.33}$$

so with probability at least $1 - \delta$,

$$F(\mathbf{z}_k) - F_{\min} \leq \frac{\mathbb{E}\left[F(\mathbf{z}_k)\right] - F_{\min}}{\delta} \leq \frac{\epsilon}{\delta} = \mathcal{O}\left(\frac{1+\sigma^2}{t^{3/4}\phi\delta}\right). \tag{A.34}$$

Finally, using the pointwise relation $|\hat{s}_{m,t} - s_{m,t}| \leq \Gamma$ established above and $\max_m |\hat{s}_{m,t} - s_{m,t}| \leq \sum_{m=1}^{M} \Gamma \leq MF(\mathbf{z}_k)$ (with the cross-entropy evaluated on request t, analogous to the expectation version), the high-probability bound (A.34) on $F(\mathbf{z}_k) - F_{\min}$ gives, with probability at least $1 - \delta$,

$$\max_m |\hat{s}_{m,t} - s_{m,t}| \leq MF_{\min} + M\left(F(\mathbf{z}_k) - F_{\min}\right) \leq MF_{\min} + \mathcal{O}\left(\frac{M(1+\sigma^2)}{t^{3/4}\phi\delta}\right), \tag{A.35}$$

After setting $\beta = M$, this completes the proof of Lemma 2.

A.4 Proof of Theorem 1

Let $\rho := \frac{q - \alpha - (1-\phi)\beta F_{\min}}{2}$, which is positive by Assumption 3. We invoke Lemma 2 with failure probability $\delta := \rho$. Since $\rho \leq \frac{q-\alpha}{2} < 1$, this is a valid choice with $\delta \in (0, 1)$. It gives

$$q - \delta - \alpha = q - \alpha - \rho = \frac{q - \alpha}{2} + \frac{(1-\phi)\beta F_{\min}}{2} > 0,$$

under Assumption 3, so $q - \delta = q - \rho > \alpha$.

With $\psi = \Theta\left(\left(\frac{\beta(1+\sigma^2)}{\phi\,\rho\left(\frac{1}{3}-\beta F_{\min}\right)}\right)^{4/3}\right)$ as defined in Theorem 1, every $t \geq \psi$ satisfies

$$\mathcal{O}\left(\frac{\beta(1+\sigma^2)}{t^{3/4}\phi\,\rho}\right) \leq \frac{1}{3} - \beta F_{\min}. \tag{A.36}$$

Thus, for $t \geq \psi$, the high-probability version of Lemma 2 with $\delta = \rho$ gives, with a probability of at least $1-\rho$,

$$\max_m |\hat{s}_{m,t} - s_{m,t}| \leq \beta F_{\min} + \mathcal{O}\left(\frac{\beta(1+\sigma^2)}{t^{3/4}\phi\,\delta}\right) = \beta F_{\min} + \mathcal{O}\left(\frac{\beta(1+\sigma^2)}{t^{3/4}\phi\,\rho}\right) \leq \frac{1}{3}, \tag{A.37}$$

where the last inequality uses (A.36). Therefore, with a probability of at least $1-\rho$, if $s_{m,t} = 1$ we must have $\hat{s}_{m,t} \geq \frac{2}{3}$, and if $s_{m,t} = 0$ we must have $\hat{s}_{m,t} \leq \frac{1}{3}$.

Using $1 - \phi \leq 1$ and (A.36), we can also obtain the following for $t \geq \psi$:

$$(1-\phi)\mathcal{O}\left(\frac{\beta(1+\sigma^2)}{t^{3/4}\phi}\right) \leq \mathcal{O}\left(\frac{\beta(1+\sigma^2)}{t^{3/4}\phi}\right) \leq \rho \cdot \mathcal{O}\left(\frac{\beta(1+\sigma^2)}{t^{3/4}\phi\,\rho}\right) \leq \rho\left(\frac{1}{3} - \beta F_{\min}\right) \leq \frac{\rho}{2}, \tag{A.38}$$

which will be used in the drift analysis later.

Let $\gamma := \max\{\alpha\psi; 3V\Delta_C\}$ and $Z_t := \max\{0, Q_t - \gamma\}$. For any given t, let $\tilde{m}$ denote the selected model obtained from the solution to (4), i.e., $y_{\tilde{m},t} = 1$ and $y_{m',t} = 0$ for $m' \neq \tilde{m}$.

The queue cannot reach γ before step ψ. The satisfaction signal driving the queue update (5) is in $[0,1]$ and $\alpha \in (0,1)$, so $Q_{t+1} \leq Q_t + \alpha$ holds for any t. With $Q_1 = 0$, this gives $Q_t \leq \alpha(t-1)$ for all t. Because $\gamma \geq \alpha\psi$, it follows that $Q_t \leq \alpha(t-1) < \alpha\psi \leq \gamma$ for every $t \leq \psi$, so $Z_t = 0$ throughout $t \leq \psi$. Equivalently, $Z_t > 0$ (i.e., $Q_t > \gamma$) can occur only when $t > \psi$. For this reason, in the above, we derived (A.37) and (A.38) only for the case of $t \geq \psi$.

Model selection under a large queue. We show that, when $Q_t > \gamma$, we have $\hat{s}_{\hat{m},t} - \hat{s}_{\tilde{m},t} < \frac{1}{3}$, where $\hat{m} := \arg\max_m \hat{s}_{m,t}$. The proof is by contradiction. Suppose $\hat{s}_{\hat{m},t} - \hat{s}_{\tilde{m},t} \geq \frac{1}{3}$, then choosing $\hat{m}$ instead of $\tilde{m}$, i.e., letting $y_{\hat{m},t} = 1$ and $y_{\tilde{m},t} = 0$, will reduce the second term of the objective (4a) by more than $\frac{Q_t}{3}$. The increase in the first term of (4a) due to this change is at most $V\Delta_C$. Since $Q_t > \gamma \geq 3V\Delta_C$, we know that this alternative model choice decreases the objective (4a), which contradicts that $\tilde{m}$ is the optimal model choice from (4).

