Causal LLM Routing: End-to-End Regret Minimization from Observational Data

Asterios Tsiourvas
MIT

Wei Sun
IBM Research

Georgia Perakis
MIT

Abstract

LLM routing aims to select the most appropriate model for each query, balancing competing performance metrics such as accuracy and cost across a pool of language models. Prior approaches typically adopt a *decoupled* strategy, where metrics are first predicted and the model is then selected based on these estimates. This setup is prone to compounding errors and often relies on *full-feedback* data, where each query is evaluated by all candidate models, which is costly to obtain and maintain in practice. In contrast, we learn from *observational data*, which records only the outcome of the model actually deployed. We propose a *causal end-to-end* framework that learns routing policies by minimizing decision-making regret from observational data. To enable efficient optimization, we introduce two theoretically grounded surrogate objectives: a classification-based upper bound, and a softmax-weighted regret approximation shown to recover the optimal policy at convergence. We further extend our framework to handle heterogeneous cost preferences via an interval-conditioned architecture. Experiments on public benchmarks show that our method outperforms existing baselines, achieving state-of-the-art performance across different embedding models.

1 Introduction

LLM routing is an emerging research area focused on optimizing model selection for each input query, balancing performance and cost across a pool of available LLMs. Because LLM performance varies significantly by task and input [Hu et al., 2024], as well as computational cost [Ong et al., 2024], dynamic routing strategies have been proposed to select the most suitable model per query [Shnitzer et al., 2023]. This challenge becomes even more critical in agentic applications, where multiple LLM calls may be made within a single workflow, making efficient model selection essential for user experience and resource allocation. As LLM deployment scales, routing also contributes to environmental sustainability by reducing unnecessary computation [Singh et al., 2025].

Routing methods can be broadly classified based on whether they invoke one or multiple models per query. *Multi-model* approaches include *non-predictive routing*, which cascades models sequentially from light to heavy [Wang et al., 2023, Chen et al., 2023], and *predictive ensembles*, which learn to combine outputs or scores from multiple LLMs [Shekhar et al., 2024, Huang et al., 2025]. While these methods can improve accuracy and robustness, they incur significant computational cost and latency due to repeated inference. By contrast, *predictive routing* methods, including our approach, select a single LLM for each query by training a router that maps input queries to the most appropriate model [Shnitzer et al., 2023, Ong et al., 2024, Somerstep et al., 2025]. This framework offers a more scalable and cost-efficient solution, particularly in settings where minimizing latency and compute resources is critical.

A common formulation in predictive routing is to maximize a utility function of the form $y_x(t) = a_x(t) - \lambda \cdot c_x(t)$, where $a_x(t)$ and $c_x(t)$ denote the quality (e.g., accuracy) and cost for model t on a given query x, and $\lambda \geq 0$ captures user cost sensitivity, or willingness-to-pay. Existing methods

39th Conference on Neural Information Processing Systems (NeurIPS 2025).

typically adopt a *decoupled* approach: separate predictors are trained for each metric, and routing is performed by selecting the model with the highest estimated utility. However, decision quality is highly sensitive to these predictors, and errors can compound, especially when incorporating additional metrics (e.g., latency, faithfulness, alignment), increasing complexity and uncertainty.

Another fundamental limitation in the existing literature on predictive routing is its reliance on *full-feedback* datasets. Prior work [Ong et al., 2024] [Somerstep et al., 2025] assumes access to data where each query has been evaluated by all available LLMs. This assumption is impractical: (i) the computational and monetary cost of exhaustively querying all models is prohibitive; and (ii) the rapid pace of LLM development makes it challenging to maintain comprehensive, up-to-date evaluation datasets. In contrast, *observational data*, where each query is evaluated by only one model, is readily available from real-world LLM deployments, making it far more scalable than full feedback datasets. However, it introduces *treatment bias* from historical routing policies, which can lead to suboptimal decisions if not properly addressed [Swaminathan and Joachims, 2015, Künzel et al., 2019].

To the best of our knowledge, this is the first work to (i) learn LLM routing from observational data and (ii) introduce an integrated learning framework for routing. Our main contributions are:

- We propose a *causal end-to-end* framework that learns routing policies by directly minimizing decision-making regret using observational data. Unlike the predominant decoupled paradigm, where various performance metrics (e.g., accuracy, cost) are first predicted and then used to inform routing decisions, our method integrates prediction and prescription into a unified objective. By optimizing for regret directly, the framework is explicitly aligned with the final routing decision quality. Furthermore, it is designed to scale efficiently and leverage readily available observational data, while accounting for treatment bias without requiring costly full-feedback datasets.
- As the original regret minimization objective is not directly differentiable, we derive two *surrogate objectives* to enable end-to-end policy learning. The first is a classification-based upper bound that reframes regret minimization as a multiclass prediction problem under mild Lipschitz assumptions, allowing efficient training with standard methods. The second is a softmax-weighted regret surrogate that smoothly approximates regret using a softmax distribution and provably recovers optimal decisions at convergence.
- We extend our framework to support *heterogeneous cost preferences* by introducing a unified model that conditions on both the query and the user's cost sensitivity. Leveraging the affine structure of the utility function, we design an efficient interpolation scheme using only two endpoint models per interval. We theoretically show that the optimal treatment is piecewise constant in the cost parameter and that our architecture can exactly represent the optimal policy, enabling flexible and scalable routing across diverse preferences.
- We conduct *comprehensive experiments* on two public benchmarks, demonstrating that our regret-minimizing and heterogeneous cost-aware approaches consistently outperform existing baselines. Our methods achieve state-of-the-art performance across both BERT-based and LLaMA-based embeddings, highlighting their robustness and practical effectiveness.

2 Methodology

2.1 Problem Formulation

We consider a dataset of n observational samples, denoted by $\mathcal{D} = \{(x_i, t_i, a_i, c_i)\}_{i=1}^{n}$, where each sample is independently drawn from the joint distribution $p(x, t, a, c)$. Here, $x_i \in \mathcal{X} \subset \mathbb{R}^d$ denotes a feature vector, typically an embedding, that characterizes the query; $t_i \in [\mathcal{T}] := \{0, 1, \ldots, T-1\}$ specifies the LLM assigned to the query; $a_i \in \mathbb{R}_{\geq 0}$ denotes a numeric quality score of the LLM's response, such as accuracy, or a preference rating; and $c_i \in \mathbb{R}_{\geq 0}$ represents the cost incurred by model t_i when processing query x_i.

Given a query $x \in \mathcal{X}$, the objective of an LLM router is to select a model $t \in \mathcal{T}$ that maximizes the cost-aware performance, or utility, defined as $y_x(t) := a_x(t) - \lambda \cdot c_x(t)$. Here, $\lambda \geq 0$ is a user-specified parameter, modeling the trade-off between accuracy and cost. Higher values of $y_x(t)$, corresponding to greater performance, are preferred.

2.2 End-to-End Regret Minimization

Our goal is to route each prompt x to the LLM t such that the decision leads to the highest possible utility $y_x(t)$. To this end, we aim to learn an end-to-end policy $f : \mathcal{X} \to \mathcal{T}$ that minimizes the decision-making regret [Fernández-Loría and Provost, 2022, Zou et al., 2022]. More formally,

$$f^* := \arg\min_f Regret(f)$$

where regret is defined as

$$Regret(f) := \mathbb{E}_X[Y_X(t_X^*) - Y_X(f(X))] = \mathbb{E}_X[Y_X(t_X^*)] - \mathbb{E}_X[Y_X(f(X))], \tag{1}$$

with $t_X^* := \arg\max_{t\in\mathcal{T}} Y_X(t)$.

We want to point out that unlike full-feedback datasets used in prior routing work, which record outcomes for all models $t \in \mathcal{T}$, observational datasets contain only *partial feedback*, logging the outcome of a single model t_i selected by historical policies. As a result, counterfactual outcomes for unobserved LLMs are missing, making it necessary to estimate them while correcting for treatment bias. We address this challenge of estimating $\hat{Y}_X(\cdot)$ using causal inference techniques, as detailed in Section 2.3.

With an accurate approximation $\hat{Y}_X(\cdot)$, the empirical regret can be approximated as

$$Regret(f) \approx \frac{1}{n}\sum_{i=1}^{n}\left(\hat{Y}_{x_i}(t_i^*) - \hat{Y}_{x_i}(f(x_i))\right), \tag{2}$$

where $t_i^* := \arg\max_{t\in\mathcal{T}} \hat{Y}_{x_i}(t)$ is the estimated optimal decision for query x_i.

A key challenge with the objective in Equation (2) is its dependence on the discrete routing decision $f(x_i)$, which makes the regret non-differentiable. To address this, we introduce two surrogate loss functions that serve as differentiable approximations.

Surrogate Loss 1: Classification-Based Upper Bound Our first approach is to derive a tractable upper bound on the regret and directly minimize it. To do so, we define the following notion of Lipschitz continuity for utility functions over the probability simplex.

Definition 1. *Let $\hat{Y}_x : \mathcal{T} \to \mathbb{R}$ be an estimated utility function assigning a scalar utility to each model $t \in \mathcal{T}$ for a given input x. We extend $\hat{Y}_x$ to the probability simplex $\Delta^{|\mathcal{T}|}$ by defining*

$$\hat{Y}_x(p) := \sum_{t\in\mathcal{T}} p(t)\hat{Y}_x(t), \tag{3}$$

for any $p \in \Delta^{|\mathcal{T}|}$. We say that $\hat{Y}_x$ is L-Lipschitz over the simplex with respect to the ℓ_1 norm if for all $p, q \in \Delta^{|\mathcal{T}|}$,

$$|\hat{Y}_x(p) - \hat{Y}_x(q)| \le L \cdot \|p - q\|_1. \tag{4}$$

Here, the constant L can be taken as $L := \max_{t\in\mathcal{T}} |\hat{Y}_x(t)|$.

This condition ensures that small changes in the model distribution lead to bounded changes in expected utility. Since $\mathcal{T}$ is a finite set and $\hat{Y}_x(\cdot)$ is typically learned via bounded smooth function approximators (e.g., neural networks), it is natural to expect bounded variation in utility values across nearby treatments. Note that when p is a one-hot vector e_{t^*} and $q = f(x)$ is a probabilistic policy, this setting corresponds to our model selection problem, where we seek to minimize the regret between the optimal choice and a stochastic routing decision.

Proposition 1. Suppose the estimated utility function $\hat{Y}_x : \mathcal{T} \to \mathbb{R}$ is L-Lipschitz continuous over the probability simplex with respect to the ℓ_1 norm, as in Definition 1. Then, for a policy $f : \mathcal{X} \to \Delta^{|\mathcal{T}|}$ that outputs a distribution $f(x)$ over $\mathcal{T}$, the regret can be upper bounded by:

$$\text{Regret}(f) \le L \cdot \frac{1}{n}\sum_{i=1}^{n}\sqrt{2 \cdot \text{CE}(t_i^*, f(x_i))}, \tag{5}$$

where $t_i^* := \arg\max_{t\in\mathcal{T}} \hat{Y}_{x_i}(t)$ is the optimal treatment for input x_i, and $\text{CE}(t_i^*, f(x_i)) := -\log f(x_i)_{t_i^*}$ denotes the cross-entropy loss.

This motivates a classification-based surrogate objective: rather than modeling the full utility surface, we directly learn a policy $f : \mathcal{X} \to \mathcal{T}$ by solving a supervised learning problem, where the target label for each input x_i is the estimated optimal decision t_i^*. Optimizing a classification loss $d(t_i^*, f(x_i))$, serves as a tractable surrogate for minimizing the regret in Equation (2), as it upper bounds the regret under mild assumptions. This formulation reduces policy learning to a multiclass classification task, enabling efficient training using standard techniques.

