MARS: A Benchmark for Multi-LLM Algorithmic Routing System

Anonymous authors
Paper under double-blind review

Abstract

As the range of applications for Large Language Models (LLMs) continues to grow, the demand for effective serving solutions becomes increasingly critical. Despite the versatility of LLMs, no single model can optimally address all tasks and applications, particularly when balancing performance with cost. This limitation has led to the development of LLM routing systems, which combine the strengths of various models to overcome the constraints of individual LLMs. Yet, the absence of a standardized benchmark for evaluating the performance of LLM routers hinders progress in this area. To bridge this gap, we present MARS, a novel evaluation framework designed to systematically assess the efficacy of LLM routing systems, along with a comprehensive dataset comprising over 405k inference outcomes from representative LLMs to support the development of routing strategies. We further propose a theoretical framework for LLM routing, and deliver a comparative analysis of various routing approaches through MARS, highlighting their potentials and limitations within our evaluation framework. This work not only formalizes and advances the development of LLM routing systems but also sets a standard for their assessment, paving the way for more accessible and economically viable LLM deployments.

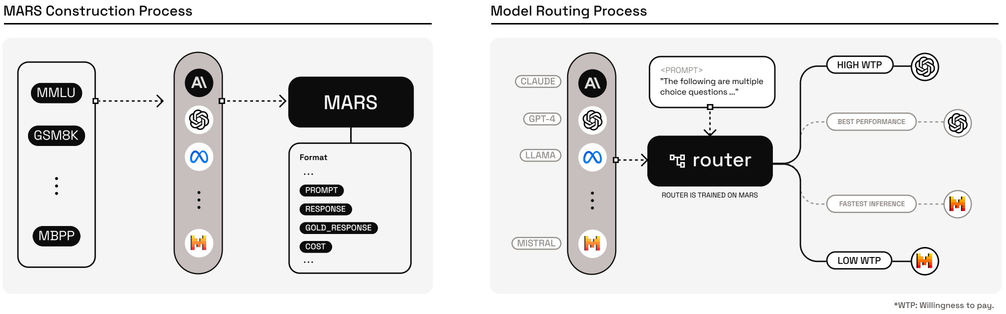

Figure 1: **Left**: The MARS Construction Process integrates eight datasets with eleven distinct models to develop MARS. Detailed format can be found in Appendix A.4. **Right**: The Model Routing Process shows the method of routing prompts through a router to various LLMs based on specific requests, demonstrating the dynamic allocation of resources.

1 Introduction

Large Language Models (LLMs) have exhibited remarkable capabilities in addressing a wide range of tasks across academic and industrial scenarios Bubeck et al. (2023). This has motivated both researchers and practitioners to introduce new LLMs, designed for both generic and specialized use cases, on a near-daily basis [1]. However, the proliferation of LLMs presents a challenge for LLM application builders to identify the most suitable model for their applications. While some proprietary models such as GPT-4 are distinguished by their superior performance, they often incur high economic costs due to the expensive API prices.

[1] As of January 16th, 2024, there are 469,848 models listed on huggingface.com

Many prior works focus on improving the capabilities of individual LLMs while maintaining low costs. Techniques such as prompting Wei et al. (2023), quantization Lin et al. (2023); Kim et al. (2023), and system optimization Kwon et al. (2023) may reduce a single model's serving cost, yet with new models emerging daily, these approaches may not remain feasible or scalable in long term. Moreover, the diversity of choices of LLMs available at various price and performance tiers can be daunting for users attempting to select and optimize an appropriate model[2].

An alternative solution aims to select to optimal LLM for each input through "routing." Yue et al. (2023); Shnitzer et al. (2023); Šakota et al. (2023). Routing offers several advantages over single-LLM optimization. First, it is a lightweight process, which treats each LLM as an input-output black box, avoiding the need to delve into intricate infrastructure details, thus making it flexible and broadly applicable. Second, routing systems benefit from the diversity of LLMs, while single-LLM methods may struggle to keep pace with the expanding LLMs landscape. Lastly, while single-LLM strategies often face a compromise between performance and other factors such as per-token costs, routing systems adeptly balance a spectrum of user demands Lee et al. (2023); Lu et al. (2023); Chen et al. (2023).

The rise in routing-related research has improved cost-efficiency, enhanced performance, and broadened accessibility. Despite these advances, a comprehensive benchmark for evaluating routing techniques remains absent. We introduce MARS, the first comprehensive benchmark designed specifically for assessing router mechanisms in terms of inference dollar cost and performance. MARS encompasses a diverse array of tasks and domains, with pre-generated LLM response and quality metrics, on which different routing mechanisms can be efficiently tested without inference. Our experiments revealed that while some previous routing mechanisms have difficulty generalizing to complex tasks and up-to-date models, there are several promising fields on which even simple routing demonstrated outstanding performance.

In conclusion, we present the following key contributions:

1. We have developed a comprehensive benchmark for LLM routing covering major tasks for LLMs, which includes a wide range of both open-source and proprietary models. MARS enables efficient training and testing of model routers without inference, and can be flexibly extended to cover new tasks and models.
2. We introduce a theoretical framework designed to assess the efficacy of routers across several metrics, with a particular emphasis on inference cost (expressed in dollars) and performance. This framework includes mathematical formulations that enable the seamless integration and comparative analysis of various routers and LLMs.
3. We evaluate the efficiency of routing strategies across a broad range of tasks. Our results provide insights into the performance of various routers in different contexts and demonstrate that the monetary costs of LLM services can routinely vary by factors of 2-5× for comparable levels of performance. This underscores the significance and utility of our benchmark, highlighting promising areas for future enhancements.

2 Related Work

Various strategies have been proposed to optimize the cost and performance of current LLMs. We provide an overview of them with a focus on routing-related approaches.

2.1 Single LLM Enhancement

Fine-tuning is used to improve models for specific tasks, which requires additional training and domain-specific data Rafailov et al. (2023). Prompting mechanisms like Chain-of-Thought (CoT) Wei et al. (2023); Zhou et al. (2023); Wang et al. (2023a), Tree of Thoughts (ToT) Yao et al. (2023), and Algorithm of Thoughts (AOT) Sel et al. (2023) could bolster LLM performance without additional fine-tuning. However, these single-LLM enhancements are usually model and scenario specific, and could not benefit from the explosion of LLMs.

[2] As of January 29th, 2024, there are 22,177 language models with 7 billion parameters listed on huggingface.com

2.2 LLM Synthesis

Beyond single LLM approaches, LLM synthesis utilizes the ensemble of multiple LLMs, integrating their outputs into an enhanced final result Jiang et al. (2023). Another approach has shown that a strategic combination of smaller models can match or even outperform larger models Lu et al. (2024). However, these methods require at least two steps: text generation and synthesis, which lead to increased costs and latency and challenge the implementation of this method in production.