The selected model is satisfactory with high probability. Consider any t with $Q_t > \gamma$. Let $A := \{\max_m |\hat{s}_{m,t} - s_{m,t}| \leq \frac{1}{3}\}$. Since $t > \psi$ when $Q_t > \gamma$, we have $\Pr\{A\} \geq 1 - \delta$. By Assumption 3, there is a model m^* with $\Pr\{s_{m^*,t} = 1\} \geq q$. On event A, whenever $s_{m^*,t} = 1$ we have $\hat{s}_{m^*,t} \geq \frac{2}{3}$, hence $\hat{s}_{\hat{m},t} \geq \frac{2}{3}$, and the selection bound above gives $\hat{s}_{\tilde{m},t} > \hat{s}_{\hat{m},t} - \frac{1}{3} \geq \frac{1}{3}$, which forces $s_{\tilde{m},t} = 1$. Thus $\{s_{m^*,t} = 1\} \cap A \subseteq \{s_{\tilde{m},t} = 1\}$, and by the union bound

$$\Pr\{s_{\tilde{m},t} = 1\} \geq \Pr\{s_{m^*,t} = 1\} - \Pr\{A^c\} \geq q - \delta.$$

Because requests are IID and Z_t is past-measurable, the same argument holds after conditioning on Z_t, giving $\mathbb{E}\left[s_{\tilde{m},t} \middle| Z_t\right] = \Pr\{s_{\tilde{m},t} = 1 \mid Z_t\} \geq q - \delta$.

Let

$$u_{\tilde{m},t} := \begin{cases} s_{\tilde{m},t}, & \text{if } s_{\tilde{m},t} \neq \emptyset \\ \hat{s}_{\tilde{m},t}, & \text{if } s_{\tilde{m},t} = \emptyset \end{cases}. \tag{A.39}$$

We also note that the queue arrival at step t satisfies $(\alpha - u_{\tilde{m},t})^2 \leq 1$, since $\alpha \in (0,1)$ and $u_{\tilde{m},t} \in [0,1]$.

Lyapunov drift. We now bound the drift of Z_t. If $Z_t = 0$, then $Z_{t+1} = \max\{0, (Q_t - \gamma) + \alpha - u_{\tilde{m},t}\} \leq \max\{0, \alpha - u_{\tilde{m},t}\} \leq 1$, so $\frac{1}{2}\mathbb{E}\left[Z_{t+1}^2 - Z_t^2 \middle| Z_t\right] \leq \frac{1}{2} = -\frac{\rho}{2}Z_t + \frac{1}{2}$ (as $Z_t = 0$). If instead $Z_t > 0$, then $Q_t > \gamma$ and $t > \psi$, so (A.37), (A.38), and $\mathbb{E}\left[s_{\tilde{m},t} \middle| Z_t\right] \geq q - \delta$ all apply, and we obtain

$$\frac{1}{2}\mathbb{E}\left[Z_{t+1}^2 - Z_t^2 \middle| Z_t\right]$$

$$
\begin{aligned}
&= \frac{1}{2}\mathbb{E}\left[\left(\max\{0, Z_t + \alpha - u_{\tilde{m},t}\}\right)^2 - Z_t^2 \middle| Z_t\right] \\
&\leq \frac{1}{2}\mathbb{E}\left[\left(Z_t + \alpha - u_{\tilde{m},t}\right)^2 - Z_t^2 \middle| Z_t\right] \\
&\leq \mathbb{E}\left[Z_t\left(\alpha - u_{\tilde{m},t}\right) + \frac{1}{2}\left(\alpha - u_{\tilde{m},t}\right)^2 \middle| Z_t\right] \\
&\overset{(a)}{\leq} \mathbb{E}\left[Z_t\left(\alpha - s_{\tilde{m},t}\right) \middle| Z_t\right] + \mathbb{E}\left[Z_t\left(s_{\tilde{m},t} - u_{\tilde{m},t}\right) \middle| Z_t\right] + \frac{1}{2} \\
&\overset{(b)}{\leq} \mathbb{E}\left[Z_t\left(\alpha - s_{\tilde{m},t}\right) \middle| Z_t\right] + (1-\phi)\, Z_t\, \mathbb{E}\left[|\hat{s}_{\tilde{m},t} - s_{\tilde{m},t}|\right] + \frac{1}{2} \\
&\overset{(c)}{\leq} Z_t\left(\alpha - (q-\delta)\right) + (1-\phi)\, Z_t\left(\beta F_{\min} + \mathcal{O}\left(\frac{\beta(1+\sigma^2)}{t^{3/4}\phi}\right)\right) + \frac{1}{2} \\
&\overset{(d)}{=} Z_t\left(-\rho + (1-\phi)\mathcal{O}\left(\frac{\beta(1+\sigma^2)}{t^{3/4}\phi}\right)\right) + \frac{1}{2} \\
&\overset{(e)}{\leq} -\frac{\rho}{2} Z_t + \frac{1}{2},
\end{aligned}
$$

