RAGRouter: Learning to Route Queries to Multiple Retrieval-Augmented Language Models

Jiarui Zhang[1] **Xiangyu Liu**[2] **Yong Hu**[2] **Chaoyue Niu**[1*] **Fan Wu**[1] **Guihai Chen**[1]
[1]Shanghai Jiao Tong University [2]WeChat, Tencent Inc

Abstract

Retrieval-Augmented Generation (RAG) significantly improves the performance of Large Language Models (LLMs) on knowledge-intensive tasks. However, varying response quality across LLMs under RAG necessitates intelligent routing mechanisms, which select the most suitable model for each query from multiple retrieval-augmented LLMs via a dedicated router model. We observe that external documents dynamically affect LLMs' ability to answer queries, while existing routing methods, which rely on static parametric knowledge representations, exhibit suboptimal performance in RAG scenarios. To address this, we formally define the new retrieval-augmented LLM routing problem, incorporating the influence of retrieved documents into the routing framework. We propose RAGRouter, a RAG-aware routing design, which leverages document embeddings and RAG capability embeddings with contrastive learning to capture knowledge representation shifts and enable informed routing decisions. Extensive experiments on diverse knowledge-intensive tasks and retrieval settings, covering open and closed-source LLMs, show that RAGRouter outperforms the best individual LLM and existing routing methods. With an extended score-threshold-based mechanism, it also achieves strong performance-efficiency trade-offs under low-latency constraints. [2]

1 Introduction

The rapid advancement of large language models (LLMs) has led to an increasingly diverse model landscape, with significant heterogeneity in parametric knowledge stemming from variations in training data, architectures, and learning objectives [45, 35, 36, 3, 4, 57, 53]. However, LLMs remain limited by outdated knowledge, hallucinations, and insufficient domain coverage [17, 22, 29]. Retrieval-Augmented Generation (RAG) [27, 15] addresses these issues by injecting external knowledge at inference time, effectively reducing hallucinations and improving performance on knowledge-intensive tasks [5, 13, 47, 20, 41].

While RAG enhances LLM performance by incorporating external knowledge, different LLMs exhibit substantial variation in their ability to utilize retrieved content. Prior studies [6, 11] show that, given identical documents, LLMs differ in information extraction, integration, and robustness to noise—reflecting inherent heterogeneity in RAG capabilities stemming from differences in architecture, training data, and optimization. Such diversity suggests that combining multiple models can yield complementary strengths, enabling performance that surpasses any single model. LLM routing, which pre-selects the most suitable model for each query from a pool of LLMs without invoking them all, offers an efficient and effective fusion strategy [9]. Under dual heterogeneity in parametric knowledge and RAG capability, intelligent query routing presents a promising direction for leveraging RAG to achieve superior performance, motivating the central question: *How can we effectively route queries to the most capable LLM under the RAG paradigm?*

[*]Chaoyue Niu is the corresponding author (rvince@sjtu.edu.cn).
[2]The code and data are available at `https://github.com/OwwO99/RAGRouter`.

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

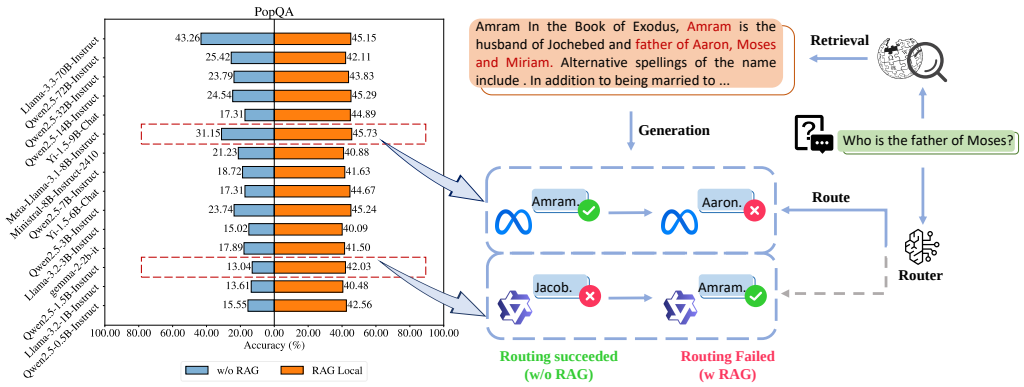

Figure 1: **Left**: Accuracy of various LLMs on the PopQA task before and after RAG. **Right**: An example query where retrieved documents improve Qwen's response (unanswerable → answerable) but impair Llama's (answerable → unanswerable), illustrating how existing routing methods fail under RAG due to their inability to capture such dynamic shifts.