2.3 Routing

Unlike LLM Synthesis, routing can select the suitable model for specific input without performing inference on every candidate model. Routing can be classified in two categories, non-predictive routing and predictive routing. Non-predictive routing strategies retrieve outputs from LLMs and directly pick one without a model-assisted synthesis step. Several studies Madaan et al. (2023); Yue et al. (2023); Lee et al. (2023); Chen et al. (2023) have explored systems that integrate Small Language Models (SLMs) with Large Language Models (LLMs). Another methodology involves a layered inference framework, re-routing more complex queries to an advanced model for improved results Wang et al. (2023b). Predictive routing selects the optimal LLM without requiring to evaluate the output. Some research have implemented routers utilizing supervised learning algorithms Shnitzer et al. (2023), while some other using reward model-based techniques Hari & Thomson (2023); Lu et al. (2023). Furthermore, meta-model, trained on inputs along with a model-specific token to predict the performance score, represents another approach to determining the most appropriate LLM for use Šakota et al. (2023). In short, predictive router could bring substantial cost and performance improvement without sacrificing latency, with a number of early works dedicated to this field.

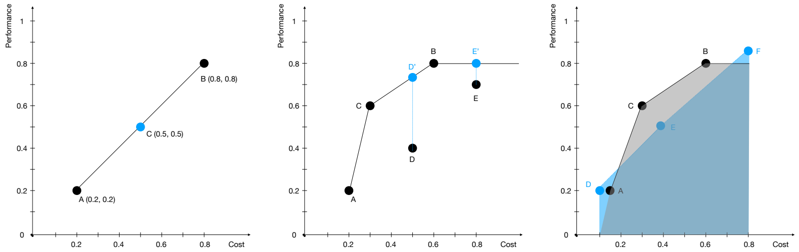

Figure 2: **Left**: linear interpolation is the process of achieving the cost-performance trade-off between any concrete routers. Point A and B are routers with different input parameters. To achieve the average of A and B, we build router C which routes to A or B with 50% probability each, and it performs the average of A and B in expectation. **Middle**: Consider points A to E, we can construct the non-decreasing convex hull consists of points A, B, and C. D and E as they can be replaced by strictly superior affine combination of A, B, and C. **Right**: ABC and DEF are two routing systems (already convexified with ABC extended to (0.1,0) for fair comparison). To compare, we interpolate A and B to $c_{min} = 0.1$ and $c_{max} = 0.8$, respectively, and then calculate the area under the curve normalized by $c_{max} - c_{min}$ to derive AIQ.

3 Math Formulation for Router Evaluation

The primary challenge in assessing the performance of routing systems lies in balancing two conflicting objectives: maximizing efficiency and minimizing cost. To effectively compare routers, we have developed a framework that captures the multi-faceted nature with one metric.

3.1 Setup and Basic Operations

Consider a set of models $L = \{LLM_1, \ldots, LLM_m\}$ a dataset D consisting of examples $x_i \in \{x_1, ..., x_{|D|}\}$. For each model LLM_j, we evaluate its performance by generating an output $o_i^j = LLM_j(x_i)$ for each example x_i. Each output o_i^j has two associated quantities: the cost $c(o_i^j)$ of generating that output and the quality or performance $q(o_i^j)$ of the output itself. Through this process, we establish an expected cost c_m and an expected quality q_m for each model LLM_m across the dataset D.

$$c_m = E[c(LLM_m(x))|x \in D]$$
$$q_m = E[q(LLM_m(x))|x \in D]$$

A *router* R, define as a function, takes in a prompt x and a set of parameters θ, subsequently selecting the most suitable model LLM_i from a set L to complete the prompt, i.e.

$$R_\theta(x) \mapsto LLM_i \in L.$$

The parameters θ typically include maximum price the user is willing to pay, the desired latency, or number of layers of neural networks for the router model, etc. More details of router parameters will be elaborated and discussed in Section 5.1.

The expected cost of a router R_{θ_1} across dataset D is defined as

$$c_{R_{\theta_1}}(D) = E[c(R_{\theta_1}(x))|x \in D]$$

and the expected performance of a router R_{θ_1} can be defined similarly.

By experimenting with various router parameters $\theta_1, ..., \theta_k$, we obtain a series of data points $(c_{R_{\theta_1}}, q_{R_{\theta_1}}), ..., (c_{R_{\theta_k}}, q_{R_{\theta_k}})$ which can be graphically represented in the cost-quality $(c - q)$ plane alongside the results of LLMs for comparative analysis.

Linear Interpolation The initial operation we introduce within this framework is *linear interpolation*, which enables the computation of a weighted average between any two points on the cost-quality $(c - q)$ plane.

Consider two routers, R_{θ_1} and R_{θ_2}. We can formulate a third router, $R_{int}(R_{\theta_1}, R_{\theta_2})$, based on the following principle: given a prompt x select $t \in [0, 1]$ such that:

$$R_{int}(R_{\theta_1}, R_{\theta_2}, t)(x) = \begin{cases} R_{\theta_1}(x), & \text{w.p. } t \\ R_{\theta_2}(x), & \text{w.p. } 1 - t \end{cases}$$

Through the principle of linearity of expectation, we can deduce the expected cost of $R_{int}(R_{\theta_1}, R_{\theta_2}, t)(x)$ in terms of LLM_1 and LLM_2:

$$E[c_{R_{int}(x)}|x \in D] = t \cdot c_{R_{\theta_1}} + (1 - t) \cdot c_{R_{\theta_2}}$$

and the expected performance of $R_{int}(R_{\theta_1}, R_{\theta_2}, t)(x)$ can be defined similarly.

Notably, for two data points (c_1, q_1) and (c_2, q_2) corresponding to R_{θ_1} and R_{θ_2} respectively, $R_{int}(t)$ can precisely interpolate any point along the line segment connecting (c_1, q_1) and (c_2, q_2).

Extrapolation To ensure all router can be enrich our mathematical framework, the second operation we introduce is *extrapolation* which enables the extension along the cost axis. For a given router R_θ and a positive integer k, we can construct an extrapolated router $R_{ext}(R_\theta, k)$. Upon receiving a prompt x, $R_{ext}(R_\theta, k)(x)$ executes $R(x)$ k times, ignoring the first $k - 1$ responses and considering only the final output. This approach enables linear interpolation with the derived output, achieving a desired performance level at a proportionally increased cost for sufficiently large values of k.

It is essential to note that the routers discussed are functionally analogous to LLMs within this context, as both can be represented as coordinates in the cost-quality $(c - q)$ plane.

3.2 Non-Decreasing Convex Hull

When working with multiple routers, it's feasible to construct any affine combination of points through linear interpolation among them. Specifically, for a set S of points in the cost-quality $(c-q)$ plane, these affine combinations can target any point (c, q) in $\mathbb{R}^2$ lying within the convex hull formed by S. We identify $S_{ch} \subseteq S$ as the subset of points that delineate the vertices of this convex hull.

Furthermore, it's possible to configure a non-decreasing convex hull from S_{ch}, ensuring that for any two points (c_1, q_1) and (c_2, q_2) where $c_2 \geq c_1$, it follows that $q_2 \geq q_1$. Intuitively, if the extra cost of $c_2 - c_1$ does not bring any performance improvement, it is advisable to simply extrapolate (c_1, q_1) to the domain of c_2, and (c_2, q_2) could be (c_2, q_1).