where (a) adds and subtracts $s_{\tilde{m},t}$ and uses $\frac{1}{2}(\alpha - u_{\tilde{m},t})^2 \leq \frac{1}{2}$; (b) uses $s_{\tilde{m},t} - u_{\tilde{m},t} = (1 - Y_t)(s_{\tilde{m},t} - \hat{s}_{\tilde{m},t})$, where $Y_t \sim \text{Bernoulli}(\phi)$ is the feedback indicator independent of the satisfaction outcomes (Assumption 2), so that $\mathbb{E}\left[Z_t(s_{\tilde{m},t} - u_{\tilde{m},t}) \middle| Z_t\right] = (1-\phi) Z_t\, \mathbb{E}\left[s_{\tilde{m},t} - \hat{s}_{\tilde{m},t}\right] \leq (1 - \phi) Z_t\, \mathbb{E}\left[|\hat{s}_{\tilde{m},t} - s_{\tilde{m},t}|\right]$; (c) applies Lemma 2 (expectation version) together with $\mathbb{E}\left[s_{\tilde{m},t} \middle| Z_t\right] \geq q - \delta$; (d) uses $\alpha - (q-\delta) + (1-\phi)\beta F_{\min} = -\rho$, which holds by the choice $\delta = \rho = \frac{q - \alpha - (1-\phi)\beta F_{\min}}{2}$; and (e) holds because $t \geq \psi$ ensures $(1-\phi)\mathcal{O}\left(\frac{\beta(1+\sigma^2)}{t^{3/4}\phi}\right) \leq \frac{\rho}{2}$ from (A.38).

Combining the two cases, the drift bound $\frac{1}{2}\mathbb{E}\left[Z_{t+1}^2 - Z_t^2 \middle| Z_t\right] \leq -\frac{\rho}{2} Z_t + \frac{1}{2}$ holds for all $t \geq 1$. Taking total expectation gives

$$
\mathbb{E}\left[Z_{t+1}^2 - Z_t^2\right] \leq -\rho\, \mathbb{E}\left[Z_t\right] + 1.
$$

Since $Z_1 = 0$, telescoping gives

$$
\mathbb{E}\left[Z_t^2\right] \leq (t-1) - \rho \sum_{\tau=1}^{t-1} \mathbb{E}\left[Z_\tau\right], \tag{A.40}
$$

for $t > 1$.

Noting that $\rho > 0$ and $Z_\tau \geq 0$, we have $\mathbb{E}\left[Z_t^2\right] \leq t - 1$, so by Jensen's inequality,

$$
\mathbb{E}\left[Z_t\right] \leq \sqrt{\mathbb{E}\left[Z_t^2\right]} \leq \sqrt{t-1} \leq \mathcal{O}\left(\sqrt{t}\right), \tag{A.41}
$$

for $t \geq 1$.

In addition, from (A.40) and $\mathbb{E}\left[Z_t^2\right] \geq 0$, we have

$$
\rho \sum_{\tau=1}^{t-1} \mathbb{E}\left[Z_\tau\right] \leq (t-1) - \mathbb{E}\left[Z_t^2\right] \leq t - 1.
$$

Therefore,

$$
\frac{1}{t}\sum_{\tau=1}^{t} \mathbb{E}\left[Z_\tau\right] \leq \frac{1}{t}\left(\frac{t-1}{\rho} + \mathbb{E}\left[Z_t\right]\right) \leq \mathcal{O}\left(\frac{1}{\rho}\right), \tag{A.42}
$$

for $t \geq 1$.

The final result then follows by combining $Q_t \leq \gamma + Z_t$ with (A.41) and (A.42), giving the two bounds. □

A.5 Proof of Corollary 1

Using the definition in (A.39), the virtual queue update (5) can be written as $Q_{t+1} = \max\{0, Q_t + \alpha - u_{\tilde{m},t}\}$. We have $Q_{t+1} \geq Q_t + \alpha - u_{\tilde{m},t}$, i.e., $\alpha - u_{\tilde{m},t} \leq Q_{t+1} - Q_t$. Summing over $t = 1, \ldots, T$ and using $Q_1 = 0$ gives

$$\sum_{t=1}^{T} (\alpha - u_{\tilde{m},t}) \leq Q_{T+1} - Q_1 = Q_{T+1}, \tag{A.43}$$

so that

$$\frac{1}{T}\sum_{t=1}^{T} u_{\tilde{m},t} \geq \alpha - \frac{Q_{T+1}}{T}. \tag{A.44}$$

Let $Y_t \sim \text{Bernoulli}(\phi)$ denote the feedback-availability indicator at step t, which is independent of the routing decision and the satisfaction outcomes under Assumption 2. By the definition of $u_{\tilde{m},t}$ in (A.39), we have $u_{\tilde{m},t} = Y_t\, s_{\tilde{m},t} + (1 - Y_t)\, \hat{s}_{\tilde{m},t}$. Taking expectation and using the independence of Y_t, we have

$$\mathbb{E}\left[u_{\tilde{m},t}\right] = \phi\, \mathbb{E}\left[s_{\tilde{m},t}\right] + (1-\phi)\mathbb{E}\left[\hat{s}_{\tilde{m},t}\right] = \mathbb{E}\left[s_{\tilde{m},t}\right] + (1-\phi)\mathbb{E}\left[\hat{s}_{\tilde{m},t} - s_{\tilde{m},t}\right]. \tag{A.45}$$

Since $\hat{s}_{\tilde{m},t} - s_{\tilde{m},t} \leq |\hat{s}_{\tilde{m},t} - s_{\tilde{m},t}| \leq \max_m |\hat{s}_{m,t} - s_{m,t}|$, the expectation version of Lemma 2 gives

$$\mathbb{E}\left[u_{\tilde{m},t}\right] \leq \mathbb{E}\left[s_{\tilde{m},t}\right] + (1-\phi)\left(\beta F_{\min} + \mathcal{O}\left(\frac{\beta(1+\sigma^2)}{t^{3/4}\phi}\right)\right). \tag{A.46}$$