Current multi-model routing methods primarily match queries to LLMs based on their inherent parametric knowledge [40, 19, 31, 43, 38, 10, 7, 33, 58, 12]. Several approaches [7, 33, 58, 12] construct compact vector representations of LLMs to enable efficient query-model compatibility estimation. These methods all assume static knowledge representations for non-RAG scenarios.

However, these approaches face critical limitations in RAG settings, as they fail to account for the dynamic impact of knowledge injection. As illustrated in Figure 1, RAG dramatically shifts the distribution of response quality across input queries—external documents can reverse a model's ability to answer a question, rendering routing strategies designed for non-RAG scenarios obsolete. The core issue lies in the **Static Knowledge Assumption**: existing approaches assume fixed LLM knowledge, ignoring how retrieved content dynamically reshapes their capabilities. In practice, RAG response quality depends on the interplay between a model's internal knowledge and external information. This leads to shortcomings: **Missing Doc Interaction**—existing methods focus on queries and model embeddings, overlooking document features and their interaction with models; and **Ignoring RAG Capability**—prior work captures only static knowledge differences, neglecting LLMs' differing ability to leverage documents. These gaps highlight shortcomings of current LLM routing strategies—they fail to adapt to the dynamic relationship between LLM and external knowledge.

To tackle this issue, we propose **RAGRouter**, a contrastive learning-based routing framework that explicitly models knowledge shifts in RAG scenarios. RAGRouter is designed to route queries across LLMs by modeling key factors that affect post-retrieval performance. At the **architecture level**, RAGRouter incorporates a document encoder and a cross encoder to capture document semantics and query interactions, thereby addressing missing document interaction, and assigns each LLM a RAG capability embedding—a learnable vector representing its proficiency in utilizing retrieved content—to mitigate ignoring RAG capability. However, directly optimizing such a router is challenging due to inherent variations introduced by retrieval. To address this, at the **optimization level**, we employ a contrastive learning objective, where positive and negative samples—i.e., representations of LLMs that correctly or incorrectly respond to a query—are drawn from both *Cross-Setting* (between non-RAG and RAG settings) and *Intra-Setting* (within each setting). Taking the query representation as an anchor, the objective encourages alignment between answerable model-query pairs while pushing apart unanswerable ones. This allows RAGRouter to effectively model retrieval-induced behavior shifts, moving beyond the static knowledge assumption.

We evaluate RAGRouter on a suite of knowledge-intensive tasks [32, 34, 25, 2, 23] and retrieval settings. Experimental results show that RAGRouter surpasses the performance of the best individual LLM, highlighting its ability to leverage the complementary strengths of multiple models in retrieval-augmented scenarios. Furthermore, RAGRouter substantially outperforms existing non-RAG-aware routing methods, validating the effectiveness of modeling retrieval-induced knowledge shifts.

Our main contributions are summarized as follows: (i) To the best of our knowledge, this is the first work exploring LLM routing in the RAG setting; (ii) We propose RAGRouter, a contrastive learning-based routing mechanism that is aware of knowledge shifts, incorporating RAG capability and document-aware representations to effectively address the failure modes of existing routing strategies in RAG; (iii) We validate the effectiveness of our method on five knowledge-intensive tasks

under local and online retrieval settings, using open-source LLMs such as the Qwen and LLaMA series scales from 0.5B to 72B and closed-source LLMs including GPT-4o, Qwen2.5-Max, and DeepSeek-R1. Results show that RAGRouter outperforms the best individual LLM and existing non-RAG-aware routing methods by 1.67%–9.33%; (iv) We apply an extended score-threshold-based mechanism to RAGRouter, and results show that its accuracy–latency curve generally lies above those of all baselines, indicating superior performance-efficiency trade-offs under low-latency constraints.