For a given routing system R_1, constituted by LLMs and routers plotted in the $c-q$ plane for dataset D, we can conceptualize a new routing system $\widetilde{R_1}$. This involves constructing routers $R_{\theta_1}, ..., R_{\theta_k}$, yielding points $(c_1, q_1), ..., (c_k, q_k)$. By establishing a non-decreasing convex hull S_{ndch} from these points and for any cost c within the range $[c_{min}, c_{max}]$, optimal performance is attainable by interpolating between the two closest cost points. This process effectively creates a new routing system spans the complete domain $[c_{min}, c_{max}]$.

Given the framework established, we define the **Zero Router** (R_{zero}) as a router that selects LLMs from $\{LLM_1, \ldots, LLM_m\}$ based on their collective non-decreasing convex hull. For a specified cost c, R_{zero} provides a probabilistic mix of LLMs that maximizes expected output quality with a simple, mathematics-driven routing strategy. R_{zero} serves as a basic benchmark for assessing the efficacy of other routing systems; a router is deemed significant only if it demonstrates superior performance compared to R_{zero}.

3.3 Comparing Different Routing Systems

Given the agnostic nature of our comparison framework towards the router's structure, routing systems can produce an assorted set of points on the $c-q$ plane that may be non-deterministic and non-parametric, complicating direct comparisons. Leveraging the methodologies delineated previously, we have the capacity to condense these disparate points into a streamlined function—specifically, a non-decreasing convex hull—and subsequently distill this representation into a singular metric that encapsulates the system's characteristics.

Routing systems often generate multiple points on the cost-quality ($c-q$) plane, making it difficult to compare the underlying systems. However, our framework allows us to transform these non-parametric points into a simpler function, specifically a non-decreasing convex hull, which can be characterized by a single numerical value.

Let's consider two different routing systems (for example KNN and MLP-based routers), R_θ where $\theta \in \Theta$, and R_λ where $\lambda \in \Lambda$. To compare their effectiveness, we parametrize them by sampling from Θ, Λ to generate a set of points: $R_{\theta_1}, \ldots, R_{\theta_k}$, and $R_{\lambda_1}, \ldots, R_{\lambda_k}$. Then, we construct non-decreasing convex hull for both groups, $\widetilde{R_\theta}$ and $\widetilde{R_\lambda}$, defined on a shared domain $[c_{min}, c_{max}]$.

We define AIQ (Average Improvement in Quality) for one of the routing system as follows:

$$AIQ(R_\theta) = \frac{1}{c_{max} - c_{min}} \int_{c_{min}}^{c_{max}} \widetilde{R_\theta} \mathrm{d}c$$

With the equation above, we can calculate AIQs for any group of routing systems to get a clear understanding of their relative performance. Rather than performing complex graphic analysis, AIQ allows users to measure router performance in a straightforward way.

4 Benchmark Construction - MARS

To systematically assess router performance, we have developed a dataset, MARS. This comprehensive dataset consists of a broad spectrum of tasks, including commonsense reasoning, knowledge-based language understanding, conversation, math, coding and retrieval-augmented generation (RAG). MARS is constructed by leveraging existing datasets that are widely recognized and utilized in the evaluation of leading LLMs, such as GPT-4, Gemini Team et al. (2023), and Claude Anthropic (2023). This approach ensures that MARS is representative of the diverse challenges and requirements pertinent to mainstream LLM performance evaluation.

4.1 Principles in benchmark construction

The construction of MARS are are guided by the following principles:

- Extensive Coverage: Our selection process identified a diverse array of fields where LLMs are widely utilized, aiming for wide-ranging applicability.
- Practical Relevance: The benchmarks chosen are of considerable significance to the industry's current applications of LLM systems, presenting a balanced challenge to the state-of-the-art LLMs, that is not too difficult nor too simplistic.

- Extensibility: MARSis designed for seamless integration of additional metrics, such as latencies and throughputs, ensuring adaptability to the evolving landscape of LLM.

4.2 Benchmark Dataset

For the initial release, we have curated a selection of 8 representative datasets from multiple different tasks. Detailed descriptions are in Appendix A.3.

- **Commonsense Reasoning**: Hellaswag Zellers et al. (2019), Winogrande Sakaguchi et al. (2021), and ARC Challenge Clark et al. (2018)
- **Knowledge-based Language Understanding**: MMLU Hendrycks et al. (2021)
- **Conversation**: MT-Bench Zheng et al. (2023b)
- **Math**: GSM8K Cobbe et al. (2021)
- **Coding**: MBPP Austin et al. (2021)

RAG Dataset: Additionally, We gathered 4000 prompts from different news sources and generate question with GPT-4 as an evaluation of routers on the retrieval-augmented generation tasks.

4.3 Dataset Construction Process

For the compilation of our benchmark dataset, we perform inference with 11 different LLMs[3], including both open-source and proprietary models. This process was applied to each of the eight datasets enumerated in Section 4.2. This is also illustrate in Figure 1. The selected LLMs are as follows and more details are in Appendix A.1:

Open Source Model: Llama-70B-chat, Mixtral-8x7b-chat, Yi-Chat 34B, Mistral-7b-chat

Proprietary Model: GPT-4-turbo, GPT-3.5-turbo, Claude-instant-v1, Claude-v1, Claude-v2

In total, there are 405,467 samples in MARS, covering 11 models, 8 datasets, and 64 tasks.

4.4 A Pilot Study: The Oracle Router

We assessed the performance of various models across the datasets, with more details in (A.5 and A.6) while aggregate results are illustrated in Figure 3. The *oracle* represents the best possible router: the one that always route to the best-performing LLM (if there are multiple of them, then route to the cheapest one).

Result: We observe that the oracle router has near-perfect performance with low cost, demonstrating the high ceiling of routing among LLMs. While proprietary models like GPT-4 excel in performance, they are significantly more expensive than open source counterparts (caveat on models refuse to answer B). Some open source models, like Mistral-7B, are substantially cheaper, at the cost of lower performance. The oracle router is able to combine the strength of different LLMs, achieves near-perfect performance with low cost, thereby indicating the strong potential for lossless improvement of LLM systems via routing.