Because only one model serves request t, we have $\sum_{m=1}^{M} y_{m,t} s_{m,t} = s_{\tilde{m},t}$. Combining (A.44) and (A.46) after taking total expectation, and noting that $\frac{1}{T}\sum_{t=1}^{T} t^{-\frac{3}{4}} = \mathcal{O}(T^{-\frac{3}{4}})$, we obtain

$$\alpha - \frac{1}{T}\sum_{t=1}^{T}\sum_{m=1}^{M} \mathbb{E}\left[y_{m,t} s_{m,t}\right] \leq \frac{\mathbb{E}\left[Q_{T+1}\right]}{T} + (1-\phi)\beta F_{\min} + \mathcal{O}\left(\frac{(1-\phi)\beta(1+\sigma^2)}{\phi\, T^{3/4}}\right). \tag{A.47}$$

Finally, applying the queue-length bound $\mathbb{E}\left[Q_{T+1}\right] \leq \gamma + \mathcal{O}(\sqrt{T})$ from Theorem 1 gives $\frac{\mathbb{E}[Q_{T+1}]}{T} \leq \frac{\gamma}{T} + \mathcal{O}\left(\frac{1}{\sqrt{T}}\right)$. Since $T^{-\frac{3}{4}} \leq \mathcal{O}(T^{-\frac{1}{2}})$, the last term is absorbed, yielding

$$\alpha - \frac{1}{T}\sum_{t=1}^{T}\sum_{m=1}^{M} \mathbb{E}\left[y_{m,t} s_{m,t}\right] \leq \mathcal{O}\left(\frac{\gamma}{T} + \frac{1}{\sqrt{T}}\right) + (1-\phi)\beta F_{\min}. \tag{A.48}$$

A.6 Proof of Theorem 2

We first consider any t such that $X_t = 0$, i.e., no exploration or predictor training. As before, let $\tilde{m}$ denote the solution to (4) for a given t, i.e., $y_{\tilde{m},t} = 1$ and $y_{m',t} = 0$ for $m' \neq \tilde{m}$. We also use the definition in (A.39). We consider the following Lyapunov drift of the queue length:

$$\begin{aligned}
\frac{1}{2}\mathbb{E}\left[Q_{t+1}^2 - Q_t^2 \middle| Q_t\right] &\leq \frac{1}{2}\mathbb{E}\left[\left(Q_t + \alpha - u_{\tilde{m},t}\right)^2 - Q_t^2 \middle| Q_t\right] \\
&\leq \frac{1}{2}\mathbb{E}\left[2Q_t(\alpha - u_{\tilde{m},t}) + 1 \middle| Q_t\right] \\
&\leq \mathbb{E}\left[Q_t(\alpha - \hat{s}_{\tilde{m},t}) \middle| Q_t\right] + Q_t\mathbb{E}\left[\max_m |\hat{s}_{m,t} - s_{m,t}|\right] + \frac{1}{2},
\end{aligned} \tag{A.49}$$

where the second inequality uses $(\alpha - u_{\tilde{m},t})^2 \leq 1$, and the third adds and subtracts $\hat{s}_{\tilde{m},t}$, considers that $u_{\tilde{m},t}$ is either $s_{\tilde{m},t}$ or $\hat{s}_{\tilde{m},t}$, and bounds the resulting estimation error term.

Let $\{y_{m,t}^{\text{OPT}}, \forall m\}$ denote an optimal stationary policy for (1). Since $\tilde{m}$ minimizes $VC_{m,t} + Q_t(\alpha - \hat{s}_{m,t})$ over all models, we have

$$\mathbb{E}\left[VC_{\tilde{m},t} + Q_t\left(\alpha - \hat{s}_{\tilde{m},t}\right) \middle| Q_t\right]$$

$$\leq V\mathbb{E}\left[\sum_{m=1}^{M} y_{m,t}^{\mathrm{OPT}} C_{m,t}\right] + Q_t\mathbb{E}\left[\sum_{m=1}^{M} y_{m,t}^{\mathrm{OPT}}(s_{m,t}-\hat{s}_{m,t})\right] + Q_t\mathbb{E}\left[\sum_{m=1}^{M} y_{m,t}^{\mathrm{OPT}}(\alpha - s_{m,t})\right]$$
$$\leq VC^{\mathrm{OPT}} + Q_t\mathbb{E}\left[\max_m |\hat{s}_{m,t}-s_{m,t}|\right], \tag{A.50}$$

where the last inequality uses the fact that the optimal stationary policy satisfies (1b) with equality, so $\mathbb{E}\left[\sum_m y_{m,t}^{\mathrm{OPT}}(\alpha - s_{m,t})\right] = 0$.

Combining (A.49) and (A.50), canceling out $\mathbb{E}\left[Q_t(\alpha - \hat{s}_{\tilde{m},t})|\, Q_t\right]$ on both sides, and taking total expectation gives

$$\frac{1}{2}\mathbb{E}\left[Q_{t+1}^2 - Q_t^2\right] + V\mathbb{E}\left[\sum_{m=1}^{M} y_{m,t}C_{m,t}\right] \leq VC^{\mathrm{OPT}} + 2\mathbb{E}\left[Q_t\right]\cdot\mathbb{E}\left[\max_m |\hat{s}_{m,t}-s_{m,t}|\right] + \frac{1}{2}.$$

Rearranging and applying Lemma 2 yields

$$\mathbb{E}\left[\sum_{m=1}^{M} y_{m,t}C_{m,t}\right] \leq C^{\mathrm{OPT}} + \frac{2}{V}\mathbb{E}\left[Q_t\right]\cdot\left(\beta F_{\min} + \mathcal{O}\left(\frac{\beta(1+\sigma^2)}{t^{3/4}\phi}\right)\right) + \frac{1}{2V} + \frac{1}{2V}\mathbb{E}\left[Q_t^2 - Q_{t+1}^2\right].$$