2 Related Work

Retrieval-Augmented Generation. RAG enhances language models by integrating retrieved information from external databases [27, 15]. It typically follows a round of retrieval and sequential generation pipelines, where documents are retrieved based on the input query and concatenated with it for generation. Prior work has improved RAG by optimizing retrieval components [56, 51, 42] or enhancing the generator's ability to utilize retrieved content [21, 49, 48]. Recent studies highlight heterogeneous LM capabilities in processing external information, both in utilizing retrieved content [28, 6] and tolerating retrieval noise [11, 39]. These heterogeneous capabilities reveal optimization opportunities for ensemble approaches that strategically leverage multiple LLMs within RAG scenarios. In this work, we study the routing problem under the RAG setting.

LLM Routing. Existing LLM routing approaches [7, 10, 12, 33, 58, 31] primarily focus on non-RAG settings, where routing relies solely on the input query and each model's parametric knowledge, without incorporating external retrieved documents. For example, RouterDC [7] uses dual contrastive learning to model query-model compatibility, while EmbedLLM [58] and RouteLLM [33] apply matrix factorization to learn compact model embeddings for scalable routing. GraphRouter [12] constructs a heterogeneous graph with nodes for tasks, queries, and LLMs, and encodes their interactions as edges to capture contextual alignment between query needs and model capabilities. However, in RAG scenarios, retrieved documents induce dynamic shifts in model knowledge, which existing methods overlook. In contrast, our proposed RAGRouter models both the documents and LLMs' RAG capabilities, enabling more effective routing under retrieval-augmented settings.

3 Problem Formulation

RAG enhances LLMs by integrating external knowledge through a two-stage process: given a query q, the retriever $\text{Ret}(\mathcal{D}, q)$ selects relevant documents d from an external corpus $\mathcal{D}$, and the model $M(q, d)$ generates a response y based on both the query q and the documents d, i.e., $y = M(q, \text{Ret}(\mathcal{D}, q))$.

We formulate a LLM routing problem under RAG setting. Let $\mathcal{M} = \{M_1, \ldots, M_N\}$ be a set of candidate LLMs. A routing policy $R : \mathcal{Q} \times \mathcal{D} \to \{1, \ldots, N\}$ selects the most suitable model $M_{R(q,d)}$ for each input pair (q, d). To evaluate response quality, we define an oracle scoring function $\sigma(M_i, q, d) \in \{0, 1\}$, where $\sigma(M_i, q, d) = 1$ if the response from M_i given q and d matches the reference answer y^*. Importantly, using a fixed model can be suboptimal, as different LLMs excel on different query-document pairs. The objective is to maximize the expected routing performance:

$$\max_R \mathbb{E}_{q \sim \mathcal{Q}} \left[\sigma(M_{R(q,d)}, q, d) \right] \tag{1}$$

Notably, when no external documents are available (i.e., $d = \emptyset$), the LLM routing problem under RAG setting naturally degenerates into the conventional LLM routing problem. In this setting, the routing policy simplifies to $R : \mathcal{Q} \to \{1, \ldots, N\}$, and the oracle scoring function becomes $\sigma(M_i, q)$, which assesses the response based solely on the query. The objective becomes:

$$\max_R \mathbb{E}_{q \sim \mathcal{Q}} \left[\sigma(M_{R(q)}, q) \right] \tag{2}$$

Thus, the conventional routing problem can be seen as a special case of LLM routing under the RAG setting, corresponding to the boundary condition where $d = \emptyset$.

4 RAGRouter

4.1 Routing Model Architecture Design

We establish a conceptual framework by constructing an intuitive explanation of knowledge representation and LLM-query matching under RAG and non-RAG settings. In non-RAG settings

[7, 58], each LLM is typically associated with a compact knowledge representation vector $v_k \in \mathbb{R}^{\text{dim}}$, which implicitly reflects its parametric knowledge; and meanwhile, a query is encoded as $v_q \in \mathbb{R}^{\text{dim}}$, representing the knowledge needed to answer it. A proxy metric, like similarity $\text{sim}(v_q, v_k)$ is then used to gauge the LLM's ability to respond, guiding non-RAG routing process.