5 Experiments

5.1 Predictive Router

We propose a novel set of predictive routers, which do not require pre-generation of LLM outputs. Specifically, we introduce a router $R : x_i \rightarrow$ LLM, constructed as follows: for an input x_i, the performance score for LLM_j is calculated via:

$$\text{performance score}_{ij} = \lambda \cdot P_{ij} - \text{cost}_j$$

P denotes the predicted performance of LLM_j on sample x_i, with λ representing the **willingness to pay (WTP)** parameter that delineates the cost-performance trade-off. A higher λ indicates a

[3] Model specific to the RAG task: You.com API, Perplexity-7B, Perplexity-70B

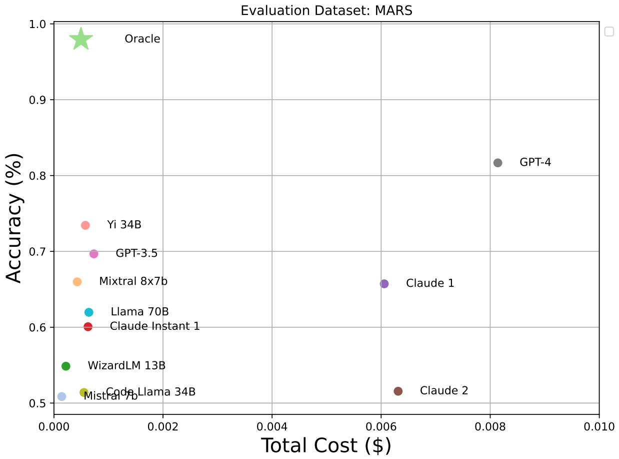

Figure 3: Accuracy vs Total Cost of eleven LLMs on MARS. The *oracle* represents the best possible router: the one that always route to the best-performing LLM (if there are multiple of them, then route to the cheapest one).

preference for superior performance at a higher cost. We approximate total cost using the cost per token metric. The routing decision for the predictive router is thus formulated as selecting the LLM that optimizes the performance score.

To estimate P for each input across models, we implemented two supervised regression approaches: **k-nearest neighbors (KNN)** and **multi-layer perceptron (MLP)** inspired by Shnitzer et al. (2023). We allocated a fraction of the dataset for training a performance predictor for each task, assessing its efficacy on the remainder.

Specifically, the **KNN router** estimates performance score$_{ij}$ by identifying the k nearest samples in the training set D_{train} and opting for LLM_i demonstrating optimal performance within this subset.

$$P_{\text{KNN}}(x_i) = \frac{1}{k} \sum_{x_j \in \text{NN}_k(x_i, D_{train})} q(o_j^i)$$

Where $NN_k(x_i, D_{train})$ signifies the subset of k nearest neighbors to the sample x_i within the training dataset D_{train}.

Similarly, for **MLP Router**, we have trained a set of MLP models to predict performance

$$P_{\text{MLP}}(x_i) = f(W_n \cdot \sigma(... \cdot \sigma(W_1 \cdot x_i + b_1)... + b_n)$$

Those series of KNN and MLP routers are trained with varying hyperparameters, and we present the experimental results derived from the optimal hyperparameter configurations.

5.2 Non-Predictive Routers

This category of routers generates answers from a sequence of Large Language Models (LLMs), evaluates these answers, and bases routing decisions on the evaluation outcomes. Drawing inspiration from Chen et al. (2023); Wang et al. (2023b), we introduce a *cascading router* comprising of a total cost parameter T, and a sequence of m LLMs, denoted as LLM_j : text $\rightarrow$ text, ranked from the least to the most expensive in terms of computational cost and expected accuracy. A key component of its operation is a scoring function g : text $\rightarrow [0, 1]$ paired with a threshold t (the ”judge”). Upon receiving a request, it is initially processed by LLM_1. If $g(o_1) > t$, the output o_1 is selected and the process terminates; otherwise, if the cumulative cost is still less than total cost T, the router proceeds to the next LLM in the sequence, and returns the current output if not.

Although developing an effective scoring function g for a specific task in a production setting presents challenges, within the context this paper, the router possesses perfect knowledge of the final score, enabling it to consistently select the most cost-effective model that yields a satisfactory response (akin to an oracle). To simulate real-world performance more accurately, we introduce an error parameter $\epsilon \in [0, 1]$. The adjusted scoring function $g_\epsilon(o)$ is defined as:

$$g_\epsilon(o) = \begin{cases} 1 - g(o) & \text{with probability } \epsilon \\ g(o) & \text{with probability } 1 - \epsilon \end{cases}$$

A variant of non-predictive router is overgenerate-and-rerank, which generates all potential outcomes from the LLM, assesses each, and outputs the optimal one as determined by a designated reward function. Although its practical application is limited due to significant costs, we will present its results for demonstration.

5.3 Main Results

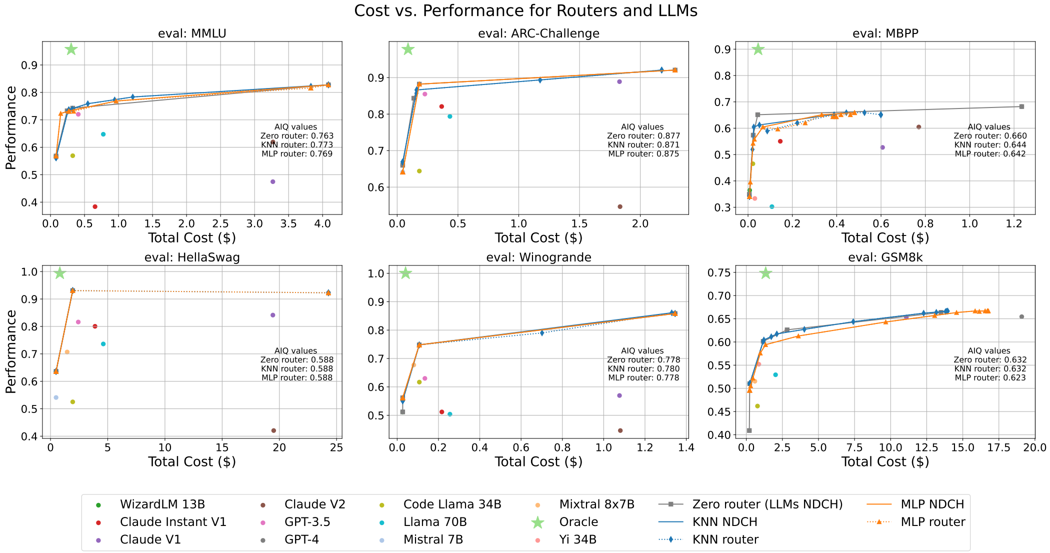

Figure 4: Total Cost vs Performance for eleven models and KNN, MLP and Zero Routers on MARS except MT-Bench. For KNN and MLP, we tested different hyper-parameters and results of the optimal ones are displayed above. The AIQ values are also calculated for all three routers. NDCH stands for non-decreasing convex hull.

Predictive Router With the KNN and MLP router design, we present the performances of predictive routers across all tasks (other than MT-Bench). The dataset for each task is randomly partitioned into two splits, where the routers are trained on 70% and evaluated on the rest 30%. We exclude MT-Bench in this set of experiments due to its limited size to perform such a train-test partition. As shown in Figure 4, both KNN routers and MLP routers achieve the level of performance to the best individual LLMs with lower or similar costs, demonstrating the effectiveness of the proposed routing solutions, despite their simplicity. However, none of the routing algorithms significantly outperform the baseline Zero Router (The routers exhibit higher AIQ than the Zero Router for MMLU and Winogrande, achieved comparable AIQ for Hellaswag and GSM8K, and underperform on Arc-challenge and MBPP), the *oracle* router consistently exceeds all other routers and LLMs in performance, underscoring the room for further advancements in routing algorithms design.