2.2.1 Surrogate Loss 2: Softmax-Weighted Regret

The second proxy directly minimizes the regret using a differentiable softmax approximation. Specifically, we model the policy function f as a neural network with $|\mathcal{T}|$ outputs passed through a softmax layer with temperature parameter $\tau > 0$, which makes the regret surrogate in Equation (2) differentiable. The first term of the regret, $\mathbb{E}_X[Y_X(t^*)]$, is approximated as:

$$\mathbb{E}_X[Y_X(t^*)] \approx \frac{1}{n}\sum_{i=1}^{n} \hat{Y}_{x_i}(t_i^*), \quad \text{where } t_i^* := \arg\max_{t\in\mathcal{T}} \hat{Y}_{x_i}(t). \tag{6}$$

The second term, $\mathbb{E}_X[Y_X(f(X))]$, is estimated by treating the softmax output as a distribution over treatments:

$$\mathbb{E}_X[Y_X(f(X))] \approx \frac{1}{n}\sum_{i=1}^{n}\sum_{t=1}^{|\mathcal{T}|} \hat{Y}_{x_i}(t) \cdot \mathrm{softmax}(f(x_i))_t. \tag{7}$$

Combining the two, we minimize the following differentiable surrogate objective:

$$\min_f \frac{1}{n}\sum_{i=1}^{n}\left(\hat{Y}_{x_i}(t_i^*) - \sum_{t=1}^{|\mathcal{T}|} \hat{Y}_{x_i}(t) \cdot \mathrm{softmax}(f(x_i))_t\right). \tag{8}$$

After training, the learned policy prescribes for each $x \in \mathcal{X}$ the treatment $\hat{t} = \arg\max_{t\in\mathcal{T}} f(x)_t$. We now show that this objective recovers pointwise optimal treatment assignment, thus providing a consistent and differentiable approximation to the original regret minimization objective.

Proposition 2. Let $f : \mathcal{X} \to \mathbb{R}^{|\mathcal{T}|}$ be a neural network whose output is passed through a softmax layer with fixed temperature $\tau > 0$, and define $t_i^* := \arg\max_{t\in\mathcal{T}} \hat{Y}_{x_i}(t)$. Then, optimizing the softmax-weighted surrogate regret objective via gradient descent

$$\min_f \frac{1}{n}\sum_{i=1}^{n}\left(\hat{Y}_{x_i}(t_i^*) - \sum_{t=1}^{|\mathcal{T}|} \hat{Y}_{x_i}(t) \cdot \mathrm{softmax}(f(x_i))_t\right) \tag{9}$$

leads the model f to concentrate all probability mass on the optimal treatment t_i^*. That is, at convergence,

$$\mathrm{softmax}(f(x_i))_t \to \begin{cases} 1 & \text{if } t = t_i^*, \\ 0 & \text{otherwise.} \end{cases} \tag{10}$$

2.3 Estimating Counterfactual Utility via Causal Inference

In the previous sections, we assumed that we have access to $\hat{Y}_X(\cdot)$. Given the observational nature of the data, the potential utility function $Y_x(\cdot)$ is not directly observable. We follow the potential outcomes framework Rosenbaum and Rubin [1983], Rubin [1984] and assume the existence of a potential utility function $Y_x(t)$. We adopt the following assumptions, which are standard in the causal inference literature.

Assumption 1 (Stable Unit Treatment Value). The potential outcome of one sample is independent of the treatment assignments on the other samples.

Assumption 2 (Ignorability). The assigned treatments and potential outcomes are independent conditional on observed covariates, i.e. $t \perp \{Y_x(t')|t' \in \mathcal{T}\}|x$ [Hirano and Imbens, 2004].

Assumption 3 (Support). For $x \in \mathcal{X}$ such that $p(x) > 0$, we have that $p(t|x) > 0$ for each $t \in \mathcal{T}$.

In the causal inference literature, counterfactual outcomes can be estimated using various methods. Such examples include the "meta-learner" [Künzel et al., 2019], or the Inverse Propensity Weighting

(IPW) estimator [Horvitz and Thompson, 1952]. In this work, we utilize the doubly robust estimator introduced by Dudík et al. [2011], defined as:

$$\hat{Y}_x(t) := \frac{(y - \hat{r}_t(x))\mathbb{1}[\pi(x) = t]}{\hat{p}(t|x)} + \hat{r}_t(x), \quad \forall t \in \mathcal{T}, \tag{11}$$

where $\hat{r}_t : \mathcal{X} \to \mathcal{Y}$ denotes the direct outcome regression model for treatment t, $\hat{p}(t|x)$ is the estimated propensity score, and $\pi : \mathcal{X} \to \mathcal{T}$ is the logging policy observed in the dataset $\mathcal{D}$.

The doubly robust estimator combines an outcome model $\hat{r}_t(x)$ and a propensity model $\hat{p}(t|x)$, yielding consistent estimates if either is correctly specified [Dudík et al., 2011]. It offers a favorable bias-variance trade-off: the propensity model corrects for selection bias, while the outcome model reduces variance by leveraging structure in the data.

Remark 1. While we use the doubly robust (DR) estimator in our experiments due to its favorable bias–variance tradeoff and strong practical performance, our framework is estimator-agnostic: any valid counterfactual estimator can be used to compute utility estimates. This includes more advanced approaches that relax or mitigate these assumptions

In the experimental section, we show that ignoring the treatment bias leads to inaccurate counterfactual estimates and causes substantial degradation in routing quality, highlighting the limitations of standard supervised learning approaches that assume full feedback.

3 Routing under Heterogeneous Cost Preferences

In the previous section, we introduced a causal end-to-end framework for learning optimal routing policies from observational data, where the objective is to maximize a utility function of the form $y = a - \lambda c$, with a *fixed* $\lambda \geq 0$ representing the trade-off between accuracy and cost.

In practice, however, user preferences vary, i.e., different queries may be associated with different values of λ. From a system design perspective, training and maintaining a separate router for each possible λ is impractical. In this section, we propose a unified approach that supports routing under heterogeneous cost sensitivities. We first present a joint model architecture that conditions on both the query and the cost parameter, and then provide a theoretical analysis to justify the proposed design.

3.1 Interval-Conditioned Joint Router

We design a neural network $f : \mathcal{X} \times \mathbb{R}_{\geq 0} \to \mathbb{R}^{|\mathcal{T}|}$ that jointly takes a query $x \in \mathcal{X}$ and a cost sensitivity parameter $\lambda \in \mathbb{R}_{\geq 0}$ as input, outputs a score vector over available LLMs and the routing decision is then made via:

$$\hat{t} = \arg\max_{t \in \mathcal{T}} f(x, \lambda)_t. \tag{12}$$

We assume access to a finite, representative set of cost preferences $\Lambda := \{\lambda_1, \ldots, \lambda_m\} \subset \mathbb{R}_{\geq 0}$. For ease of notation we assume that $\lambda_1 < \lambda_2 < \cdots < \lambda_m$. In practice, these values may correspond to discrete service quality tiers (e.g., basic, standard, premium) that reflect users' varying willingness to trade off cost for performance. For each $\lambda \in \Lambda$, we partition the training data by cost preference and estimate λ-specific utility $\hat{Y}^{\lambda}_{x_i}(t)$, which forms the basis of our joint interval-conditioned architecture.

Training Procedure.

1. For each $\lambda \in \Lambda$, we first train an individual router $f_\lambda : \mathcal{X} \to \mathbb{R}^{|\mathcal{T}|}$ using the methods introduced in Section 2.2.
2. For each interval $[\lambda_j, \lambda_{j+1}]$ $j \in [1, \ldots, m-1]$, we initialize a joint network $f(x, \lambda) : \mathcal{X} \times \mathbb{R}_{\geq 0} \to \mathbb{R}^{|\mathcal{T}|}$ that uses as input both x, $\lambda \in (\lambda_j, \lambda_{j+1})$.
3. The shared model is fine-tuned to minimize regret over the interval:

$$\min_f \frac{1}{2n} \sum_{\lambda \in \{\lambda_j, \lambda_{j+1}\}} \sum_{i=1}^{n} \text{Regret}\left(f(x_i, \lambda)\right), \tag{13}$$

where regret is computed using the doubly robust estimator as described earlier, under the corresponding λ-specific utility $\hat{Y}^{\lambda}_{x_i}(t)$.

Deployment Strategy. At inference time, given a user-specified cost sensitivity parameter $\lambda \in \mathbb{R}_{\geq 0}$:

- If $\lambda \in \Lambda$, we use the individual model $f_\lambda(x)$.
- If $\lambda \notin \Lambda$, we identify the closest neighbors $\underline{\lambda}, \overline{\lambda} \in \Lambda$ such that $\underline{\lambda} < \lambda < \overline{\lambda}$, and we use the corresponding joint network $f(x, \lambda)$ trained to generalize across the preference in $(\underline{\lambda}, \overline{\lambda})$.

3.2 Model Architecture

A key component of the heterogeneous cost preference routing setup introduced in Section 3.1 is the interval-conditioned joint model $f(x, \lambda)$, that for each interval $[\lambda_j, \lambda_{j+1}]$, interpolates between the two corresponding individual models f_{λ_j} and $f_{\lambda_{j+1}}$. Specifically, the architecture is designed to exploit the *affine structure* of the utility function with respect to λ, namely $y = a - \lambda c$. This motivates a lightweight parameterization that uses only the two endpoints of the interval $[\lambda_j, \lambda_{j+1}]$ rather than all m pre-trained models. Concretely, the joint model that we propose is defined as:

$$f(x, \lambda) = \texttt{Linear}\left([f_{\lambda_j}(x),\ f_{\lambda_{j+1}}(x)] + \ g(\lambda)\right), \tag{14}$$

where $[\cdot, \cdot]$ denotes concatenation, and $g(\lambda) := \texttt{Activation}(\texttt{Linear}(\lambda))$ is a learnable representation of the cost sensitivity parameter.

This architecture enables smooth interpolation between f_{λ_j} and $f_{\lambda_{j+1}}$ within $[\lambda_j, \lambda_{j+1}]$, allowing the router to adapt to intermediate values of λ without requiring an individual model for each one. By conditioning only on the two bounding models, this design achieves computational efficiency and strong generalization across cost preferences. The proposed architecture is illustrated in Figure 1.

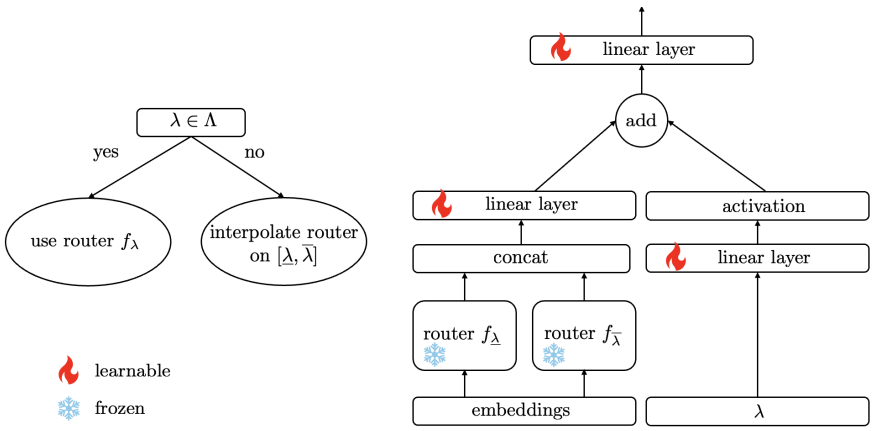

Figure 1: Overview of the proposed interval-conditioned joint router framework. **Left:** Decision logic for handling a given cost sensitivity parameter λ. **Right:** Joint router architecture.

3.3 Theoretical Guarantees

We now present theoretical guarantees that justify the structure and training strategy of joint cost-preference router. These guarantees leverage the affine nature of the utility function, which is linear in the cost parameter λ for fixed accuracy a and cost c, enabling exact interpolation for specific cost sensitivity across fixed intervals.

Proposition 3 (Piecewise Constant Optimal Policy). Fix a query $x \in \mathcal{X}$ and assume the estimated utility is affine in λ, i.e., $\hat{Y}_x^\lambda(t) = a_x(t) - \lambda \cdot c_x(t)$ for all $t \in \mathcal{T}$. Then the optimal treatment

$$t^*(\lambda) := \arg\max_{t \in \mathcal{T}} \hat{Y}_x^\lambda(t)$$

is piecewise constant in λ. That is, the cost parameter $\mathbb{R}_{\geq 0}$ can be partitioned into intervals over which the optimal treatment remains fixed.