However, in RAG settings, the LLM is augmented with retrieved documents that provide non-parametric knowledge. This additional information influences the model's response generation, rendering the original knowledge representation v_k insufficient. The effective knowledge of the LLM shifts due to the integration of external information, resulting in a new representation:

$$v'_k = v_k + v_f \tag{3}$$

where v_f is the fused knowledge representation derived from the documents. Consequently, in RAG scenarios, the similarity between the query and the updated knowledge representation $\text{sim}(v_q, v'_k)$ should serve as the new routing criterion, as it more accurately reflects the model's ability to respond.

RAGRouter is designed with this insight in mind and explicitly models the fused knowledge v_f to dynamically update the LLM's knowledge representation. We identify three core factors that influence v_f: (1) the non-parametric knowledge provided by the documents; (2) the LLM's ability to process external information, including knowledge extraction and robustness to noise; and (3) the query's role in guiding knowledge retrieval. Based on these, RAGRouter consists of the following modules, with its architecture illustrated in Figure 2.

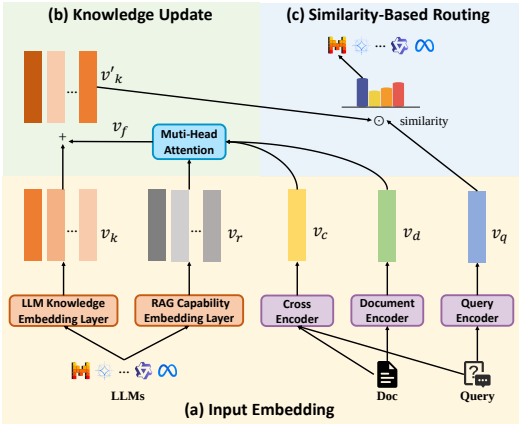

Figure 2: The inference pipeline of RAGRouter: (a) Encode query, document, cross interaction, LLM knowledge, and RAG capability; (b) Fuse RAG capability, document, and cross embeddings to update knowledge representation; (c) Route based on similarity with the query embedding.

Representing Parametric Knowledge. To obtain the original parametric knowledge representation v_k, we introduce the LLM Knowledge Embedding Layer ϕ_K, which takes the LLM ID M and outputs $v_k = \phi_K(M)$, capturing inter-model variability in parametric knowledge. For query representation, we employ a Query Encoder ϕ_Q, which encodes the query q as $v_q = \phi_Q(q)$.

Representing RAG-Aware Factors. To compute the fused knowledge v_f, RAGRouter integrates signals from three perspectives. First, **the non-parametric knowledge provided by the documents** is captured by the **Document Encoder** ϕ_D, which encodes a document d into $v_d = \phi_D(d)$. In practice, the document and query encoders share parameters to ensure consistency in the embedding space. Second, **the LLM's ability to process external information** is captured by the **RAG Capability Embedding Layer** ϕ_R, which maps each candidate model M to an embedding $v_r = \phi_R(M)$, representing its intrinsic capacity to utilize retrieved evidence. Third, **the query's role in guiding knowledge retrieval** is represented by the **Cross Encoder** ϕ_C, which processes the query-document pair (d, q) to produce an interaction representation $v_c = \phi_C(d, q)$.

Representation Update for Similarity-Based Routing. The fused knowledge representation v_f is then derived via a multi-head attention mechanism that integrates these signals:

$$v_f = \text{Attention}(v_r, v_d, v_c) \tag{4}$$

With the fused knowledge computed, the RAG-aware knowledge representation of the model becomes $v'_k = v_k + v_f$. The final routing decision is based on the similarity between the query and the updated knowledge representations of candidate models:

$$R(q, d) = \arg \max_{i \in \{1,\ldots,N\}} \{\text{sim}(v_q, v'_{k_i})\} \tag{5}$$

This formulation allows the routing policy to explicitly account for knowledge shifts introduced by document retrieval, thus maintaining accurate assessment of each LLM's ability in RAG settings.