Applying Theorem 1 to bound $\mathbb{E}\left[Q_t\right] \leq \gamma + \mathcal{O}\left(\sqrt{t}\right)$ and absorbing $1+\sigma^2$ into $\mathcal{O}(\cdot)$, we obtain

$$\mathbb{E}\left[\sum_{m=1}^{M} y_{m,t}C_{m,t}\right] \leq C^{\mathrm{OPT}} + \frac{2}{V}\left(\mathcal{O}\left((\gamma+\sqrt{t})\,\frac{\beta}{t^{3/4}}\right) + \beta F_{\min}\mathbb{E}\left[Q_t\right]\right) + \frac{1}{2V} + \frac{1}{2V}\mathbb{E}\left[Q_t^2 - Q_{t+1}^2\right].$$

For any t with $X_t = 1$, the cost incurred during exploration is at most $C_{\max}$. Averaging over all T requests and telescoping the queue term, we obtain

$$\mathbb{E}\left[\frac{1}{T}\sum_{t=1}^{T}\sum_{m=1}^{M} y_{m,t}C_{m,t}\right]$$
$$\leq C^{\mathrm{OPT}} + \frac{2}{V}\left(\frac{1}{T}\sum_{t=1}^{T}\mathcal{O}\left((\gamma+\sqrt{t})\,\frac{\beta}{t^{\frac{3}{4}}}\right) + \frac{\beta F_{\min}}{T}\sum_{t=1}^{T}\mathbb{E}\left[Q_t\right]\right) + \frac{1}{2V} + \frac{\mathbb{E}\left[K\right]C_{\max}}{T}$$
$$\leq C^{\mathrm{OPT}} + \mathcal{O}\left(\frac{1}{T}\sum_{t=1}^{T}\frac{\beta}{t^{\frac{1}{4}}} + \beta F_{\min} + \frac{1}{V} + \frac{C_{\max}}{T^{\frac{1}{4}}}\right)$$
$$= C^{\mathrm{OPT}} + \mathcal{O}\left(\frac{\beta}{\sqrt[4]{T}} + \beta F_{\min} + \frac{1}{V}\right),$$

where the second inequality applies Theorem 1 together with $\mathbb{E}\left[K\right] = \sum_{t=1}^{T} p_t = \Theta(T^{3/4})$, which follows from the same computation as in the proof of Lemma A.1 but without the feedback factor ϕ, because exploration cost is incurred regardless of whether feedback is received or not (we only know whether there is feedback after the model is chosen), and the final equality absorbs the fixed cost scale $C_{\max}$ into $\mathcal{O}(\cdot)$.

B Experimental Details & Additional Results

We provide a ready-to-run code base on GitHub: `https://anonymous.4open.science/r/slarouter-F044`. This section provides additional details to reproduce our results and additional experimental results.

B.1 Environment and Hardware

All experiments were executed in a single Python 3.12 environment managed with `uv`. A full list of locked dependencies can be found in the code base.

Random seeding. We seed all random generators in our experiment, usually with seeds 42, 1, and 1234, if not specified otherwise. For experiment where we use more than 3 seeds, we use the range $[1, n]$.

Hardware. We conducted all our experiments on a SLURM cluster equipped with NVIDIA H100 GPUs. To collect the benchmark data, we used 1 GPU for Qwen 3.5 2B, 9B-AWQ, and 35B-AWQ, as well as Llama 3.3 70B. We used 2 GPUs to run Qwen 3.5 122B-AWQ. Otherwise, all experiments presented in this paper require little GPU memory since the routing models are typically $\ll 1$B parameters.

B.2 Dataset

We collect all benchmark data by using EleutherAI's `lm-evaluation-harness` (Gao et al., 2023). We then merge the logged json outputs into a single parquet file that can be consumed by our code base. The dataset contains the cost for querying all LLMs for a specific request (in USD) as well as the actual model response. The responses are particularly relevant for the experiments with the LLM judge. We measure inference energy using the `zeus.ml` library (You et al., 2023). Each row in the dataset corresponds to one evaluation example and contains:

- Identifiers: `doc_id`, `benchmark`, `input_text`, `correct_answer`
- Per-model binary outcomes: `{model}_solved`
- Per-model energy/latency: `{model}_energy_joules`, `{model}_latency_ms`
- Per-model token counts: `{model}_prompt_tokens`, `{model}_completion_tokens`

Model zoo. We use one model zoo with 5 distinct models, specifically

1. `Qwen/Qwen3.5-2B`,
2. `QuantTrio/Qwen3.5-9B-AWQ`,
3. `QuantTrio/Qwen3.5-35B-A3B-AWQ`,
4. `casperhansen/Llama-3.3-70B-Instruct-AWQ`, and
5. `QuantTrio/Qwen3.5-122B-A10B-AWQ`,

where the Llama checkpoint is the same 4-bit AWQ quantization scheme used for the Qwen models, keeping the serving setup comparable across families.

Pricing. The dollar-denominated operating costs reported in Tables 2 and 3 use per-token list pricing retrieved from OpenRouter (`https://openrouter.ai`) for each model in the zoo, applied to the prompt and completion token counts logged for every request (OpenRouter, 2026). We use OpenRouter because it publishes a single, third-party, per-token USD price for each of our models (including the quantized checkpoints), giving a cost figure that is directly comparable to what a practitioner would pay through a commercial inference API. Table 4 lists the input and output price per 1M tokens used for each model.