Proposition 4 (Affine Closure of Utility Function). Let $\lambda_j < \lambda_{j+1}$ be two adjacent cost values and let $\lambda \in [\lambda_j, \lambda_{j+1}]$. Suppose the utility function is affine in λ, i.e., $\hat{Y}_x^\lambda(t) = a_x(t) - \lambda \cdot c_x(t)$. Then

for all $t \in \mathcal{T}$, the utility at λ is a convex combination of utilities at the endpoints:

$$\hat{Y}_x^{\lambda}(t) = \alpha \cdot \hat{Y}_x^{\lambda_j}(t) + (1-\alpha) \cdot \hat{Y}_x^{\lambda_{j+1}}(t), \quad \text{where } \alpha := \frac{\lambda_{j+1} - \lambda}{\lambda_{j+1} - \lambda_j}.$$

Corollary 1 (Sufficiency of Two Models per Interval). Under the affine assumption, the utility $\hat{Y}_x^{\lambda}(t)$ for any $\lambda \in [\lambda_j, \lambda_{j+1}]$ can be exactly reconstructed using only the endpoints $\hat{Y}_x^{\lambda_j}(t)$ and $\hat{Y}_x^{\lambda_{j+1}}(t)$. Thus, it is sufficient to use only the two corresponding models f_{λ_j} and $f_{\lambda_{j+1}}$ for interpolation within the interval.

Proposition 5 (Expressivity of Additive Two-Model joint Architecture). Let $\lambda \in [\lambda_j, \lambda_{j+1}]$, and suppose that for each $t \in \mathcal{T}$ the utility satisfies $\hat{Y}_x^{\lambda}(t) = a_x(t) - \lambda \cdot c_x(t)$. Then the optimal treatment $t^*(\lambda) := \arg\max_t \hat{Y}_x^{\lambda}(t)$ can be exactly represented by a softmax policy over a function of the form:

$$f(x, \lambda) = \texttt{Linear}\left([f_{\lambda_j}(x), f_{\lambda_{j+1}}(x)] + g(\lambda)\right),$$

where $g(\lambda)$ is any differentiable embedding of λ, and $f_{\lambda_j}, f_{\lambda_{j+1}}$ are accurate predictors trained at endpoints λ_j and λ_{j+1}.

Implications for Architecture and Training. The theoretical results above provide strong justification for both the proposed model architecture and the associated training procedure. Proposition 3 shows that the optimal treatment changes only across a small number of cost sensitivities, supporting our interval-conditioned strategy for routing. Proposition 4 guarantees that the utility for any intermediate cost preference can be exactly recovered through convex interpolation of endpoint models. Finally, Proposition 5 establishes that our joint model is expressive enough to capture the optimal policy within each interval. Our method provides a principled approach for learning a joint routing model that accommodates heterogeneous cost preferences from observational data.

4 Experiments

4.1 Datasets

We evaluate our methods on two publicly available benchmarks for LLM routing: **RouterBench** [Hu et al., 2024] and **SPROUT** [Somerstep et al., 2025].

RouterBench is a standardized benchmark comprising 35,712 prompt–response pairs from 11 language models. The prompts are drawn from eight evaluation suites spanning reasoning, factual recall, dialogue, mathematics, and code generation. Each prompt is annotated with model accuracy and execution cost, enabling supervised training and evaluation of routing policies.

SPROUT is a larger and more diverse benchmark focused on cost-aware routing, consisting of 44,241 prompts and responses from 13 state-of-the-art LLMs. The prompts cover six challenging tasks: GPQA [Rein et al., 2024], MuSR [Sprague et al., 2023], MMLU-Pro [Wang et al., 2024], MATH [Hendrycks et al., 2021], OpenHermes [Teknium, 2023], and RAGBench [Friel et al., 2024]. SPROUT includes a predefined split: 80% for training, with the remaining 20% evenly divided between validation and test sets.

4.2 Embeddings and Model Architecture

To encode input queries into vector representations x, we generate embeddings using two compact, publicly available language models: `BERT-base-uncased` (768 dimensions) and `Llama-3.2-1B` (2048 dimensions). Each input is passed once through the model, and the final hidden states are mean-pooled to obtain a fixed-length embeddings. These models were selected for their efficiency and suitability for real-time routing.

The embeddings are processed by a two-layer fully connected network with GELU activations and 200 hidden units per layer. The model is trained with the Adam optimizer (learning rate 1×10^{-4}) for up to 10,000 epochs, using early stopping with a patience of 100. A softmax output with temperature $\tau = 100$ is used to control the sharpness of the output probabilities. This architecture is used consistently across all benchmarked methods for fair comparison. For the doubly robust estimator, the same network models the direct outcomes $\hat{r}_t(x)$, while the propensity scores $\hat{p}(t \mid x)$ are estimated

using XGBoost. To reduce variance from extreme inverse propensity weights, we apply clipping at the 5th and 95th percentiles. The only architectural modification is for the interval-based model, where the softmax temperature is increased to $\tau = 1000$ to enable smoother interpolation across λ. Hyperparameters are summarized in Appendix D.

4.3 Methods

We evaluate our proposed routing strategies against a range of baselines from the causal machine learning and LLM routing literature. Since both SPROUT and RouterBench provide full-feedback datasets (i.e., responses from all models), we simulate observational data by sampling a single model per prompt. Specifically, for each prompt, we sample a model $t \in \mathcal{T}$ with probability proportional to its accuracy $\mathbb{P}[t = \tau] = \frac{e^{a_\tau}}{\sum_{\tau' \in \mathcal{T}} e^{a_{\tau'}}}$, where a_τ is the accuracy of model τ on that prompt.

As an optimistic oracle, we include a **Full-Feedback** model that learns a model $f : \mathcal{X} \to \mathbb{R}^{|\mathcal{T}|}$ using the complete outcome vector for each query and optimizes a standard multi-class classification loss. We benchmark against a common decoupled routing strategy denoted **Baseline**, which independently estimates model accuracy $a_x(t)$ and cost $c_x(t)$, without accounting for selection bias. This reflects the approach taken in prior predictive routing methods such as CARROT [Somerstep et al., 2025], which has demonstrated superior performance over alternatives like RouteLLM [Ong et al., 2024] and RoRF [Jain et al., 2023]. To adjust for treatment assignment bias, we consider a **Regress-and-Compare (R&C)** method, which fits outcome models $\hat{Y}_x(t)$ for each treatment t, and selects the action $\hat{t} = \arg\max_t \hat{Y}_x(t)$. Building on this, we implement a **Causal-CARROT** variant by adapting both the parametric and kNN instantiations of CARROT to the R&C framework. We additionally include **CF-Regression**, which models $f : \mathcal{X} \to \mathbb{R}^{|\mathcal{T}|}$ and is trained to minimize MSE against the counterfactual utility function from the doubly robust estimator: $\min_f \sum_{i=1}^{n} \sum_{t=1}^{|\mathcal{T}|} (\hat{Y}_{x_i}(t) - f(x_i)_t)^2$. Decisions are made by selecting the treatment with the highest predicted value, i.e., $\hat{t} = \arg\max_t f(x)_t$.

Finally, we evaluate our regret-minimization methods. **RM-Classification** formulates the task as multi-class prediction over optimal treatments, serving as a classification-based upper bound. **RM-Softmax** directly minimizes a softmax-weighted regret surrogate. In the heterogeneous preference setting, we also assess **RM-Interval** (Section 3), which generalizes across a continuum of cost sensitivities by interpolating between models trained at discrete λ values.

Table 1: Comparison of routing methods by causal reasoning and end-to-end training.

Method	Causal	End-to-End
Full-Feedback	✓	✓
Baseline	✗	✗
R&C	✓	✗
Causal-CARROT (kNN & Embed)	✓	✗
CF-Regression	✓	✗
RM-Classification (Ours)	✓	✓
RM-Softmax (Ours)	✓	✓
RM-Interval (Ours)	✓	✓

4.4 Evaluations

We evaluate our methods in two settings. In the λ-specific setting, each cost sensitivity value $\lambda \in \{0, 100, \ldots, 1000\}$ defines a separate routing task, with models trained and evaluated independently. In the heterogeneous preference setting, **RM-Interval** is trained on a subset $\{0, 200, \ldots, 1000\}$ and tested on held-out values $\{100, 300, 500, 700, 900\}$ to assess generalization across cost sensitivities.

We report the average utility across 10 independent trials for each routing method for SPROUT dataset and BERT embeddings in Table 2, where each trial involves randomly sampling observational data and retraining all models. In Figure 2, we also visualize the corresponding accuracy–cost curve associated with each method. Additional plots for the rest of the datasets and embeddings, along with detailed router performance are provided in Appendix B.

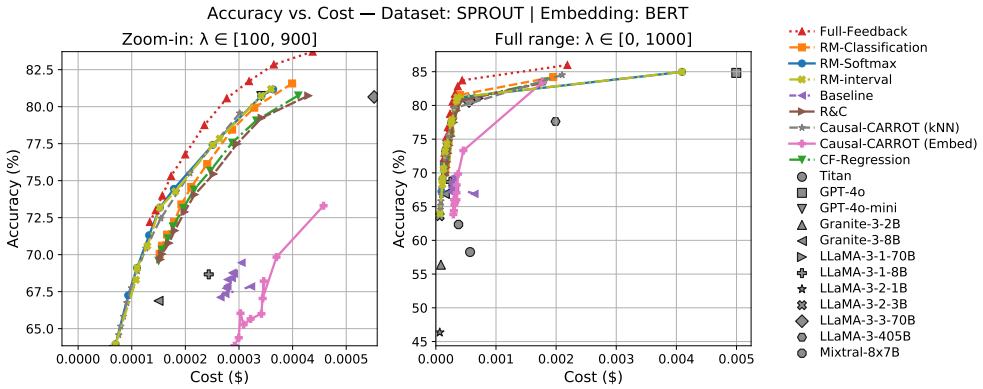

Figure 2: Accuracy–cost trade-off curve for SPROUT with BERT embeddings.

Table 2: Utility on **SPROUT** with `BERT` embeddings. *Full Feedback* serves as an oracle upper bound. The best-performing method for each column is highlighted in bold.