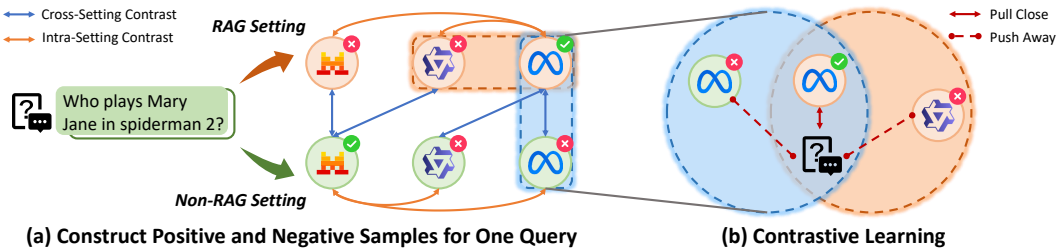

Figure 3: (a) CSC constructs positive and negative samples under different settings based on response quality (e.g., Llama w/ RAG (✓) vs. Llama w/o RAG (✗)), while ISC constructs them under the same setting (e.g., Llama w/ RAG (✓) vs. Qwen w/ RAG (✗)); (b) By combining CSC and ISC, contrastive learning pulls positive samples closer to the query representation and pushes negative ones away.

4.2 Optimization

In RAG settings, the incorporation of retrieved documents often leads to significant changes in LLM answerability—some LLMs become able to answer queries they previously could not, while others fail after retrieval. These shifts in answerability, effectively label transitions, reflect corresponding changes in the model's knowledge representation. Such transitions naturally yield structured positive and negative pairs across different knowledge states. This setting aligns well with the principles of contrastive learning [8, 18], which is particularly well-suited for capturing and optimizing the knowledge representation shifts induced by external knowledge injection in RAGRouter.

To this end, we design the **Cross-Setting Contrast (CSC)** mechanism to model representation differences between the non-RAG and RAG settings, and introduce the **Intra-Setting Contrast (ISC)** mechanism to model representation differences within the same setting. As shown in Figure 3, using the query representation v_q as the anchor, CSC constructs positive and negative samples by selecting knowledge representations with different response qualities from the non-RAG and RAG settings (blue arrows). ISC, on the other hand, selects positive and negative samples from models with different response qualities within the same setting (orange arrows). This enables CSC to help RAGRouter distinguish between different knowledge transfer patterns induced by documents, while ISC enhances the model's discriminative ability across LLMs within the same setting.

Combining CSC and ISC, we construct a comprehensive set of positive and negative samples to train the RAGRouter. For a given query q, we define the positive and negative sets as follows:

$$\begin{cases} V_+ = \{v_{k_i} \mid \sigma(M_i, q) = 1\} \cup \{v'_{k_j} \mid \sigma(M_j, d, q) = 1\} \\ V_- = \{v_{k_i} \mid \sigma(M_i, q) = 0\} \cup \{v'_{k_j} \mid \sigma(M_j, d, q) = 0\} \end{cases} \tag{6}$$

The corresponding contrastive loss is defined as:

$$\mathcal{L}_{CT}(q) = \sum_{v_{k+} \in V_+} -\log \frac{\exp(\text{sim}(v_q, v_{k+})/\tau)}{\exp(\text{sim}(v_q, v_{k+})/\tau) + \sum_{v_{k-} \in V_-} \exp(\text{sim}(v_q, v_{k-})/\tau)} \tag{7}$$

where τ is a temperature hyperparameter. This loss encourages the query embedding v_q to be closer to positive samples and further from negative ones, enabling the learning of representations that are sensitive to both knowledge shifts and model heterogeneity. When retrieved documents alter a model's response ability—e.g., from unanswerable to answerable—the mechanism captures these dynamic transitions, enhancing routing accuracy and knowledge adaptability in RAGRouter.