Cascading Router We present results for cascading routers on MMLU, MBPP, and GSM8K in Figure 5. The results indicate that with each error rate, as the total cost T increases, the cascading router's performance improves due to the availability of a larger budget for selecting more appropriate models. For lower error rates, the cascading router demonstrates superior performance compared to the Zero Router, as evidenced by the higher AIQ value. The router with a zero error rate judge quickly approximates the performance of the *oracle* at the same cost and achieves comparable results as the cost further increases. Figure 5 illustrates the cascading routers' effectiveness, showing they surpass both individual LLMs and the Zero Router by a significant margin when the router's judge has an error rate of up to 0.1. This indicates the routing technique's potential when paired with an effective judge.

However, as the judge's error rates increase, the performance of the cascading router may deteriorate rapidly, particularly when the error rate exceeds 0.2. Achieving a sufficiently low error rate

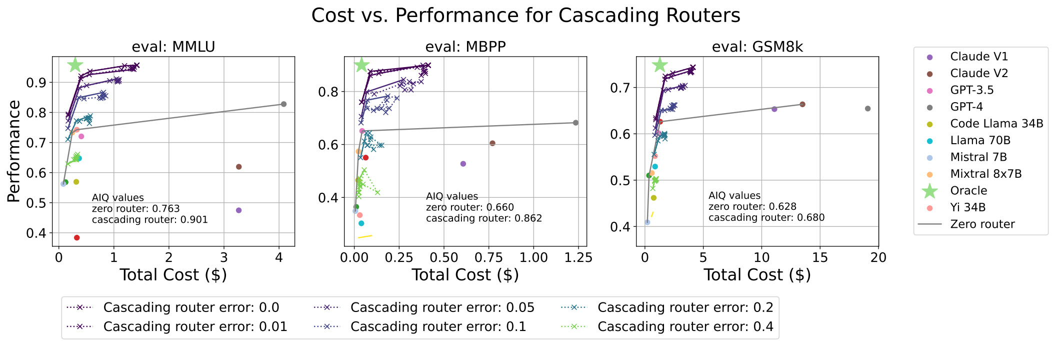

Figure 5: Total Cost vs Performance for eleven models and cascading routers on MMLU, MBPP, and GSM8K. Different error rates are tested, and the AIQ value is computed for Zero Router and zero error rate cascading router. The solid lines represent non-decreasing convex hull.

for certain real-world tasks to benefit from cascading routers might be challenging. Additionally, the sequence in which LLMs are chosen plays a crucial role in performance and offers room for optimization Chen et al. (2023). Our findings present a simulated upper limit for this method, highlighting the significant potential and the necessity of exploring the optimal implementation of cascading routers for specific applications.

5.4 RAG Results

Given previous results on predictive and cascading routers, we conduct a comparison of all router types simultaneously on RAG dataset. We use the same setting for KNN routers and MLP routers while selecting error rate 0.2 for cascading routers. We randomly partitioned RAG dataset into two splits, where the predictive routers are trained on 70% and both routers are evaluated on the rest 30%. As shown in Figure 9, all three routers significantly outperform the best individual LLM and surpass Zero Router. KNN routers, MLP routers, and cascading routers all exhibited higher AIQ scores than the zero router, showcasing the effectiveness of even basic routers in specific tasks such as retrieval-augmented generation.

6 Limitations and Future Work

MARScurrently only focuses on performance and economic cost. It is meaningful to include more evaluation criteria, such as latency, throughput and others to capture a more comprehensive understanding of router capabilities and limitations. There are also many LLMs and tasks that are not included in MARSdue to the limitation of time, and future iterations of this benchmark would include datasets that cover more tasks to effectively evaluate the ever-growing capability of LLMs, and also to add newer LLMs as they are being released.

Our current work only evaluates the efficacy of predictive and cascading routers, but there remains significant room for exploring additional router designs (as indicated in Section 5.3). It is important to dive into the exploration of more sophisticated router designs to further improve routing efficiency.

7 Conclusion

We present MARS, a benchmark specifically designed for the evaluation of router mechanisms within multi-LLM systems. By addressing the critical need for standardized evaluation in this domain, our benchmark provides a comprehensive dataset and a theoretical framework specifically designed for the nuanced analysis of router cost-efficiency and performance. The insights from our study shed light on the effectiveness of various routing strategies, emphasizing the necessity for advanced LLM routing systems and refined routing methods. This work establishes a robust and scalable benchmark for router evaluation and aims to facilitate future progress in the efficient and cost-effective deployment of Large Language Models.

References

Norah Alzahrani, Hisham Abdullah Alyahya, Yazeed Alnumay, Sultan Alrashed, Shaykhah Alsubaie, Yusef Almushaykeh, Faisal Mirza, Nouf Alotaibi, Nora Altwairesh, Areeb Alowisheq, M Saiful Bari, and Haidar Khan. When benchmarks are targets: Revealing the sensitivity of large language model leaderboards. *arXiv preprint arXiv: 2402.01781*, 2024.

Anthropic. Model card and evaluations for claude models, 2023. URL `https://www-cdn.anthropic.com/files/4zrzovbb/website/bd2a28d2535bfb0494cc8e2a3bf135d2e7523226.pdf`.

Jacob Austin, Augustus Odena, Maxwell Nye, Maarten Bosma, Henryk Michalewski, David Dohan, Ellen Jiang, Carrie Cai, Michael Terry, Quoc Le, and Charles Sutton. Program synthesis with large language models, 2021.

Sébastien Bubeck, Varun Chandrasekaran, Ronen Eldan, Johannes Gehrke, Eric Horvitz, Ece Kamar, Peter Lee, Yin Tat Lee, Yuanzhi Li, Scott Lundberg, et al. Sparks of artificial general intelligence: Early experiments with gpt-4. *arXiv preprint arXiv:2303.12712*, 2023.

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

Peter Clark, Isaac Cowhey, Oren Etzioni, Tushar Khot, Ashish Sabharwal, Carissa Schoenick, and Oyvind Tafjord. Think you have solved question answering? try arc, the ai2 reasoning challenge. *arXiv preprint arXiv:1803.05457*, 2018.

Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, Christopher Hesse, and John Schulman. Training verifiers to solve math word problems, 2021.

Surya Narayanan Hari and Matt Thomson. Tryage: Real-time, intelligent routing of user prompts to large language models, 2023.

Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. Measuring massive multitask language understanding, 2021.

Dongfu Jiang, Xiang Ren, and Bill Yuchen Lin. LLM-blender: Ensembling large language models with pairwise ranking and generative fusion. In Anna Rogers, Jordan Boyd-Graber, and Naoaki Okazaki (eds.), *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 14165–14178, Toronto, Canada, July 2023. Association for Computational Linguistics. doi: 10.18653/v1/2023.acl-long.792. URL `https://aclanthology.org/2023.acl-long.792`.

Sehoon Kim, Coleman Hooper, Amir Gholami, Zhen Dong, Xiuyu Li, Sheng Shen, Michael W. Mahoney, and Kurt Keutzer. Squeezellm: Dense-and-sparse quantization, 2023.