Method	$\lambda = 0$	$\lambda = 100$	$\lambda = 200$	$\lambda = 300$	$\lambda = 400$	$\lambda = 500$
Full-Feedback	*85.99*$_{\pm 0.17}$	*79.34*$_{\pm 0.13}$	*75.55*$_{\pm 0.29}$	*72.16*$_{\pm 0.24}$	*69.47*$_{\pm 0.23}$	*66.97*$_{\pm 0.37}$
Baseline	66.88$_{\pm 1.55}$	64.62$_{\pm 0.66}$	61.85$_{\pm 0.78}$	59.15$_{\pm 0.84}$	56.73$_{\pm 0.67}$	54.08$_{\pm 0.57}$
R&C	83.34$_{\pm 0.32}$	76.45$_{\pm 0.53}$	72.40$_{\pm 0.56}$	68.59$_{\pm 0.69}$	65.34$_{\pm 0.66}$	63.19$_{\pm 0.83}$
Causal-CARROT (kNN)	84.52$_{\pm 0.35}$	76.55$_{\pm 0.37}$	71.34$_{\pm 0.23}$	67.82$_{\pm 0.43}$	65.38$_{\pm 0.40}$	63.38$_{\pm 0.43}$
Causal-CARROT (EmbedNet)	83.46$_{\pm 0.39}$	68.73$_{\pm 0.74}$	62.44$_{\pm 2.40}$	56.72$_{\pm 1.68}$	54.37$_{\pm 2.12}$	48.91$_{\pm 2.41}$
CF-Regression	83.54$_{\pm 0.25}$	76.65$_{\pm 0.46}$	72.42$_{\pm 0.53}$	68.95$_{\pm 0.60}$	65.81$_{\pm 0.67}$	63.58$_{\pm 0.67}$
RM-Classification	84.20$_{\pm 0.24}$	**77.58**$_{\pm 0.34}$	73.36$_{\pm 0.49}$	69.84$_{\pm 0.48}$	66.49$_{\pm 0.63}$	64.03$_{\pm 1.02}$
RM-Softmax	**84.97**$_{\pm 0.39}$	77.53$_{\pm 0.81}$	**73.89**$_{\pm 0.00}$	**70.47**$_{\pm 0.00}$	**67.38**$_{\pm 0.47}$	**65.51**$_{\pm 0.60}$
RM-Interval	**84.97**$_{\pm 0.39}$	77.60$_{\pm 0.62}$	**73.89**$_{\pm 0.00}$	69.92$_{\pm 0.57}$	**67.38**$_{\pm 0.47}$	65.20$_{\pm 0.67}$
Method	$\lambda = 600$	$\lambda = 700$	$\lambda = 800$	$\lambda = 900$	$\lambda = 1000$	
Full-Feedback	*64.79*$_{\pm 0.31}$	*63.18*$_{\pm 0.21}$	*61.43*$_{\pm 0.27}$	*59.98*$_{\pm 0.30}$	*58.83*$_{\pm 0.24}$	
Baseline	51.17$_{\pm 0.54}$	48.79$_{\pm 0.67}$	45.74$_{\pm 1.21}$	42.58$_{\pm 1.36}$	38.97$_{\pm 2.65}$	
R&C	60.98$_{\pm 0.81}$	59.01$_{\pm 0.86}$	57.21$_{\pm 0.99}$	55.97$_{\pm 1.11}$	54.33$_{\pm 1.37}$	
Causal-CARROT (kNN)	61.77$_{\pm 0.43}$	60.39$_{\pm 0.38}$	59.07$_{\pm 0.37}$	57.99$_{\pm 0.35}$	56.99$_{\pm 0.34}$	
Causal-CARROT (EmbedNet)	46.35$_{\pm 1.53}$	43.66$_{\pm 3.04}$	41.81$_{\pm 2.34}$	37.42$_{\pm 5.86}$	34.69$_{\pm 4.44}$	
CF-Regression	61.37$_{\pm 0.64}$	59.52$_{\pm 0.54}$	57.72$_{\pm 0.72}$	56.31$_{\pm 0.66}$	54.56$_{\pm 0.71}$	
RM-Classification	61.84$_{\pm 1.13}$	59.68$_{\pm 1.48}$	58.09$_{\pm 1.22}$	56.52$_{\pm 1.63}$	54.85$_{\pm 1.76}$	
RM-Softmax	**64.03**$_{\pm 0.39}$	**62.04**$_{\pm 1.17}$	**60.32**$_{\pm 1.31}$	**58.85**$_{\pm 1.13}$	**56.95**$_{\pm 0.55}$	
RM-Interval	**64.03**$_{\pm 0.39}$	61.54$_{\pm 1.31}$	**60.32**$_{\pm 1.31}$	58.58$_{\pm 1.15}$	**56.95**$_{\pm 0.55}$	

Our RM-based approaches consistently deliver the strongest performance overall, with **RM-Softmax** and **RM-Interval** standing out for both high utility and low variance. Notably, **RM-Interval** generalizes remarkably well to unseen budget levels (i.e., odd λ values), many times even outperforming models trained specifically on those points. These results underscore the effectiveness of our regret-minimization framework in both fixed and variable cost settings.

The standard **Baseline** method, which reflects the common decoupled approach used in prior work and ignores treatment selection bias, performs the worst across most values of λ, underscoring the importance of accounting for treatment bias in observational data. Notably, the simple **R&C** method and the **Causal-CARROT** variants (which incorporate causal corrections) achieve substantial improvements over it, validating our claim that bias-aware routing significantly improves performance.

The performance gap between **CF-Regression** and our RM-based methods demonstrates the benefit of an integrated end-to-end approach. Whereas **CF-Regression** focuses on approximating the counterfactual utility function and then selecting the best model based on predicted outcomes, our methods directly minimize regret, leading to superior and more stable results. Comparing our two surrogate formulations, **RM-Softmax** generally outperforms **RM-Classification** in both utility and variance, indicating the advantage of optimizing a differentiable surrogate objective. Finally, among the **Causal-CARROT** variants, the kNN version consistently outperforms the EmbedNet variant, suggesting that non-parametric estimators may offer greater robustness in this setting.

5 Conclusion

We propose a causal end-to-end framework for routing queries to LLMs under observational data. Our approach introduces a regret-minimizing objective grounded in counterfactual estimation, enabling principled policy learning that accounts for treatment selection bias without requiring full-feedback data. Unlike prior approaches that rely on decoupled prediction of accuracy and cost, where errors can compound, our method directly optimizes the decision objective. To support heterogeneous user preferences, we develop an interval-conditioned routing architecture that generalizes across a continuum of cost-sensitivity parameters. Theoretical analysis provides guarantees on interpolation sufficiency and regret bounds, while empirical evaluations on public routing benchmarks demonstrate that our methods consistently outperform strong baselines, including recent routing algorithms, across multiple embedding models. Future work includes extending the framework to accommodate additional user-defined metrics or hard constraints that cannot be readily incorporated as soft penalties in the objective. Another promising direction is to explore online or adaptive routing in dynamic environments, as well as extending causal regret minimization to multi-turn settings.

References

Lars Buitinck, Gilles Louppe, Mathieu Blondel, Fabian Pedregosa, Andreas Mueller, Olivier Grisel, Vlad Niculae, Peter Prettenhofer, Alexandre Gramfort, Jaques Grobler, Robert Layton, Jake VanderPlas, Arnaud Joly, Brian Holt, and Gaël Varoquaux. API design for machine learning software: experiences from the scikit-learn project. *CoRR*, abs/1309.0238, 2013. URL `http://arxiv.org/abs/1309.0238`.

Lingjiao Chen, Matei Zaharia, and James Zou. Frugalgpt: How to use large language models while reducing cost and improving performance. *arXiv preprint arXiv:2305.05176*, 2023.

Dheeru Dua and Casey Graff. UCI Machine Learning Repository, 2017. URL `http://archive.ics.uci.edu/ml`.

Miroslav Dudík, John Langford, and Lihong Li. Doubly robust policy evaluation and learning. *arXiv preprint arXiv:1103.4601*, 2011.

Carlos Fernández-Loría and Foster Provost. Causal classification: Treatment effect estimation vs. outcome prediction. *Journal of Machine Learning Research*, 23(59):1–35, 2022.

Robert Friel, Masha Belyi, and Atindriyo Sanyal. Ragbench: Explainable benchmark for retrieval-augmented generation systems, 2025. *URL https://arxiv. org/abs/2407.11005*, 2024.

Gurobi Optimization, LLC. Gurobi Optimizer Reference Manual, 2023. URL `https://www.gurobi.com`.

Dan Hendrycks, Collin Burns, Saurav Kadavath, Akul Arora, Steven Basart, Eric Tang, Dawn Song, and Jacob Steinhardt. Measuring mathematical problem solving with the math dataset. *arXiv preprint arXiv:2103.03874*, 2021.

Keisuke Hirano and Guido W Imbens. The propensity score with continuous treatments. *Applied Bayesian modeling and causal inference from incomplete-data perspectives*, 226164:73–84, 2004.

Daniel G Horvitz and Donovan J Thompson. A generalization of sampling without replacement from a finite universe. *Journal of the American statistical Association*, 47(260):663–685, 1952.

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

Keke Huang, Yimin Shi, Dujian Ding, Yifei Li, Yang Fei, Laks Lakshmanan, and Xiaokui Xiao. Thriftllm: On cost-effective selection of large language models for classification queries. *arXiv preprint arXiv:2501.04901*, 2025.

D. Jain, T.-Y. Tung, and T. H. Kofman. RoRF: Routing on random forests. `https://www.notdiamond.ai/blog/rorf`, 2023. Accessed: 2025-01-02.

Sören R Künzel, Jasjeet S Sekhon, Peter J Bickel, and Bin Yu. Metalearners for estimating heterogeneous treatment effects using machine learning. *Proceedings of the national academy of sciences*, 116(10):4156–4165, 2019.

Isaac Ong, Amjad Almahairi, Vincent Wu, Wei-Lin Chiang, Tianhao Wu, Joseph E Gonzalez, M Waleed Kadous, and Ion Stoica. Routellm: Learning to route llms from preference data. In *The Thirteenth International Conference on Learning Representations*, 2024.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. Pytorch: An imperative style, high-performance deep learning library. In *Advances in Neural Information Processing Systems 32*, pages 8024–8035. Curran Associates, Inc., 2019. URL `http://papers.neurips.cc/paper/9015-pytorch-an-imperative-style-high-performance-deep-learning-library.pdf`.

David Rein, Betty Li Hou, Asa Cooper Stickland, Jackson Petty, Richard Yuanzhe Pang, Julien Dirani, Julian Michael, and Samuel R Bowman. Gpqa: A graduate-level google-proof q&a benchmark. In *First Conference on Language Modeling*, 2024.

Paul R Rosenbaum and Donald B Rubin. The central role of the propensity score in observational studies for causal effects. *Biometrika*, 70(1):41–55, 1983.

Donald B Rubin. Bayesianly justifiable and relevant frequency calculations for the applied statistician. *The Annals of Statistics*, pages 1151–1172, 1984.

Shivanshu Shekhar, Tanishq Dubey, Koyel Mukherjee, Apoorv Saxena, Atharv Tyagi, and Nishanth Kotla. Towards optimizing the costs of llm usage. *arXiv preprint arXiv:2402.01742*, 2024.

Tal Shnitzer, Anthony Ou, Mírian Silva, Kate Soule, Yuekai Sun, Justin Solomon, Neil Thompson, and Mikhail Yurochkin. Large language model routing with benchmark datasets. *arXiv preprint arXiv:2309.15789*, 2023.

Aditi Singh, Nirmal Prakashbhai Patel, Abul Ehtesham, Saket Kumar, and Tala Talaei Khoei. A survey of sustainability in large language models: Applications, economics, and challenges. In *2025 IEEE 15th Annual Computing and Communication Workshop and Conference (CCWC)*, pages 00008–00014. IEEE, 2025.

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

Zayne Sprague, Xi Ye, Kaj Bostrom, Swarat Chaudhuri, and Greg Durrett. Musr: Testing the limits of chain-of-thought with multistep soft reasoning. *arXiv preprint arXiv:2310.16049*, 2023.

Adith Swaminathan and Thorsten Joachims. Batch learning from logged bandit feedback through counterfactual risk minimization. *The Journal of Machine Learning Research*, 16(1):1731–1755, 2015.

Teknium. Openhermes 2.5. `https://huggingface.co/datasets/teknium/OpenHermes-2.5`, 2023. Accessed: 2025-01-30.

Guido Van Rossum and Fred L. Drake. *Python 3 Reference Manual*. CreateSpace, Scotts Valley, CA, 2009. ISBN 1441412697.

Hongyi Wang, Felipe Maia Polo, Yuekai Sun, Souvik Kundu, Eric Xing, and Mikhail Yurochkin. Fusing models with complementary expertise. *arXiv preprint arXiv:2310.01542*, 2023.

Yubo Wang, Xueguang Ma, Ge Zhang, Yuansheng Ni, Abhranil Chandra, Shiguang Guo, Weiming Ren, Aaran Arulraj, Xuan He, Ziyan Jiang, et al. Mmlu-pro: A more robust and challenging multi-task language understanding benchmark. In *The Thirty-eight Conference on Neural Information Processing Systems Datasets and Benchmarks Track*, 2024.

Hao Zou, Bo Li, Jiangang Han, Shuiping Chen, Xuetao Ding, and Peng Cui. Counterfactual prediction for outcome-oriented treatments. In Kamalika Chaudhuri, Stefanie Jegelka, Le Song, Csaba Szepesvari, Gang Niu, and Sivan Sabato, editors, *Proceedings of the 39th International Conference on Machine Learning*, volume 162 of *Proceedings of Machine Learning Research*, pages 27693–27706. PMLR, 17–23 Jul 2022. URL https://proceedings.mlr.press/v162/zou22a.html.