To further enhance LLM discrimination, we introduce a binary classification loss. For the original model M, define $s_{M,q} = \text{Sigmoid}(\text{sim}(v_k, v_q))$; for the RAG-enhanced model M', define $s_{M',q} = \text{Sigmoid}(\text{sim}(v'_k, v_q))$. Let $y_{M,q} = \sigma(M, q)$ and $y_{M',q} = \sigma(M, d, q)$ be the ground-truth labels. The classification loss is:

$$\mathcal{L}_{CLS}(q) = -\sum_{M \in \mathcal{M} \cup \mathcal{M}'} [y_{M,q} \log s_{M,q} + (1 - y_{M,q}) \log(1 - s_{M,q})] \tag{8}$$

The total loss is the weighted sum of the contrastive loss and classification loss, with $\lambda > 0$ as a balancing hyperparameter:

$$\mathcal{L}(q) = \mathcal{L}_{CT}(q) + \lambda \mathcal{L}_{CLS}(q) \tag{9}$$

4.3 Latency-Aware Extended Design

While RAGRouter does not explicitly model LLM's latency, it outputs a relevance score for each candidate LLM given a query, which can be exploited to support flexible trade-offs between performance and efficiency. To this end, we introduce a score-threshold-based routing mechanism. Concretely, we pre-sort the N available LLMs as $[M_1, M_2, \ldots, M_N]$ based on their prior efficiency profiles, such as smaller parameter sizes and lower latency—meaning that M_1 is the most efficient and M_N the least.

Given a query, suppose M_i receives the highest predicted score from RAGRouter (i.e., it is the performance-optimal model). Instead of routing directly to M_i, we traverse the list from M_1 to M_i and select the first LLM M_j whose score satisfies $s_{M_i,q} - s_{M_j,q} \leq \theta$, where θ is a user-defined score margin threshold. This mechanism sacrifices a small amount of accuracy for significantly improved efficiency, making RAGRouter adaptable to latency-constrained or resource-limited scenarios.

5 Experiments

5.1 Experimental Setup

Datasets. We select queries from five different knowledge-intensive tasks: (i) PopQA [32] is an open-domain question-answering benchmark covering diverse factual topics from broad knowledge domains; (ii) MedMCQA [34] is a multiple-choice benchmark focused on biomedical knowledge and clinical reasoning; (iii) Natural Questions (NQ) [25] is an open-domain benchmark based on real-world search queries requiring span-level answer retrieval from Wikipedia; (iv) WebQuestions (WebQ) [2] is a knowledge base-driven benchmark grounded in Freebase relations, designed to evaluate entity-centric factual reasoning; and (v) TriviaQA (TQA) [23] is an open-domain benchmark centered on factoid-style questions sourced from trivia enthusiasts and web documents. Following [42], we adopt Cover Exact Match as the evaluation metric for PopQA, NQ, WebQ, and TriviaQA. Further data processing details and dataset statistics are summarized in Appendix A.2.

Candidate LLMs. We selected 15 mainstream LLMs [52, 14, 44, 55, 1] with parameter size ranging from 0.5B to 72B. Comprehensive statistics on model scales and latency [3] are presented in Table 1, and implementation details are provided in Appendix A.1.

Table 1: Statistics of different LLMs and their latency.

LLM	Params (B)	Latency (ms)
Qwen2.5-0.5B-Instruct	0.494	24.54
Llama-3.2-1B-Instruct	1.240	20.47
Qwen2.5-1.5B-Instruct	1.500	24.79
gemma-2-2b-it	2.614	31.80
Llama-3.2-3B-Instruct	3.213	81.82
Qwen2.5-3B-Instruct	3.000	24.39
Yi-1.5-6B-Chat	6.061	142.67
Qwen2.5-7B-Instruct	7.616	80.83
Ministral-8B-Instruct-2410	8.020	26.13
Meta-Llama-3.1-8B-Instruct	8.030	177.37
Yi-1.5-9B-Chat	8.829	199.61
Qwen2.5-14B-Instruct	14.770	175.42
Qwen2.5-32B-Instruct	32.764	156.26
Qwen2.5-72B-Instruct	72.706	1610.00
Llama-3.3-70B-Instruct	70.554	1970.00

Retrieval Settings. Following [42], we adopt both local and online retrieval strategies for PopQA and MedMCQA to reflect realistic RAG scenarios. Local retrieval uses the 2018 English Wikipedia dump [24] with BGE-large-en-v1.5 [50] as the dense retriever. Online retrieval leverages the DuckDuckGo Web Search API [4] to access up-to-date external content. For NQ, WebQ, and TriviaQA, we follow [11] and construct retrieval contexts from Wikipedia passages augmented with synthetic noise (e.g., irrelevant distractors, counterfactual noise) to simulate imperfect retrieval. This setting enables evaluate the effectiveness of the routing model under noisy conditions.