Note the non-monotonicity in output price: Llama-3.3-70B-Instruct-AWQ ($0.32/1M) is more than 3× cheaper than Qwen3.5-35B-A3B-AWQ ($1.00/1M) despite having twice the parameter count, reflecting market-driven serving costs on OpenRouter rather than a price that scales with model size. This is the source of the non-linear price-performance relationship discussed in the main text. In all of our experiments, we always assume there are no cached tokens for any of our requests.

Table 4: OpenRouter list price (USD per 1M tokens) used to compute the dollar-denominated operating costs reported in the main paper.

Model	Input price ($/1M tok)	Output price ($/1M tok)
Qwen3.5-2B	0.02	0.10
Qwen3.5-9B-AWQ	0.10	0.15
Qwen3.5-35B-A3B-AWQ	0.14	1.00
Llama-3.3-70B-Instruct-AWQ	0.10	0.32
Qwen3.5-122B-A10B-AWQ	0.26	2.08

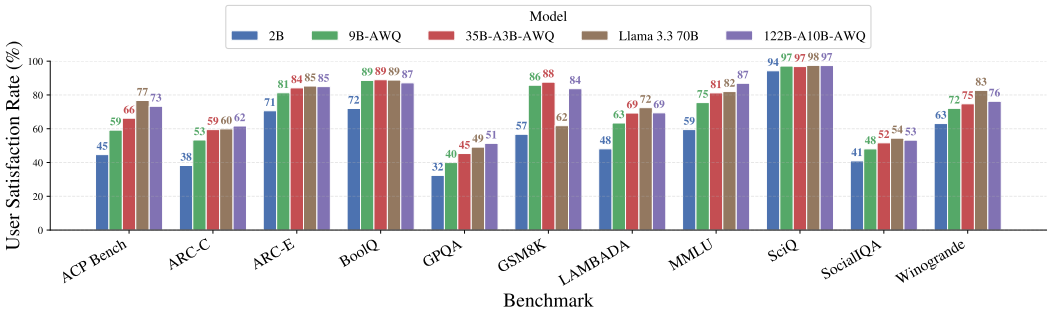


Figure B.1: Per-model accuracy across all 11 benchmarks. Each group of bars corresponds to one benchmark, with individual bars showing the accuracy of each Qwen 3.5 model. Accuracy values are printed on top of each bar.

Model performance. Figure B.1 shows per-model accuracy across all 11 benchmarks. Each group of bars corresponds to one benchmark, with individual bars showing the accuracy of each model in the zoo. The chart makes the cross-family non-monotonicity discussed in the main text directly visible: Llama 3.3 70B (brown) is not simply interpolated between the 35B and 122B Qwen models in terms of accuracy. It edges out the much larger 122B model on ACPBench, BoolQ, LAMBADA, SciQ, SocialIQA, and WinoGrande, yet trails it on GPQA, GSM8K, MMLU, and, more narrowly, ARC Challenge. Combined with its lower per-token price on several of these benchmarks (Table 4), Llama 3.3 70B is at times simultaneously cheaper and more accurate than the 35B and 122B Qwen models, which is precisely the situation a router must detect from data rather than assume from model size. This also explains why no single fixed model can satisfy α across all benchmarks (Table 2) and why we need routing.

B.3 Predictor: Architecture and Training Details

Architecture. The satisfaction predictor is a multi-label binary classifier built on top of a frozen ModernBERT encoder ($D = 768$ dimensions). The classification head is a two-layer MLP: $\texttt{Dropout}(p = 0.1) \to \texttt{Linear}(D, D) \to \texttt{LayerNorm} \to \texttt{ReLU} \to \texttt{Dropout}(p = 0.1) \to \texttt{Linear}(D, |M|)$, where M is the number of candidate models. At inference time the model outputs M logits, one per model head, which are passed through a sigmoid to obtain $\hat{s}_m \in [0, 1]^M$.

Optimizer and training hyperparameters. Only the classifier-head parameters are trained; the encoder remains frozen. Optimization uses SGD with momentum 0.90 and weight decay 0.01. The default learning rate is $\eta = 0.006$ (identified via extensive hyperparameter tuning). Gradients are clipped to a maximum ℓ_2-norm of 1.0.

Constant versus diminishing learning rate. Our analysis (Lemma 1) uses the diminishing schedule $\eta_k = \min\{4/\mu(k+1),\ 1/2L\}$, which is the standard technique for last-iterate convergence to the exact minimizer $F_{\min}$. In practice we found a constant rate already sufficient for good performance, so we use it for simplicity. The only cost is convergence to an $\mathcal{O}(\eta\sigma^2)$ neighborhood of $F_{\min}$ instead of $F_{\min}$ exactly, which inflates the floor $\beta F_{\min}$ by a constant and leaves all guarantees intact (they already carry this floor).

Loss function. Training minimizes a masked binary cross-entropy loss. Because each exploration step reveals feedback for only one model, the label matrix contains NaN for all unobserved heads; the loss back-propagates only through the observed entries.

Table 5: Matched cost sensitivity hyperparameters for CARROT (μ) and Causal rm_interval (λ) routers, tuned to meet SLARouter target accuracy α per benchmark.