A Proofs of Propositions

Proposition 1. Suppose the estimated utility function $\hat{Y}_x : \mathcal{T} \to \mathbb{R}$ is L-Lipschitz continuous over the probability simplex with respect to the ℓ_1 norm, as in Definition 1. Then, for a policy $f : \mathcal{X} \to \Delta^{|\mathcal{T}|}$ that outputs a distribution $f(x)$ over $\mathcal{T}$, the regret can be upper bounded by:

$$\text{Regret}(f) \leq L \cdot \frac{1}{n} \sum_{i=1}^{n} \sqrt{2 \cdot \text{CE}(t_i^*, f(x_i))}, \tag{15}$$

where $t_i^* := \arg\max_{t \in \mathcal{T}} \hat{Y}_{x_i}(t)$ is the optimal treatment for input x_i, and $\text{CE}(t_i^*, f(x_i)) := -\log f(x_i)_{t_i^*}$ denotes the cross-entropy loss.

Proof. Let $e_{t_i^*} \in \Delta^{|\mathcal{T}|}$ denote the one-hot distribution over the optimal treatment t_i^*. By the definition of regret:

$$\text{Regret}(f) = \frac{1}{n} \sum_{i=1}^{n} \left[\hat{Y}_{x_i}(e_{t_i^*}) - \hat{Y}_{x_i}(f(x_i)) \right]. \tag{16}$$

Using the L-Lipschitz continuity of $\hat{Y}_{x_i}(\cdot)$ under the ℓ_1 norm:

$$\left| \hat{Y}_{x_i}(e_{t_i^*}) - \hat{Y}_{x_i}(f(x_i)) \right| \leq L \cdot \| e_{t_i^*} - f(x_i) \|_1. \tag{17}$$

Applying Pinsker's inequality:

$$\| e_{t_i^*} - f(x_i) \|_1 \leq \sqrt{2 \cdot \text{KL}(e_{t_i^*} \parallel f(x_i))} = \sqrt{2 \cdot \text{CE}(t_i^*, f(x_i))}, \tag{18}$$

where the equality follows because KL divergence from a one-hot distribution to a probability vector reduces to cross-entropy. Combining the above:

$$\text{Regret}(f) \leq L \cdot \frac{1}{n} \sum_{i=1}^{n} \sqrt{2 \cdot \text{CE}(t_i^*, f(x_i))}, \tag{19}$$

which completes the proof. □

Proposition 2. Let $f : \mathcal{X} \to \mathbb{R}^{|\mathcal{T}|}$ be a neural network whose output is passed through a softmax layer with fixed temperature $\tau > 0$, and define $t_i^* := \arg\max_{t \in \mathcal{T}} \hat{Y}_{x_i}(t)$. Then, optimizing the following objective using gradient descent

$$\min_f \frac{1}{n} \sum_{i=1}^{n} \left(\hat{Y}_{x_i}(t_i^*) - \sum_{t=1}^{|\mathcal{T}|} \hat{Y}_{x_i}(t) \cdot \text{softmax}(f(x_i))_t \right) \tag{20}$$

leads the model f to place all probability mass on the optimal treatment t_i^*. That is, at convergence,

$$\text{softmax}(f(x_i))_t \to \begin{cases} 1 & \text{if } t = t_i^*, \\ 0 & \text{otherwise.} \end{cases} \tag{21}$$

Proof. Let $\hat{Y}_{x_i} \in \mathbb{R}^{|\mathcal{T}|}$ denote the vector of estimated potential outcomes for input x_i, and let $f(x_i) \in \mathbb{R}^{|\mathcal{T}|}$ be the output of the neural network before the softmax layer. The objective for a single instance x_i can be written as minimizing the regret surrogate:

$$\hat{Y}_{x_i}(t_i^*) - \sum_{t=1}^{|\mathcal{T}|} \hat{Y}_{x_i}(t) \cdot \text{softmax}(f(x_i))_t. \tag{22}$$

This is equivalent to maximizing the inner product:

$$\langle \hat{Y}_{x_i}, \text{softmax}(f(x_i)) \rangle. \tag{23}$$

Let us denote $p := \mathrm{softmax}(f(x_i)) \in \Delta^{|\mathcal{T}|-1}$, the probability simplex. We now show that the inner product $\langle \hat{Y}_{x_i}, p \rangle$ increases at each gradient step. Since $p = \mathrm{softmax}(f(x_i))$, we can compute the gradient of the objective with respect to $f(x_i)$ as:

$$\nabla_{f(x_i)} \langle \hat{Y}_{x_i}, \mathrm{softmax}(f(x_i)) \rangle = J_{\mathrm{softmax}}(f(x_i))^\top \hat{Y}_{x_i}, \tag{24}$$

where $J_{\mathrm{softmax}}(f(x_i))$ is the Jacobian of the softmax function, given by:

$$J_{\mathrm{softmax}}(f(x_i))_{t,s} = \frac{\partial \,\mathrm{softmax}(f(x_i))_t}{\partial f(x_i)_s} = \mathrm{softmax}(f(x_i))_t \left(\delta_{t,s} - \mathrm{softmax}(f(x_i))_s\right). \tag{25}$$

This gradient direction corresponds to increasing the logit value of actions with higher $\hat{Y}_{x_i}(t)$ and decreasing those with lower values, pushing the softmax distribution toward the mode of $\hat{Y}_{x_i}$. In other words, the gradient ascent step increases the inner product at each iteration k:

$$\langle \hat{Y}_{x_i}, \mathrm{softmax}(f(x_i)) \rangle^{(k+1)} > \langle \hat{Y}_{x_i}, \mathrm{softmax}(f(x_i)) \rangle^{(k)}. \tag{26}$$

Since $\hat{Y}_{x_i}$ is fixed and the softmax is smooth and bounded, this sequence is monotonically increasing and converges to the maximum possible value:

$$\langle \hat{Y}_{x_i}, \mathrm{softmax}(f(x_i)) \rangle \to \max_t \hat{Y}_{x_i}(t) = \hat{Y}_{x_i}(t_i^*), \tag{27}$$

which implies:

$$\mathrm{softmax}(f(x_i))_t \to \begin{cases} 1 & \text{if } t = t_i^*, \\ 0 & \text{otherwise.} \end{cases} \tag{28}$$

Thus, the regret surrogate converges to zero:

$$\hat{Y}_{x_i}(t_i^*) - \langle \hat{Y}_{x_i}, \mathrm{softmax}(f(x_i)) \rangle \to 0, \tag{29}$$

and the learned policy selects the treatment maximizing the estimated outcome. □

Proposition 3 (Piecewise Constant Optimal Policy). Fix a query $x \in \mathcal{X}$ and assume the estimated utility function is affine in λ, i.e., $\hat{Y}_x^\lambda(t) = a_x(t) - \lambda \cdot c_x(t)$ for all $t \in \mathcal{T}$. Then the optimal treatment

$$t^*(\lambda) := \arg\max_{t \in \mathcal{T}} \hat{Y}_x^\lambda(t)$$

is piecewise constant in λ. That is, the budget space $\mathbb{R}_{\geq 0}$ can be partitioned into intervals over which the optimal treatment remains fixed.

Proof. For fixed x, each $\hat{Y}_x^\lambda(t)$ is an affine function of λ. The pointwise maximum of a finite collection of affine functions is piecewise affine, and the argmax corresponds to the highest line at each λ. Since each pair of lines can intersect at most once, the number of intervals over which a single treatment is optimal is bounded by $|\mathcal{T}| - 1$. Therefore, $t^*(\lambda)$ changes only at these intersection points and remains constant within each interval. □

Proposition 4 (Affine Closure of Utility Function). Let $\lambda_j < \lambda_{j+1}$ be two adjacent budget values and let $\lambda \in [\lambda_j, \lambda_{j+1}]$. Suppose the utility function is affine in λ:

$$\hat{Y}_x^\lambda(t) = a_x(t) - \lambda \cdot c_x(t).$$

Then for all $t \in \mathcal{T}$, the utility at λ is a convex combination of utilities at the endpoints:

$$\hat{Y}_x^\lambda(t) = \alpha \cdot \hat{Y}_x^{\lambda_j}(t) + (1 - \alpha) \cdot \hat{Y}_x^{\lambda_{j+1}}(t), \quad \text{where } \alpha := \frac{\lambda_{j+1} - \lambda}{\lambda_{j+1} - \lambda_j}.$$

Proof. We expand each term:

$$\hat{Y}_x^{\lambda_j}(t) = a_x(t) - \lambda_j \cdot c_x(t),$$
$$\hat{Y}_x^{\lambda_{j+1}}(t) = a_x(t) - \lambda_{j+1} \cdot c_x(t).$$

Then:

$$\begin{aligned}\alpha \cdot \hat{Y}_x^{\lambda_j}(t) + (1-\alpha) \cdot \hat{Y}_x^{\lambda_{j+1}}(t) &= \alpha \cdot (a_x(t) - \lambda_j c_x(t)) + (1-\alpha) \cdot (a_x(t) - \lambda_{j+1} c_x(t)) \\ &= a_x(t) - [\alpha\lambda_j + (1-\alpha)\lambda_{j+1}] \cdot c_x(t) \\ &= a_x(t) - \lambda \cdot c_x(t) = \hat{Y}_x^{\lambda}(t),\end{aligned}$$

since:

$$\alpha\lambda_j + (1-\alpha)\lambda_{j+1} = \lambda.$$

□

Corollary 1 (Sufficiency of Two Models per Interval). Under the affine assumption, the utility $\hat{Y}_x^{\lambda}(t)$ for any $\lambda \in [\lambda_j, \lambda_{j+1}]$ can be exactly reconstructed using only the endpoints $\hat{Y}_x^{\lambda_j}(t)$ and $\hat{Y}_x^{\lambda_{j+1}}(t)$. Thus, it is sufficient to use only the two corresponding models f_{λ_j} and $f_{\lambda_{j+1}}$ for interpolation within the interval.

Proof. This follows immediately from the statement of Proposition 4. □

Proposition 5 (Expressivity of Additive Two-Model joint Architecture). Let $\lambda \in [\lambda_j, \lambda_{j+1}]$, and suppose that for each $t \in \mathcal{T}$ the utility function satisfies $\hat{Y}_x^{\lambda}(t) = a_x(t) - \lambda \cdot c_x(t)$. Then the optimal treatment $t^*(\lambda) := \arg\max_t \hat{Y}_x^{\lambda}(t)$ can be exactly represented by a softmax policy over a function of the form:

$$f(x, \lambda) = \texttt{Linear}\left([f_{\lambda_j}(x), f_{\lambda_{j+1}}(x)] + g(\lambda)\right),$$

where $g(\lambda)$ is any differentiable embedding of λ, and $f_{\lambda_j}, f_{\lambda_{j+1}}$ are accurate predictors trained at endpoints λ_j and λ_{j+1}.

Proof. From Proposition 4, the utility $\hat{Y}_x^{\lambda}(t)$ is a convex combination of $\hat{Y}_x^{\lambda_j}(t)$ and $\hat{Y}_x^{\lambda_{j+1}}(t)$. If the network $f(x, \lambda)$ linearly combines the outputs of $f_{\lambda_j}(x)$ and $f_{\lambda_{j+1}}(x)$, then its scores can match $\hat{Y}_x^{\lambda}(t)$ up to a scalar transformation. Applying softmax preserves the argmax.

Including $g(\lambda)$ allows the architecture to learn any additional monotonic reweighting of the interpolation, ensuring the output scores can be shaped to approximate the true utility surface exactly. Thus, the architecture can represent the optimal policy within each interval. □

B Additional Results

In this section, we present the additional plots for the rest of the datasets as well as the exact values of utility for value λ. We begin by presenting the rest of the figures.

B.1 Additional Figures

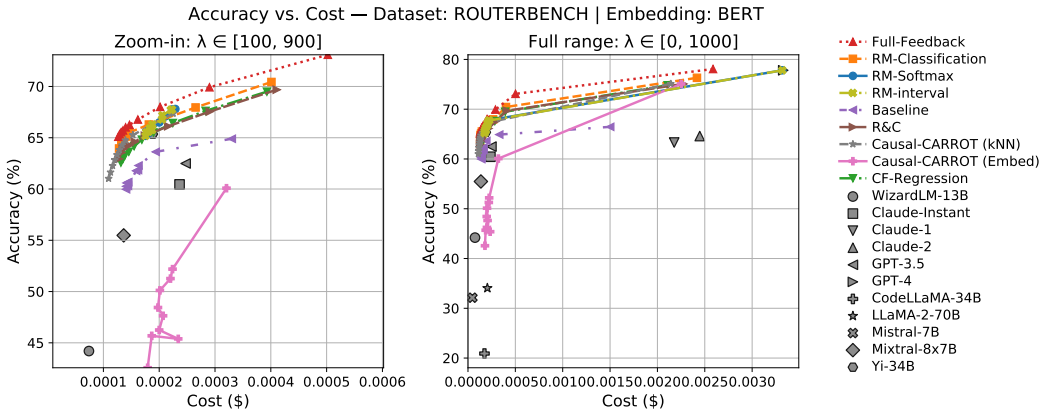

Figure 3: Accuracy–cost trade-off curve for RouterBench with BERT embeddings.