Woosuk Kwon, Zhuohan Li, Siyuan Zhuang, Ying Sheng, Lianmin Zheng, Cody Hao Yu, Joseph Gonzalez, Hao Zhang, and Ion Stoica. Efficient memory management for large language model serving with pagedattention. In *Proceedings of the 29th Symposium on Operating Systems Principles*, SOSP '23, pp. 611–626, 2023.

Chia-Hsuan Lee, Hao Cheng, and Mari Ostendorf. Orchestrallm: Efficient orchestration of language models for dialogue state tracking, 2023.

Hector Levesque, Ernest Davis, and Leora Morgenstern. The winograd schema challenge. In *Thirteenth international conference on the principles of knowledge representation and reasoning*, 2012.

Ji Lin, Jiaming Tang, Haotian Tang, Shang Yang, Xingyu Dang, Chuang Gan, and Song Han. Awq: Activation-aware weight quantization for llm compression and acceleration, 2023.

Keming Lu, Hongyi Yuan, Runji Lin, Junyang Lin, Zheng Yuan, Chang Zhou, and Jingren Zhou. Routing to the expert: Efficient reward-guided ensemble of large language models, 2023.

Xiaoding Lu, Adian Liusie, Vyas Raina, Yuwen Zhang, and William Beauchamp. Blending is all you need: Cheaper, better alternative to trillion-parameters llm, 2024.

Aman Madaan, Pranjal Aggarwal, Ankit Anand, Srividya Pranavi Potharaju, Swaroop Mishra, Pei Zhou, Aditya Gupta, Dheeraj Rajagopal, Karthik Kappaganthu, Yiming Yang, Shyam Upadhyay, Mausam, and Manaal Faruqui. Automix: Automatically mixing language models, 2023.

Rafael Rafailov, Archit Sharma, Eric Mitchell, Stefano Ermon, Christopher D. Manning, and Chelsea Finn. Direct preference optimization: Your language model is secretly a reward model, 2023.

Keisuke Sakaguchi, Ronan Le Bras, Chandra Bhagavatula, and Yejin Choi. Winogrande: An adversarial winograd schema challenge at scale. *Communications of the ACM*, 64(9):99–106, 2021.

Marija Šakota, Maxime Peyrard, and Robert West. Fly-swat or cannon? cost-effective language model choice via meta-modeling. *arXiv preprint arXiv:2308.06077*, 2023.

Bilgehan Sel, Ahmad Al-Tawaha, Vanshaj Khattar, Ruoxi Jia, and Ming Jin. Algorithm of thoughts: Enhancing exploration of ideas in large language models, 2023.

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

Rohan Taori, Ishaan Gulrajani, Tianyi Zhang, Yann Dubois, Xuechen Li, Carlos Guestrin, Percy Liang, and Tatsunori B Hashimoto. Stanford alpaca: An instruction-following llama model, 2023.

Gemini Team, Rohan Anil, Sebastian Borgeaud, Yonghui Wu, Jean-Baptiste Alayrac, Jiahui Yu, Radu Soricut, Johan Schalkwyk, Andrew M Dai, Anja Hauth, et al. Gemini: a family of highly capable multimodal models. *arXiv preprint arXiv:2312.11805*, 2023.

Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, Dan Bikel, Lukas Blecher, Cristian Canton Ferrer, Moya Chen, Guillem Cucurull, David Esiobu, Jude Fernandes, Jeremy Fu, Wenyin Fu, Brian Fuller, Cynthia Gao, Vedanuj Goswami, Naman Goyal, Anthony Hartshorn, Saghar Hosseini, Rui Hou, Hakan Inan, Marcin Kardas, Viktor Kerkez, Madian Khabsa, Isabel Kloumann, Artem Korenev, Punit Singh Koura, Marie-Anne Lachaux, Thibaut Lavril, Jenya Lee, Diana Liskovich, Yinghai Lu, Yuning Mao, Xavier Martinet, Todor Mihaylov, Pushkar Mishra, Igor Molybog, Yixin Nie, Andrew Poulton, Jeremy Reizenstein, Rashi Rungta, Kalyan Saladi, Alan Schelten, Ruan Silva, Eric Michael Smith, Ranjan Subramanian, Xiaoqing Ellen Tan, Binh Tang, Ross Taylor, Adina Williams, Jian Xiang Kuan, Puxin Xu, Zheng Yan, Iliyan Zarov, Yuchen Zhang, Angela Fan, Melanie Kambadur, Sharan Narang, Aurelien Rodriguez, Robert Stojnic, Sergey Edunov, and Thomas Scialom. Llama 2: Open foundation and fine-tuned chat models, 2023.

Xuezhi Wang, Jason Wei, Dale Schuurmans, Quoc Le, Ed Chi, Sharan Narang, Aakanksha Chowdhery, and Denny Zhou. Self-consistency improves chain of thought reasoning in language models, 2023a.

Yiding Wang, Kai Chen, Haisheng Tan, and Kun Guo. Tabi: An efficient multi-level inference system for large language models. In *Proceedings of the Eighteenth European Conference on Computer Systems*, EuroSys '23, pp. 233–248, New York, NY, USA, 2023b. Association for Computing Machinery. ISBN 9781450394871. doi: 10.1145/3552326.3587438. URL `https://doi.org/10.1145/3552326.3587438`.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Brian Ichter, Fei Xia, Ed Chi, Quoc Le, and Denny Zhou. Chain-of-thought prompting elicits reasoning in large language models, 2023.

Shunyu Yao, Dian Yu, Jeffrey Zhao, Izhak Shafran, Thomas L. Griffiths, Yuan Cao, and Karthik Narasimhan. Tree of thoughts: Deliberate problem solving with large language models, 2023.

Murong Yue, Jie Zhao, Min Zhang, Liang Du, and Ziyu Yao. Large language model cascades with mixture of thoughts representations for cost-efficient reasoning, 2023.

Rowan Zellers, Ari Holtzman, Yonatan Bisk, Ali Farhadi, and Yejin Choi. Hellaswag: Can a machine really finish your sentence?, 2019.

Chujie Zheng, Hao Zhou, Fandong Meng, Jie Zhou, and Minlie Huang. Large language models are not robust multiple choice selectors. In *The Twelfth International Conference on Learning Representations*, 2024. URL `https://openreview.net/forum?id=shr9PXz7T0`.

Lianmin Zheng, Wei-Lin Chiang, Ying Sheng, Siyuan Zhuang, Zhanghao Wu, Yonghao Zhuang, Zi Lin, Zhuohan Li, Dacheng Li, Eric Xing, et al. Judging llm-as-a-judge with mt-bench and chatbot arena. *arXiv preprint arXiv:2306.05685*, 2023a.

Lianmin Zheng, Wei-Lin Chiang, Ying Sheng, Siyuan Zhuang, Zhanghao Wu, Yonghao Zhuang, Zi Lin, Zhuohan Li, Dacheng Li, Eric P. Xing, Hao Zhang, Joseph E. Gonzalez, and Ion Stoica. Judging llm-as-a-judge with mt-bench and chatbot arena, 2023b.