Benchmark	$\boldsymbol{\mu}$ (CARROT)	$\boldsymbol{\lambda}$ (Causal)
ACPBench ($\alpha = 0.69$)	0.5	0.00
ARC Challenge ($\alpha = 0.56$)	0.8	0.01
ARC-Easy ($\alpha = 0.82$)	0.8	0.01
BoolQ ($\alpha = 0.85$)	0.7	0.01
GPQA ($\alpha = 0.47$)	0.3	0.00
GSM8K ($\alpha = 0.80$)	0.6	0.00
LAMBADA ($\alpha = 0.66$)	0.6	0.01
MMLU ($\alpha = 0.80$)	0.7	0.00
SciQ ($\alpha = 0.97$)	0.8	0.01
SocialIQA ($\alpha = 0.51$)	0.8	0.05
WinoGrande ($\alpha = 0.78$)	0.7	0.00

Minibatch buffer. Training occurs only when the minibatch buffer reaches capacity `batch_size` $= 16$. The buffer stores tuples $(\text{text}, m, \text{satisfaction})$. We sample minibatches randomly from the buffer for every predictor update. When full, the buffer yields a label matrix of shape (B, M) with exactly one label per request. By default a single SGD step is taken per full minibatch (`classifier_epochs` $= 1$); setting this > 1 enables multiple passes.

Default hyperparameters (full list). $\eta = 0.006$, momentum$= 0.90$, weight decay$= 0.01$, batch size$= 16$, dropout$= 0.1$, queue mode$=$ `random`, explore mode$=$ `uniform`, classifier epochs$= 1$, `lr_schedule` $=$ `fixed`.

B.4 Performance Matching Between Baselines and SLARouter

We carefully tune the cost parameters of CARROT and Causal Router to match the performance requirements we set for each benchmarks. For CARROT, we search the range $\mu \in [0, 1]$ with increments of 0.05 and for Causal Router we search $\lambda \in [0, 1]$ in increments of 0.01. We match the performance against the SLA requirement α and pick the configuration that comes closest to α. In order to ensure there is enough training data available for the baselines, we chose `-target-feedback-rate 0.2`. To train the CARROT and Causal Router classifiers, we use a separate training dataset that consists of the benchmark trainings splits. We collect the data in the same way per our procedure described above for the actual benchmark datasets. We train the classifiers with the same amount of feedback as what MESS+ and SLARouter see at inference time to ensure fair comparison. Full training scripts are provided in the code base. The detailed parameterization is provided in Table 5.

The two baseline classifiers fail in complementary ways. CARROT RoBERTa reaches high recall at the cost of low overall accuracy: it is biased toward predicting that a response will satisfy the user, which lets it recover most true positives but at the expense of many false positives. This is consistent with the aggressive, satisfaction-prioritizing hyperparameters we describe below. Causal RM-Softmax shows the opposite imbalance, with low precision and recall alike and an undefined ROC-AUC, because its softmax output collapses to a near-constant score across our validation split, leaving no threshold at which the ROC curve is well-defined. Both patterns motivate why we tune μ and λ per benchmark, rather than reusing a single classifier configuration, when matching these baselines against SLARouter's target accuracy α.

Table 5 shows that μ and λ broadly track the strictness of α rather than any property of the benchmark's domain: GPQA and ACPBench are the only benchmarks that need $\mu \leq 0.5$, whereas the three hardest targets ($\alpha \geq 0.82$: ARC-Easy, BoolQ, and SciQ) all sit at $\mu \geq 0.7$, with SciQ and ARC-Easy reaching $\mu = 0.8$, the upper end of our search range. The relationship is not strictly monotonic, since SocialIQA already requires $\mu = 0.8$ at a comparatively low $\alpha = 0.51$, but the overall pattern is the quantitative counterpart to the claim in the main text that CARROT and the Causal router must be pushed toward configurations that strongly prioritize user satisfaction over cost to meet α: as the target gets harder to reach on average, both baselines compensate by turning up their cost-sensitivity parameter, which is exactly the hyperparameter that is fixed and untunable at inference time once deployed.

B.5 NON-STATIONARY BENCHMARK SCENARIOS

We conduct a non-stationary benchmark experiment by concatenating the three most challenging benchmarks in our setup, specifically ACPBench, GPQA, and WinoGrande. The sudden shift in incoming requests causes our classifier to adapt to the change in distribution. We benchmark SLARouter and MESS+ under identical conditions with $\alpha = 72\%$, which is the average across the 11 benchmarks we present in Table 2. For WinoGrande and ACPBench, the most powerful models in our zoo are able to reach α while for GPQA the gap is at 23.7 percentage points between the most powerful model and the α requirement. Results are available in Table 6.

Table 6: Non-stationary benchmark

Method	WinoGrande			GPQA			ACPBench			Total		
	User Satisf. (%)	Operating Cost ($)	MCR (%)	User Satisf. (%)	Operating Cost ($)	MCR (%)	User Satisf. (%)	Operating Cost ($)	MCR (%)	User Satisf. (%)	Operating Cost ($)	MCR (%)
MESS+	$77.2_{\pm 1.2}$	$\$0.020_{\pm 0.001}$	5%/4%/7%/49%/35%	$43.7_{\pm 0.5}$	$\$0.071_{\pm 0.003}$	8%/4%/17%/41%/30%	$70.3_{\pm 7.7}$	$\$0.577_{\pm 0.027}$	8%/6%/13%/44%/29%	$59.3_{\pm 5.1}$	$\$0.668_{\pm 0.037}$	7%/5%/10%/46%/32%
SLARouter	$77.3_{\pm 1.1}$	$\$0.018_{\pm 0.004}$	9%/5%/6%/52%/28%	$48.4_{\pm 1.8}$	$\$0.092_{\pm 0.011}$	4%/5%/11%/56%/24%	$71.2_{\pm 1.8}$	$\$0.557_{\pm 0.103}$	9%/5%/17%/45%/24%	$60.4_{\pm 1.3}$	$\$0.667_{\pm 0.107}$	8%/5%/11%/50%/26%

SLARouter matches or exceeds MESS+ on user satisfaction on every segment of the concatenated stream, with the largest gap on GPQA (48.4% vs. 43.7%), the segment with the largest shortfall relative to $\alpha = 75\%$ and therefore the one where accurate constraint tracking matters most. This comes at a higher operating cost on GPQA ($0.092 vs. $0.071) but a lower cost on WinoGrande and ACPBench, so that the two methods end up within $0.001 of each other in total operating cost while SLARouter sustains a consistently higher satisfaction rate. The model-call ratios (2B/9B/35B/70B/122B) also shift visibly at each segment boundary for both methods, confirming that the predictor is reacting to the distribution shift rather than converging to a fixed routing policy; SLARouter shifts weight toward the 70B and 122B tiers on GPQA specifically, consistent with these being the two most accurate models on that benchmark in Figure B.1.