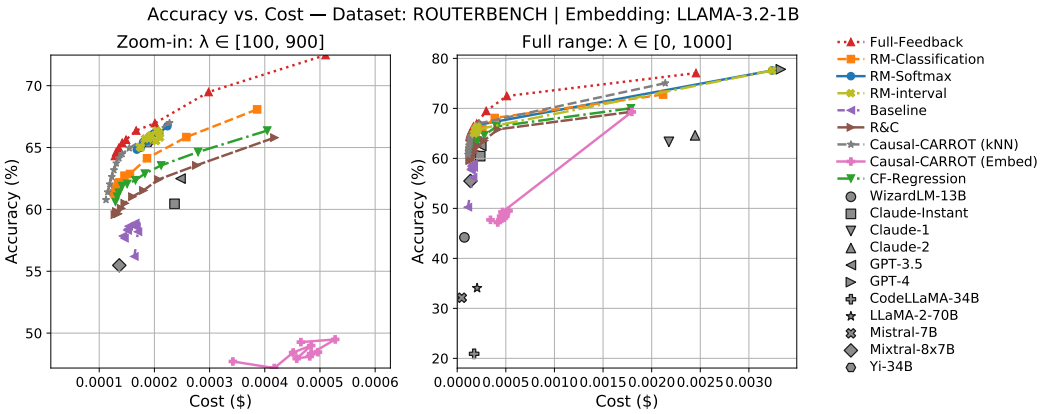

Figure 4: Accuracy–cost trade-off curve for RouterBench with LLaMa-3.2-1B embeddings.

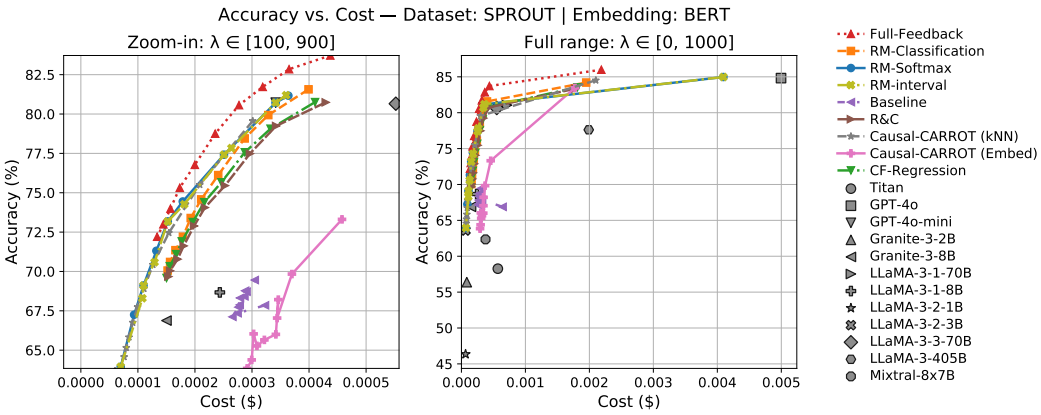

Figure 5: Accuracy–cost trade-off curve for SPROUT with BERT embeddings.

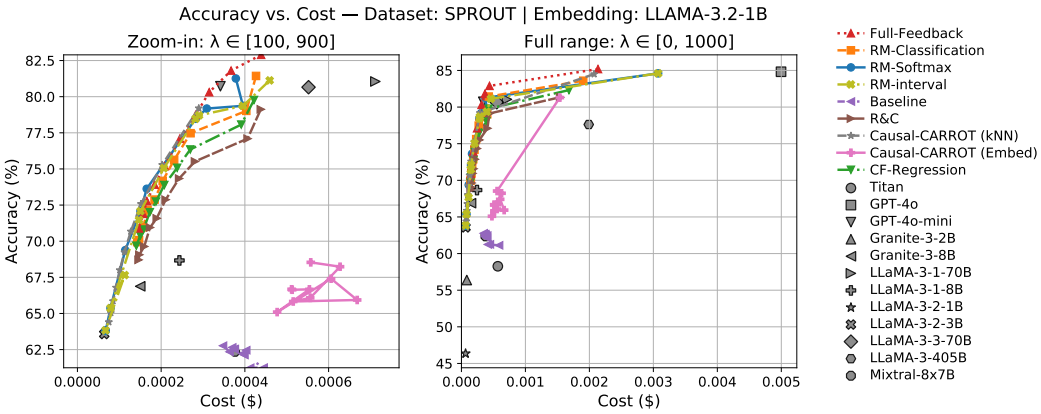

Figure 6: Accuracy–cost trade-off curve for SPROUT with LLaMa-3.2-1B embeddings.

B.2 Additional Tables

Table 3: Utility for **RouterBench** with `BERT` embeddings. *Full Feedback* serves as an oracle upper bound. The best-performing method for each column is highlighted in bold.

Method	$\lambda = 0$	$\lambda = 100$	$\lambda = 200$	$\lambda = 300$	$\lambda = 400$	$\lambda = 500$	$\lambda = 600$	$\lambda = 700$	$\lambda = 800$	$\lambda = 900$	$\lambda = 1000$
Full-Feedback	*78.07*$_{\pm 0.14}$	*68.07*$_{\pm 0.24}$	*64.12*$_{\pm 0.18}$	*61.96*$_{\pm 0.13}$	*60.31*$_{\pm 0.13}$	*58.92*$_{\pm 0.18}$	*57.62*$_{\pm 0.21}$	*56.26*$_{\pm 0.30}$	*55.00*$_{\pm 0.24}$	*53.73*$_{\pm 0.31}$	*52.42*$_{\pm 0.18}$
Baseline	66.46$_{\pm 0.66}$	61.60$_{\pm 0.64}$	59.75$_{\pm 0.46}$	57.41$_{\pm 0.49}$	55.37$_{\pm 0.51}$	53.85$_{\pm 0.96}$	51.67$_{\pm 1.04}$	50.55$_{\pm 1.07}$	48.72$_{\pm 1.28}$	47.87$_{\pm 0.76}$	46.19$_{\pm 0.46}$
R&C	75.08$_{\pm 0.49}$	65.58$_{\pm 0.28}$	61.73$_{\pm 0.53}$	59.63$_{\pm 0.53}$	58.23$_{\pm 0.58}$	56.79$_{\pm 0.63}$	55.47$_{\pm 0.64}$	54.34$_{\pm 0.46}$	53.08$_{\pm 0.58}$	51.65$_{\pm 0.59}$	50.18$_{\pm 0.53}$
Causal-CARROT (kNN)	75.12$_{\pm 0.64}$	65.12$_{\pm 0.51}$	62.21$_{\pm 0.52}$	60.56$_{\pm 0.49}$	58.94$_{\pm 0.41}$	57.39$_{\pm 0.42}$	55.86$_{\pm 0.35}$	54.41$_{\pm 0.42}$	52.95$_{\pm 0.46}$	51.47$_{\pm 0.49}$	50.08$_{\pm 0.60}$
Causal-CARROT (EmbedNet)	75.07$_{\pm 0.50}$	56.87$_{\pm 1.39}$	47.71$_{\pm 4.76}$	44.66$_{\pm 1.95}$	42.06$_{\pm 2.31}$	38.55$_{\pm 3.30}$	35.22$_{\pm 2.27}$	32.25$_{\pm 2.40}$	26.66$_{\pm 6.05}$	28.89$_{\pm 3.53}$	24.64$_{\pm 5.52}$
CF-Regression	74.81$_{\pm 0.55}$	65.56$_{\pm 0.17}$	61.95$_{\pm 0.44}$	59.71$_{\pm 0.69}$	57.91$_{\pm 0.65}$	56.36$_{\pm 0.73}$	54.87$_{\pm 0.55}$	53.44$_{\pm 0.63}$	52.19$_{\pm 0.48}$	50.78$_{\pm 0.51}$	49.43$_{\pm 0.39}$
RM-Classification	76.30$_{\pm 0.61}$	**66.43**$_{\pm 0.28}$	62.65$_{\pm 0.64}$	60.85$_{\pm 0.54}$	**59.65**$_{\pm 0.51}$	**58.07**$_{\pm 0.62}$	**56.80**$_{\pm 0.47}$	**55.29**$_{\pm 0.58}$	**54.07**$_{\pm 0.62}$	**52.45**$_{\pm 0.72}$	**51.14**$_{\pm 0.61}$
RM-Softmax	**77.82**$_{\pm 0.03}$	65.51$_{\pm 0.51}$	**63.30**$_{\pm 0.26}$	60.75$_{\pm 0.61}$	58.58$_{\pm 0.59}$	56.48$_{\pm 0.52}$	54.53$_{\pm 0.57}$	52.79$_{\pm 0.74}$	51.10$_{\pm 0.98}$	49.22$_{\pm 1.14}$	47.64$_{\pm 1.40}$
RM-Interval	**77.82**$_{\pm 0.03}$	65.51$_{\pm 0.30}$	**63.30**$_{\pm 0.26}$	**61.01**$_{\pm 0.51}$	58.58$_{\pm 0.59}$	56.92$_{\pm 0.31}$	54.53$_{\pm 0.57}$	53.11$_{\pm 0.77}$	51.10$_{\pm 0.98}$	49.60$_{\pm 1.21}$	47.64$_{\pm 1.40}$

Table 4: Utility for **RouterBench** with `Llama-3.2-1B` embeddings. The best-performing method for each column is highlighted in bold.

Method	$\lambda = 0$	$\lambda = 100$	$\lambda = 200$	$\lambda = 300$	$\lambda = 400$	$\lambda = 500$	$\lambda = 600$	$\lambda = 700$	$\lambda = 800$	$\lambda = 900$	$\lambda = 1000$
Full-Feedback	*77.06*$_{\pm 0.30}$	*67.37*$_{\pm 0.35}$	*63.52*$_{\pm 0.35}$	*60.94*$_{\pm 0.27}$	*59.70*$_{\pm 0.24}$	*58.20*$_{\pm 0.19}$	*56.88*$_{\pm 0.33}$	*55.49*$_{\pm 0.38}$	*54.03*$_{\pm 0.14}$	*52.86*$_{\pm 0.25}$	*51.47*$_{\pm 0.28}$
Baseline	50.24$_{\pm 1.28}$	54.56$_{\pm 0.96}$	54.79$_{\pm 1.11}$	53.34$_{\pm 0.63}$	52.00$_{\pm 0.57}$	50.63$_{\pm 0.68}$	49.43$_{\pm 0.66}$	47.78$_{\pm 1.22}$	46.19$_{\pm 0.61}$	44.96$_{\pm 0.66}$	43.10$_{\pm 0.84}$
R&C	69.30$_{\pm 0.63}$	61.62$_{\pm 0.70}$	58.04$_{\pm 0.44}$	56.20$_{\pm 0.52}$	54.33$_{\pm 0.58}$	53.02$_{\pm 0.62}$	51.72$_{\pm 0.57}$	50.30$_{\pm 0.50}$	48.94$_{\pm 0.48}$	48.03$_{\pm 0.56}$	46.73$_{\pm 0.56}$
Causal-CARROT (kNN)	75.04$_{\pm 0.47}$	64.72$_{\pm 0.47}$	61.86$_{\pm 0.53}$	**60.23**$_{\pm 0.54}$	**58.71**$_{\pm 0.49}$	**57.18**$_{\pm 0.48}$	**55.58**$_{\pm 0.50}$	**54.04**$_{\pm 0.49}$	**52.50**$_{\pm 0.51}$	**51.00**$_{\pm 0.59}$	**49.54**$_{\pm 0.54}$
Causal-CARROT (EmbedNet)	69.30$_{\pm 0.38}$	44.62$_{\pm 1.38}$	38.94$_{\pm 2.63}$	33.82$_{\pm 2.07}$	28.60$_{\pm 4.29}$	24.04$_{\pm 3.76}$	20.47$_{\pm 6.44}$	15.11$_{\pm 3.78}$	12.32$_{\pm 5.07}$	9.64$_{\pm 6.26}$	13.48$_{\pm 1.45}$
CF-Regression	70.02$_{\pm 0.33}$	62.33$_{\pm 0.78}$	59.05$_{\pm 0.24}$	57.19$_{\pm 0.42}$	55.58$_{\pm 0.56}$	54.08$_{\pm 0.43}$	53.03$_{\pm 0.64}$	51.97$_{\pm 0.64}$	50.50$_{\pm 0.56}$	49.10$_{\pm 0.45}$	47.68$_{\pm 0.44}$
RM-Classification	72.76$_{\pm 0.50}$	64.22$_{\pm 0.65}$	60.68$_{\pm 0.56}$	58.54$_{\pm 0.46}$	56.64$_{\pm 0.72}$	55.44$_{\pm 0.33}$	53.98$_{\pm 0.46}$	52.71$_{\pm 0.44}$	51.14$_{\pm 0.59}$	50.06$_{\pm 0.55}$	48.63$_{\pm 0.36}$
RM-Softmax	**77.58**$_{\pm 0.49}$	**64.49**$_{\pm 0.50}$	**61.97**$_{\pm 0.52}$	60.17$_{\pm 0.61}$	58.20$_{\pm 0.46}$	56.21$_{\pm 0.40}$	54.34$_{\pm 0.51}$	52.86$_{\pm 0.79}$	50.96$_{\pm 0.92}$	49.70$_{\pm 1.19}$	47.69$_{\pm 1.65}$
RM-Interval	**77.58**$_{\pm 0.49}$	63.70$_{\pm 1.47}$	**61.97**$_{\pm 0.52}$	59.43$_{\pm 1.56}$	58.20$_{\pm 0.46}$	56.31$_{\pm 0.47}$	54.34$_{\pm 0.51}$	52.55$_{\pm 0.91}$	50.96$_{\pm 0.92}$	49.23$_{\pm 1.81}$	47.69$_{\pm 1.65}$