Denny Zhou, Nathanael Schärli, Le Hou, Jason Wei, Nathan Scales, Xuezhi Wang, Dale Schuurmans, Claire Cui, Olivier Bousquet, Quoc Le, and Ed Chi. Least-to-most prompting enables complex reasoning in large language models, 2023.

A Additional Dataset Details

A.1 Model Details & Cost Estimation

For all proprietary models, we calculate the cost of input and output results based on their API pricing. For open-source models, we utilize Together AI [4] to obtain results and reference costs. For the RAG experiment, we refer to the API pricing of You.com [5] and Perplexity [6] for cost estimation.

A.2 Dataset Generation for RAG

Rationale behind choosing topics for RAG dataset

- **One year ago public information, Year 2023** Some of the most up-to-date LLMs might have been trained with such web content, while others might not. This section helps qualify LLMs among each-other, rewarding the ones that are more recently and more comprehensively trained.
- **Three years ago public information, Year 2020** Time sensitive public information that was available during the training of all modern LLMs. This section helps qualify LLMs according to the comprehensiveness of their training method, as well as their ability to absorb information during training, as compared to web connected systems which have dynamic access to such information.
- **Specific, not too time sensitive, public information** The content needed to answer those questions is not time sensitive, yet is very specific. For example “What is the average cost of a meal in a mid-range restaurant in Barcelona, Spain?”. This section helps quantify the advantage of bigger LLMs over smaller ones, in their ability to retain fine grained information about the world during training, and compare them to the information-retrieval power of web connected systems.

A.3 Dataset Details

MMLU Hendrycks et al. (2021): A benchmark that measures the knowledge acquired by models during pretraining and evaluates models in zero-shot and few-shot settings across 57 tasks, testing both knowledge and reasoning on different fields of human knowledge.

Hellaswag Zellers et al. (2019): A dataset that challenges models to pick the best ending choice to a sentence given. It uses Adversarial Filtering(AF) to create a Goldilocks zone of complexity, wherein generations are largely nonsensical to humans but always make models struggle.

GSM8K Cobbe et al. (2021): A dataset of diverse grade school math word problems, testing a model’s ability to perform multi-step mathematical reasoning.

ARC Challenge Clark et al. (2018) A rigorous question answering dataset, ARC-Challenge includes complex, different grade-school level questions that require reasoning beyond simple retrieval, testing the true comprehension capabilities of models. Arc Challenge dataset contains those that both a retrieval and a co-occurrence method fail to answer correctly)

Winogrande Sakaguchi et al. (2021): A large-scale and increased harness dataset inspired by the original Winograd Schema Challenge(WSC) Levesque et al. (2012) tests models on their ability to resolve pronoun ambiguity and their ability to understand the context with commonsense knowledge.

MBPP Austin et al. (2021): The benchmark is designed to be solvable by entry level programmers, covering programming fundamentals, standard library functionality, and so on. Each problem consists of a task description, code solution and 3 automated test cases.

MT-Bench Zheng et al. (2023b): This dataset contains 3.3K expert-level pairwise human preferences for model responses generated by 6 models in response to 80 MT-bench questions, multi-run

[4] https://www.together.ai/pricing

[5] https://api.you.com/

[6] https://docs.perplexity.ai/docs/pricing

QA. The 6 models are GPT-4, GPT-3.5[7], Claude-v1, Vicuna-13B Zheng et al. (2023a), Alpaca-13B Taori et al. (2023), and LLaMA-13B Touvron et al. (2023). The annotators are mostly graduate students with expertise in the topic areas of each of the questions.

A.4 More Details on Dataset Construction

Each sample in the benchmark dataset will have the following attributes:

- $sample\_id$: contain the information about the name of the sub-task, the split of dataset, and the index of the data in that dataset. Example: **mmlu-astronomy.val.5**
- $model\_name$: the model used to perform inference for this sample. Example: **GPT-4**
- $eval\_name$: the source data this specific sample comes from. Example: **hellaswag.dev.v0**
- $prompt$: prompt sentence. Example: **The following are multiple choice questions...**
- $model\_response$: Model's output. Example: **The answer is A)**
- $performance$: the result compared to true label. Example: **True/False**
- $cost$: for proprietary model, we use API cost to calculate; for open source model, we use Together AI[8] to call the model and use their cost as reference. Example: **0.00019**
- $true\_label$: the true label or gold response for this prompt. Example: **True/False**

A.5 Evaluation Metrics

We will perform 5-shot inference on **MMLU**, **HellaSwag**, **GSM8K**, **ARC Challenge**, **Winogrande** and 0-shot inference on **MBPP**, **MT-Bench** and **RAG**.

For the datasets **MMLU**, **HellaSwag**, **GSM8K**, **ARC Challenge**, and **Winogrande**, we use the exact match method to compute the final results. In contrast, for **MBPP**, **MT-Bench**, and **RAG**, we use GPT-4 for answer evaluation. Results categorized as False/True are converted to a binary 0/1 format. In cases where the results are based on ratings, we normalize all outcomes to a [0, 1] scale.

A.6 Individual Dataset Result

The MARS pilot study result has been shown in Fig 3. Here are the breakdown of each dataset in MARS.

B Issues with Overly-aligned Models

Some models exhibit reluctance in responding to certain inputs, often replying with statements like ”I do not understand...” or ”I am not sure about...”. We have identified two primary reasons for models' refusal to respond:

Insufficient Context Perception Despite being provided with enough context, these models perceive the information as inadequate. Our hypothesis is that the models' capabilities might not be robust enough to generate answers or perform tasks effectively under these conditions. A potential remedy is to modify the prompting strategy to encourage output generation.

Uncertainty Avoidance Some models appear to be fine-tuned to function as 'safe' assistants, refraining from providing responses when they lack certainty. This cautious approach likely aims to prevent potential errors stemming from uncertain answers. Claude 2 exhibits this behavior most frequently.

LLMs have been known to have such kind of issues as documented in various previous studies Zheng et al. (2024); Alzahrani et al. (2024). It is essential to develop methods to make LLM outputs more controllable and structured when routing, which warrants further exploration in future research.