B.6 JUDGE FEEDBACK ALIGNMENT

When user feedback is unavailable, we use an LLM-based judge as a scalable alternative to obtain synthetic feedback signals. Specifically, we prompt GPT-5.4 to assess the input/response pairs, and collect binary (correct/incorrect) judgment scores. We measure the alignment (agreement rate) between the judge's outputs and ground-truth benchmark labels across two challenging evaluations: GPQA (Rein et al., 2024), a graduate-level science benchmark requiring domain expertise, and ACPBench (Kokel et al., 2025), a planning benchmark that demands multi-step reasoning. We see that disabling reasoning leads to poor alignment (Table 7) and substantially higher error rates across both benchmarks. This confirms the assumption that the SOTA reasoning-capable LLMs are necessary to reliably replicate the user feedback. In our experiments we use the latter setting with feedback collected from a reasoning-enabled judge. Our prompt template to obtain feedback from the judge is geared towards receiving binary feedback to serve as a plug-and-play substitute for actual user feedback (Listing 1). The template is rendered with a lightweight Jinja-style templating syntax, where each `{% ... %}` block is a branch that is filled in or dropped depending on whether the underlying benchmark question has a fixed set of labeled answer choices (e.g., multiple-choice questions) or requires a free-form answer (e.g., a number, a yes/no judgment, or a completion). Both branches enforce the same anti-anchoring discipline: the judge is instructed to independently derive the correct answer from the question alone, *before* it is shown the model's response, so that its judgment is not biased by the model's phrasing or by superficial plausibility. Only after committing to its own answer does the judge compare it against the response under branch-specific normalization rules (numeric equivalence, yes/no synonyms, or paraphrase tolerance for word- and phrase-level answers), and it is required to return exactly one machine-parseable JSON label, which lets us plug its output directly into the same feedback-collection code path used for real user feedback in Algorithm 1 without any additional parsing logic.

Listing 1: GPT-5.4 judge prompt template

```
---SYSTEM---
You are an expert evaluator for AI benchmark answers.

{%
This question has a fixed set of labeled answer choices - the model
must select exactly one.
```

```
Follow this process in order:
1. Read the question and all answer choices.
2. Independently determine which choice is correct - do this BEFORE
reading the model's response.
3. Compare: does the model's final answer match the correct choice?

{%
This question requires a specific answer (a number, yes/no, word, or
sentence completion) - there are no labeled choices.

Follow this process in order:
1. Read the question and any provided passage or context.
2. Independently determine the correct answer using your knowledge
and reasoning - do this BEFORE reading the model's response.
   - Math: solve step by step and identify the final numerical answer.

   - Yes/No: derive from the passage or your knowledge.
   - Word/phrase prediction: identify the correct word or phrase from
 context.
   - Fill-in-the-blank: determine which completion is logically and
grammatically correct.
3. Compare: does the model's answer match the correct answer?
   - Numbers: treat equivalent forms as equal ("18", "\$18", "18.0"
all match 18).
   - Yes/No: treat "True"/"1" as yes and "False"/"0" as no.
   - Words/phrases: accept exact match or an unambiguous paraphrase.
{%

Scoring:
- 1 (Correct): Model's final answer matches the correct answer.
- 0 (Incorrect): Model's answer is wrong, ambiguous, or the model
failed to commit.

---USER---
{%
Think through the correct answer first, then evaluate the model.
{%

Question:
{{ input_text }}

{%
Answer choices:
{{ choices }}
{%

Model response to evaluate:
{{ response }}

Respond ONLY with valid JSON:
{"label": <0 or 1>}
```

Table 7: Agreement rate between the judge (GPT-5.4) and the user feedback (ground truth labels).

Benchmark	Reasoning Enabled	Agreement Rate	False Positive Rate	False Negative Rate
ACPBench	×	68%	26.3%	35.5%
GPQA	×	62%	32.1%	45.5%
ACPBench	✓	94.4%	6.6%	4.9%
GPQA	✓	87.9%	12.9%	10.9%

Enabling reasoning rebalances the error profile. Without reasoning, the judge's false negative rate exceeds its false positive rate on both benchmarks (35.5% vs. 26.3% on ACPBench, 45.5% vs. 32.1% on GPQA), meaning the non-reasoning judge is disproportionately likely to mark a correct model response as incorrect, understating true performance. With reasoning enabled, both error rates drop

substantially and become closer to symmetric (6.6% vs. 4.9% on ACPBench, 12.9% vs. 10.9% on GPQA), with a small remaining skew toward false positives. Because a false positive feeds an overly optimistic satisfaction label into the predictor while a false negative feeds an overly pessimistic one, this shift matters for SLARouter beyond the raw agreement rate: a judge with a large, direction-biased error rate would systematically distort the virtual queue update in (5), whereas the near-symmetric error profile of the reasoning-enabled judge means its residual errors act closer to unbiased noise around the true satisfaction signal. This is also why we consistently enable reasoning in the judge deployment used throughout Section 5.2, rather than treating it as an optional cost-saving toggle.