Table 5: Utility for **SPOUT** with `BERT` embeddings. The best-performing method for each column is highlighted in bold.

Method	$\lambda = 0$	$\lambda = 100$	$\lambda = 200$	$\lambda = 300$	$\lambda = 400$	$\lambda = 500$	$\lambda = 600$	$\lambda = 700$	$\lambda = 800$	$\lambda = 900$	$\lambda = 1000$
Full-Feedback	*85.99*$_{\pm 0.17}$	*79.34*$_{\pm 0.13}$	*75.55*$_{\pm 0.29}$	*72.16*$_{\pm 0.24}$	*69.47*$_{\pm 0.23}$	*66.97*$_{\pm 0.37}$	*64.79*$_{\pm 0.31}$	*63.18*$_{\pm 0.21}$	*61.43*$_{\pm 0.27}$	*59.98*$_{\pm 0.30}$	*58.83*$_{\pm 0.24}$
Baseline	66.88$_{\pm 1.55}$	64.62$_{\pm 0.66}$	61.85$_{\pm 0.78}$	59.15$_{\pm 0.84}$	56.73$_{\pm 0.67}$	54.08$_{\pm 0.57}$	51.17$_{\pm 0.54}$	48.79$_{\pm 0.67}$	45.74$_{\pm 1.21}$	42.58$_{\pm 1.36}$	38.97$_{\pm 2.65}$
R&C	83.34$_{\pm 0.32}$	76.45$_{\pm 0.53}$	72.40$_{\pm 0.56}$	68.59$_{\pm 0.69}$	65.34$_{\pm 0.66}$	63.19$_{\pm 0.83}$	60.98$_{\pm 0.81}$	59.01$_{\pm 0.86}$	57.21$_{\pm 0.99}$	55.97$_{\pm 1.11}$	54.33$_{\pm 1.37}$
Causal-CARROT (kNN)	84.52$_{\pm 0.35}$	76.55$_{\pm 0.37}$	71.34$_{\pm 0.23}$	67.82$_{\pm 0.43}$	65.38$_{\pm 0.40}$	63.38$_{\pm 0.43}$	61.77$_{\pm 0.43}$	60.39$_{\pm 0.38}$	59.07$_{\pm 0.37}$	57.99$_{\pm 0.35}$	56.99$_{\pm 0.34}$
Causal-CARROT (EmbedNet)	83.46$_{\pm 0.39}$	68.73$_{\pm 0.74}$	62.44$_{\pm 2.40}$	56.72$_{\pm 1.68}$	54.37$_{\pm 2.12}$	48.91$_{\pm 2.41}$	46.35$_{\pm 1.53}$	43.66$_{\pm 3.04}$	41.81$_{\pm 2.34}$	37.42$_{\pm 5.86}$	34.69$_{\pm 4.44}$
CF-Regression	83.54$_{\pm 0.25}$	76.65$_{\pm 0.46}$	72.42$_{\pm 0.53}$	68.95$_{\pm 0.60}$	65.81$_{\pm 0.67}$	63.58$_{\pm 0.67}$	61.37$_{\pm 0.64}$	59.52$_{\pm 0.54}$	57.72$_{\pm 0.72}$	56.31$_{\pm 0.66}$	54.56$_{\pm 0.71}$
RM-Classification	84.20$_{\pm 0.24}$	**77.58**$_{\pm 0.34}$	73.36$_{\pm 0.49}$	69.84$_{\pm 0.48}$	66.49$_{\pm 0.63}$	64.03$_{\pm 1.02}$	61.84$_{\pm 1.13}$	59.68$_{\pm 1.48}$	58.09$_{\pm 1.22}$	56.52$_{\pm 1.63}$	54.85$_{\pm 1.76}$
RM-Softmax	**84.97**$_{\pm 0.39}$	77.53$_{\pm 0.81}$	**73.89**$_{\pm 0.00}$	**70.47**$_{\pm 0.00}$	**67.38**$_{\pm 0.47}$	**65.51**$_{\pm 0.60}$	**64.03**$_{\pm 0.39}$	**62.04**$_{\pm 1.17}$	**60.32**$_{\pm 1.31}$	**58.85**$_{\pm 1.13}$	**56.95**$_{\pm 0.55}$
RM-Interval	**84.97**$_{\pm 0.39}$	77.60$_{\pm 0.62}$	**73.89**$_{\pm 0.00}$	69.92$_{\pm 0.57}$	**67.38**$_{\pm 0.47}$	65.20$_{\pm 0.67}$	**64.03**$_{\pm 0.39}$	61.54$_{\pm 1.31}$	**60.32**$_{\pm 1.31}$	58.58$_{\pm 1.15}$	**56.95**$_{\pm 0.55}$

Table 6: Utility for **SPOUT** with `LLaMa-3.2-1B` embeddings. The best-performing method for each column is highlighted in bold.

Method	$\lambda = 0$	$\lambda = 100$	$\lambda = 200$	$\lambda = 300$	$\lambda = 400$	$\lambda = 500$	$\lambda = 600$	$\lambda = 700$	$\lambda = 800$	$\lambda = 900$	$\lambda = 1000$
Full-Feedback	*85.19*$_{\pm 0.17}$	*78.50*$_{\pm 0.35}$	*74.47*$_{\pm 0.28}$	*70.87*$_{\pm 0.23}$	*67.44*$_{\pm 0.22}$	*64.87*$_{\pm 0.54}$	62.69$_{\pm 0.36}$	*61.01*$_{\pm 0.57}$	*59.51*$_{\pm 0.55}$	*57.75*$_{\pm 0.60}$	*56.22*$_{\pm 0.58}$
Baseline	61.12$_{\pm 1.95}$	57.12$_{\pm 1.66}$	52.35$_{\pm 2.63}$	51.23$_{\pm 1.04}$	47.86$_{\pm 2.19}$	42.50$_{\pm 2.64}$	40.05$_{\pm 4.79}$	34.20$_{\pm 5.47}$	30.20$_{\pm 5.82}$	29.64$_{\pm 5.25}$	27.99$_{\pm 5.66}$
R&C	81.34$_{\pm 0.41}$	74.75$_{\pm 0.54}$	68.97$_{\pm 5.81}$	67.09$_{\pm 0.67}$	64.62$_{\pm 0.57}$	62.33$_{\pm 0.63}$	60.22$_{\pm 0.46}$	58.81$_{\pm 0.48}$	56.86$_{\pm 0.81}$	55.76$_{\pm 1.17}$	54.10$_{\pm 1.48}$
Causal-CARROT (kNN)	84.50$_{\pm 0.38}$	76.30$_{\pm 0.42}$	71.24$_{\pm 0.67}$	68.02$_{\pm 0.54}$	65.57$_{\pm 0.53}$	63.62$_{\pm 0.47}$	61.93$_{\pm 0.43}$	60.33$_{\pm 0.41}$	**59.05**$_{\pm 0.37}$	**57.86**$_{\pm 0.35}$	56.89$_{\pm 0.29}$
Causal-CARROT (EmbedNet)	81.29$_{\pm 0.48}$	62.96$_{\pm 1.32}$	55.69$_{\pm 4.27}$	49.40$_{\pm 5.41}$	45.21$_{\pm 4.47}$	32.52$_{\pm 14.00}$	31.05$_{\pm 11.57}$	31.70$_{\pm 6.75}$	22.34$_{\pm 9.09}$	20.47$_{\pm 9.64}$	15.52$_{\pm 6.70}$
CF-Regression	82.32$_{\pm 0.45}$	75.55$_{\pm 0.43}$	70.24$_{\pm 5.37}$	68.26$_{\pm 0.64}$	65.55$_{\pm 0.58}$	63.51$_{\pm 0.52}$	61.49$_{\pm 0.53}$	59.90$_{\pm 0.60}$	58.19$_{\pm 0.60}$	56.86$_{\pm 0.61}$	55.61$_{\pm 0.76}$
RM-Classification	83.60$_{\pm 0.43}$	77.17$_{\pm 0.47}$	70.95$_{\pm 6.42}$	69.36$_{\pm 0.62}$	66.38$_{\pm 0.76}$	63.99$_{\pm 0.89}$	61.82$_{\pm 0.71}$	60.45$_{\pm 0.68}$	58.38$_{\pm 0.71}$	56.83$_{\pm 0.83}$	55.64$_{\pm 1.35}$
RM-Softmax	**84.60**$_{\pm 0.89}$	**77.48**$_{\pm 1.21}$	**71.45**$_{\pm 6.01}$	69.91$_{\pm 0.85}$	**67.18**$_{\pm 0.51}$	**65.35**$_{\pm 0.60}$	**63.04**$_{\pm 1.35}$	**61.38**$_{\pm 1.61}$	59.04$_{\pm 1.05}$	57.72$_{\pm 0.33}$	**57.12**$_{\pm 0.21}$
RM-Interval	**84.60**$_{\pm 0.89}$	76.54$_{\pm 4.06}$	**71.45**$_{\pm 6.01}$	**69.99**$_{\pm 0.62}$	**67.18**$_{\pm 0.51}$	64.73$_{\pm 0.97}$	**63.04**$_{\pm 1.35}$	61.16$_{\pm 1.84}$	59.04$_{\pm 1.05}$	57.48$_{\pm 2.90}$	**57.12**$_{\pm 0.21}$

We also report the AUC (Area Under the Curve) of the accuracy–cost trade-off curve for each method. This is computed using the `sklearn.metrics.auc` function Buitinck et al. [2013]. AUC provides a single scalar summary of performance that captures how well a model balances accuracy and computational cost. A higher AUC indicates a more favorable overall trade-off across budgets, making it a robust evaluation metric for comparing routing strategies.

Table 7: Average AUC over 10 trials across datasets and embedding models. Higher is better. Abbreviations: RB = RouterBench, SP = SPROUT.