[7] https://openai.com/blog/chatgpt

[8] https://www.together.ai/

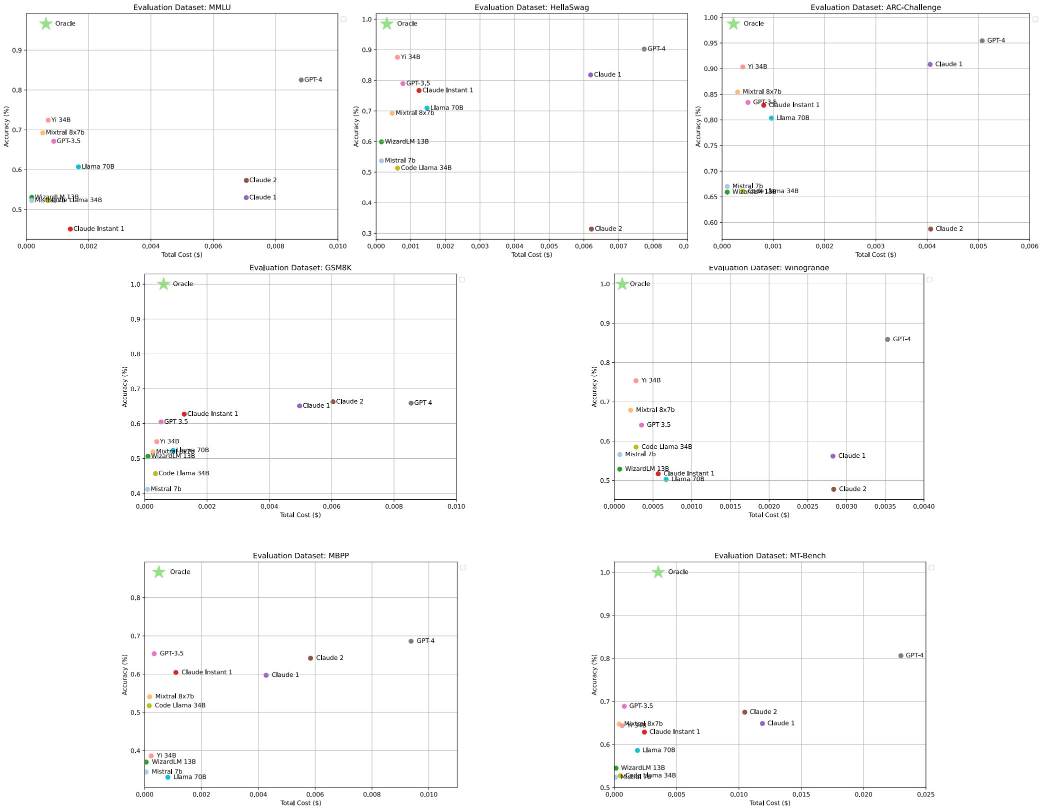

Figure 6: Accuracy vs Total cost of each LLM on each sub dataset in MARS.

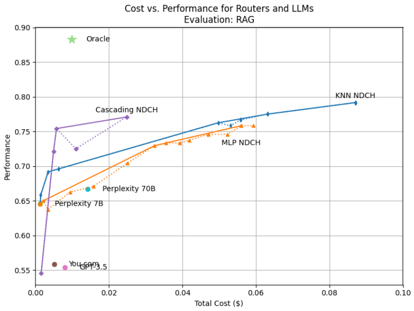

Figure 7: Zoom in version of Performance vs Total Cost of each LLM on **RAG**. Full version in Fig 9

C Training Data Distribution

We also conduct Out-domain experiments where we train on held-out tasks in MARSfor each dataset and evaluate on MT-Bench, MBPP and GSM8K in Figure 8. It shows the router designs in this work are still effective in this out-domain setting.

D RAG Result

RAG result for Section 5.4 is in Figure 9.

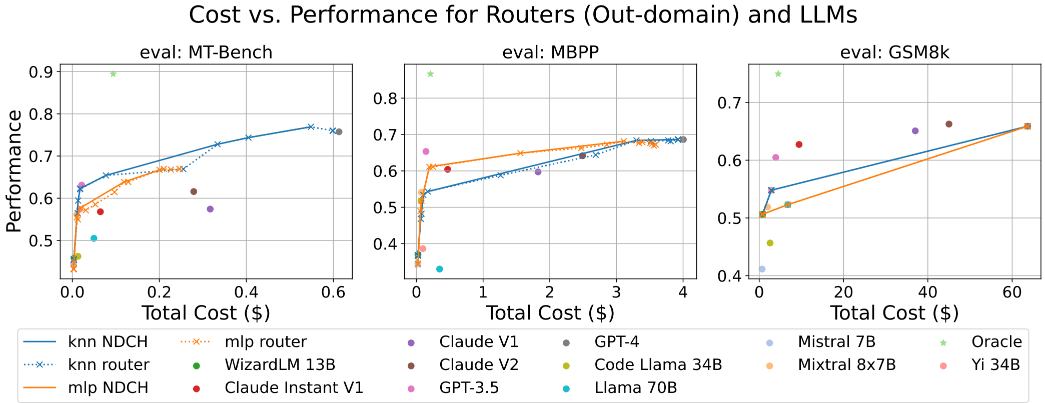

Figure 8: Total Cost vs Performance for eleven models and KNN, MLP routers on MT-Bench, MBPP, GSM8K. NDCH stands for non-decreasing convex hull

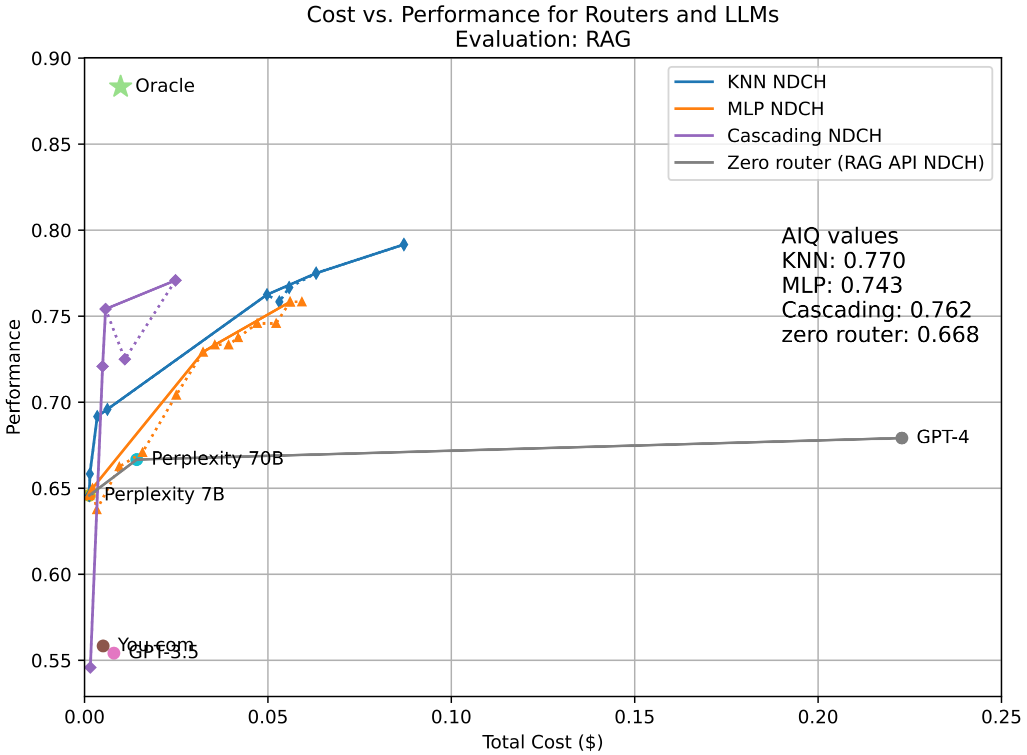

Figure 9: Total Cost vs Performance for five models and four routers on RAG dataset. The AIQ values are also calculated for all four routers. NDCH represents non-decreasing convex hull. Zoom-in version is in Figure 7