Method	RB-BERT	RB-LLaMa	SP-BERT	SP-LLaMa
Full-Feedback	0.1839	0.1719	0.1727	0.1655
Baseline	0.0890	0.0134	0.0255	0.0148
R&C	0.1539	0.1108	0.1316	0.1105
Causal-CARROT (kNN)	0.1441	0.1433	0.1642	0.1618
Causal-CARROT (EmbedNet)	0.1376	0.0842	0.1148	0.0768
CF-Regression	0.1412	0.1119	0.1352	0.1233
RM-Classification	0.1665	0.1389	0.1477	0.1436
RM-Softmax	**0.2286**	**0.2213**	**0.3320**	**0.2444**
RM-Interval	0.2285	0.2196	0.3320	0.2464

C Contribution of Each Component

The results highlight the impact of causal bias correction, end-to-end training, and regret-focused objectives as follows:

Causal Inference (Bias Correction): Our Baseline corresponds to CARROT without any causal correction - that is, a decoupled predictor trained directly on observational data without accounting for treatment bias. It consistently performs the worst across all settings. In contrast, incorporating causal techniques for bias correction yields substantial gains: both R&C and Causal-CARROT, which integrate causal adjustments into routing, achieve +10–15% routing accuracy at comparable cost, demonstrating that accounting for selection bias is critical for effective routing. These results validate that bias-aware routing significantly improves utility over naive predictors trained on biased data.

End-to-End Learning vs. Two-Stage: Among bias-corrected approaches, our end-to-end regret-minimizing methods (RM) consistently outperform the two-stage methods (CF-Regression, Causal-CARROT, R&C). The performance gap (+1–3% routing accuracy at comparable cost) demonstrates the benefit of an integrated end-to-end approach: by directly optimizing the decision-quality objective (regret) rather than optimizing intermediate predictions, our method achieves superior and more stable results.

Regret Minimization Objective: Even compared to other causal learners, our specific training objective provides an edge. For instance, RM-Softmax (differentiable surrogate) slightly outperforms RM-Classification (upper-bound surrogate) in most cases, with lower variance (+0.5–1% routing accuracy at comparable cost). This highlights the advantage of our softmax-weighted regret objective and its alignment with the true decision loss. Moreover, both RM methods outperform Causal-CARROT and CF-Regression, underscoring that minimizing expected regret is more effective than surrogate approaches that focus only on accuracy/cost prediction.

D Experimental Details

All experiments were implemented in Python 3.8.12 Van Rossum and Drake [2009], using PyTorch 2.4.1+cu121 Paszke et al. [2019] and Scikit-learn Buitinck et al. [2013]. Experiments were conducted on an internal compute cluster equipped with an Intel(R) Xeon(R) Platinum 8260 CPU @ 2.40GHz, 512 GB of RAM, and two NVIDIA V100 GPUs with 16 GB memory each.

Prompt Encoding & Augmentation To encode input queries into vector representations x, we employ a two-stage embedding process. First, we enrich each prompt with contextual metadata by prepending a natural language prefix that identifies the source dataset. Specifically, for a prompt `p` originating from dataset `D` (e.g., `openhermes/teknium`), we construct the following context-augmented input: *"The following prompt comes from the dataset D. The prompt is: `p`"*. This step provides the embedding model with useful dataset-level context, which is particularly beneficial in multi-domain routing scenarios. The template is flexible and can be extended to include additional metadata if desired.

Datasets. **RouterBench** [Hu et al., 2024] is a standardized benchmark for LLM routing, comprising 35,712 prompt-response pairs collected from 11 LLMs. The prompts span eight different evaluation benchmarks covering reasoning, factual knowledge, dialogue, mathematics, and code generation. Each prompt is annotated with model accuracy and execution cost, enabling response-based decision-making. To maintain consistency in evaluation, we adopt the same split strategy for RouterBench, applied deterministically at the prompt level to ensure reproducibility. **SPROUT** [Somerstep et al., 2025] is a more recent and larger benchmark for cost-aware routing, consisting of 44,241 prompts and responses from 13 state-of-the-art language models. The prompts are drawn from six diverse benchmarks, including GPQA [Rein et al., 2024], MuSR [Sprague et al., 2023], MMLU-Pro [Wang et al., 2024], MATH [Hendrycks et al., 2021], OpenHermes [Teknium, 2023], and RAGBench [Friel et al., 2024]. SPROUT includes a predefined train/validation/test split, using 80% of the data for training and splitting the remaining 20% equally between validation and test sets.

Neural Router Models. All neural models used in our experiments share the same architecture for fairness and comparability. We use a 2-layer feedforward neural network with GELU activation and 200 hidden units per layer. Models are trained using the Adam optimizer with a learning rate of 10^{-4}, batch size of 128, and a maximum of 10,000 epochs. Early stopping is applied with a patience of 100 epochs based on validation regret. The temperature parameter for the softmax-based regret objective is set to 100, and to 1000 for the interval model to allow smoother gradients across budget intervals.

Doubly Robust Estimation. For the outcome model $\hat{r}_t(x)$, we use the same neural architecture described above, trained separately for each treatment t. For the propensity model $\hat{p}(t|x)$, we use an XGBoost classifier with the following hyperparameters: maximum depth = [1,2,3,5], number of trees = [10,20,50,100]. The estimated DR scores are clipped to the [5^{th}, 95^{th}] percentile to reduce the impact of extreme propensity weights and improve training stability.

Embedding Generation. We generate sentence-level embeddings using the `bert-base-uncased` (768-dim) and `meta-llama/Llama-3.2-1B` (2048-dim) models. Embeddings are extracted via mean pooling over the final hidden states and are precomputed in batches using GPU acceleration. These embeddings are fixed during training of all downstream routing models.

RM-Interval Network. The joint model used for budget interpolation is implemented using a small feedforward network that takes as input the concatenation of outputs from f_{λ_j}, $f_{\lambda_{j+1}}$, and a linear embedding of λ. The architecture mirrors the router described above and is fine-tuned using the regret objective over interval-specific training data. The proposed architecture is presented in Figure 1.

Inference latency and computational efficiency Latency is critical in real-time applications. Our method is designed to minimize this by using lightweight routing networks (2-layer MLP with 200 neurons per hidden layer) and precomputed light-weight embeddings (Llama-3.2-1B and BERT). For instance, with Llama-3.2-1B embeddings and an MLP-based router on the RoutherBench dataset, the end-to-end routing latency is under 2.5 ms on a single A100 GPU for a batch size of almost 25,000. To contextualize this, recent benchmarks show that on 8× A100s, LLaMA-2 or LLaMA-3 70B can generate a 100-token paragraph in 1.5 to 2.5 seconds under realistic conditions using optimized inference stacks. Even when using fewer GPUs or smaller models, generation latency typically remains one to two orders of magnitude higher than our routing time. In many cases (e.g., longer outputs or cold starts), the difference can be significantly larger. Thus, the routing overhead is negligible relative to the cost of LLM generation and does not meaningfully impact user experience. Additionally, the interval-conditioned architecture reduces deployment complexity by avoiding the need to train or store separate models for different user preferences.

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: The abstract and introduction accurately reflect the paper's contributions and scope. They clearly articulate the core innovations: (i) learning LLM routing policies from observational data, (ii) introducing an end-to-end regret minimization framework that avoids the limitations of decoupled approaches, (iii) developing differentiable surrogate objectives for training, and (iv) extending the method to support heterogeneous cost preferences via an interval-conditioned model. These contributions are substantiated in the main text, both theoretically and empirically, and are consistent with the stated goals of enabling scalable, bias-aware LLM routing in realistic deployment settings.

   Guidelines:

   - The answer NA means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A No or NA answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: The paper discusses its limitations explicitly. We note that the current framework is restricted to affine utility functions, which limits flexibility in modeling more complex user preferences (e.g., hard constraints that cannot be regularized in the objective). In the conclusion, we also outline directions for future work, including extending the method to dynamic or online learning settings and adapting it to multi-turn interactions. Additionally, the paper makes certain assumptions to support theoretical guarantees, including a mild Lipschitz continuity condition, and others that are standard in the causal inference literature (e.g., unconfoundedness). These are discussed in the relevant sections and are typical for counterfactual utility estimation from observational data.

   Guidelines:

   - The answer NA means that the paper has no limitation while the answer No means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.

- If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
- While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

   Answer: [Yes]

   Justification: All theorems, assumptions, propositions, and corollaries are included in the paper. Their complete proofs can be found in the Appendix.

   Guidelines:

   - The answer NA means that the paper does not include theoretical results.
   - All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
   - All assumptions should be clearly stated or referenced in the statement of any theorems.
   - The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
   - Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
   - Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes]

   Justification: You can find all the details to reproduce the experiments in Section 4 and in Section D of the Appendix. We will also release code in the supplementary material.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - If the paper includes experiments, a No answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
   - If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
   - Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
   - While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
     (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.

(b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
(c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
(d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

   Answer: [Yes]

   Justification: Our paper provides all the details to reproduce the experimental results in Section 4 and in Section D of the Appendix. Furthermore, the code used is provided in the supplemental materials.

   Guidelines:

   - The answer NA means that paper does not include experiments requiring code.
   - Please see the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - While we encourage the release of code and data, we understand that this might not be possible, so "No" is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
   - The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
   - The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
   - At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
   - Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer, etc.) necessary to understand the results?

   Answer: [Yes]

   Justification: We report all training and evaluation details in Section 4 and Appendix D.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

Answer: [Yes]

Justification: We run each experiment 10 times and report mean utility along with standard deviation (1-sigma). The variability reflects random initialization and train/validation splits.

Guidelines:

- The answer NA means that the paper does not include experiments.
- The authors should answer "Yes" if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
- The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
- The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
- The assumptions made should be given (e.g., Normally distributed errors).
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g. negative error rates).
- If error bars are reported in tables or plots, The authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: We state the computer resources used in Section D of the Appendix.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification: We have reviewed the NeurIPS Code of Ethics, and our work complies with its guidelines in all respects.

   Guidelines:

   - The answer NA means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer No, they should explain the special circumstances that require a deviation from the Code of Ethics.

- The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [Yes]

    Justification: The paper highlights potential positive societal impacts of this work. By enabling efficient LLM routing using observational data, our framework reduces the need for full-feedback supervision, significantly lowering computational and monetary costs. LLM routing itself contributes to more sustainable AI deployment by reducing energy consumption and resource usage. We do not anticipate negative societal impacts, as the method focuses on infrastructure-level efficiency improvements and does not involve sensitive user data or content generation.

    Guidelines:

    - The answer NA means that there is no societal impact of the work performed.
    - If the authors answer NA or No, they should explain why their work has no societal impact or why the paper does not address societal impact.
    - Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
    - The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
    - The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
    - If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pretrained language models, image generators, or scraped datasets)?

    Answer: [Yes]

    Justification: Yes. The paper uses publicly available benchmark datasets, and we plan to release our code to support transparency and reproducibility. As our work focuses on routing policies rather than generating model outputs, and does not involve training or releasing new pretrained language models or sensitive datasets, we do not identify any specific risks or misuse concerns requiring additional safeguards.

    Guidelines:

    - The answer NA means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
    - Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.

- We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [Yes]

    Justification: All external assets used in the paper, including benchmark datasets, embedding models, are properly cited in the main text. We have respected the terms of use and licenses associated with these resources, and references are provided to acknowledge the original creators.

    Guidelines:

    - The answer NA means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.
    - The name of the license (e.g., CC-BY 4.0) should be included for each asset.
    - For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
    - If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
    - For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
    - If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [Yes]

    Justification: We introduce new code assets to implement our causal end-to-end LLM routing framework. These assets are well documented, with usage instructions and reproducibility details provided to facilitate adoption and verification. Documentation will be released alongside the code repository.

    Guidelines:

    - The answer NA means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [NA]

    Justification: We did not conduct crowdsourcing experiments and research with human subjects.

Guidelines:

- The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
- Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
- According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [NA]

    Justification: The paper does not involve crowdsourcing nor research with human subjects.

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
    - Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
    - We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
    - For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

    Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does not impact the core methodology, scientific rigorousness, or originality of the research, declaration is not required.

    Answer: [NA]

    Justification: We used LLMs for grammar, formatting, and rephrasing assistance (e.g., improving proof presentation). They did not influence the methodological originality of the work.

    Guidelines:

    - The answer NA means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
    - Please refer to our LLM policy (`https://neurips.cc/Conferences/2025/LLM`) for what should or should not be described.