Baselines. We compare RAGRouter against a range of baselines. Existing routing methods were not RAG-aware and exploited only query-LLM compatibility, ignoring the impact of retrieval augmentation. This series of baselines include **Prompt LLM** [12], which employs GPT-4o for model selection via meta-prompts; **GraphRouter** [12], which models queries, tasks, and LLMs in a heterogeneous graph; **RouterDC** [7], which aligns query-LLM embeddings through contrastive learning; **KNN Router**, which [19] relies on historical performance of similar queries; and **Matrix Factorization (MF)** [33, 58], which reconstructs LLM correctness patterns via low-rank latent spaces. We also introduce some rule-based routing methods, including a **Single Fixed LLM** for all queries; **Oracle Single Best** that ideally selects the best-performing single LLM per dataset; **Random** LLM

[3] Latency refers to the average time taken by an LLM to complete a single query, including both inference time and potential network delays.

[4] `https://duckduckgo.com`

Table 2: Performance comparison of RAGRouter with rule-based and non-RAG-aware baselines across different knowledge-intensive tasks and retrieval settings. Testing accuracy (%) is reported. "Ret." indicates whether the method is retrieval-aware (✓) or not (✗). The best results are shown in **bold**, and the second-best are underlined.

Method	Ret.	PopQA		MedMCQA		NQ	WQ	TQA	Avg
		Local	Online	Local	Online				
Qwen2.5-0.5B-Instruct	-	45.19	51.11	25.93	34.81	27.08	38.75	39.17	37.43
Llama-3.2-1B-Instruct	-	39.26	46.67	17.78	36.67	25.42	31.67	43.33	34.40
Qwen2.5-1.5B-Instruct	-	44.44	48.89	30.00	42.59	30.00	35.42	46.67	39.72
gemma-2-2b-it	-	41.11	50.74	20.37	37.04	22.92	27.92	40.83	34.42
Llama-3.2-3B-Instruct	-	45.56	51.48	27.41	44.81	36.67	45.42	65.00	45.19
Qwen2.5-3B-Instruct	-	41.11	47.41	40.37	49.26	30.42	36.25	53.33	42.59
Yi-1.5-6B-Chat	-	46.67	51.48	31.11	40.00	35.83	43.33	56.67	43.58
Qwen2.5-7B-Instruct	-	42.96	48.15	35.93	43.33	29.58	35.42	55.00	41.48
Ministral-8B-Instruct-2410	-	41.48	46.30	50.74	62.22	38.33	42.08	63.75	49.27
Meta-Llama-3.1-8B-Instruct	-	46.67	51.85	41.85	52.59	39.58	46.25	69.58	49.77
Yi-1.5-9B-Chat	-	46.67	**52.59**	50.74	57.78	38.33	47.08	58.75	50.28
Qwen2.5-14B-Instruct	-	46.30	50.00	57.04	64.07	42.92	47.08	72.50	54.27
Qwen2.5-32B-Instruct	-	45.93	48.52	43.33	49.63	44.58	50.42	80.42	51.83
Qwen2.5-72B-Instruct	-	44.81	47.78	67.04	70.00	40.00	48.75	79.17	56.79
Llama-3.3-70B-Instruct	-	46.30	50.37	68.89	70.37	51.67	50.42	87.92	60.85
Oracle Single Best	-	46.67	**52.59**	68.89	70.37	51.67	50.42	87.92	61.22
Random	-	44.30	49.56	40.57	50.35	35.56	41.75	60.81	46.13
Weighted	-	46.35	50.53	68.31	70.18	46.87	48.39	86.09	59.53
Prompt LLM [12]	✗	46.67	51.48	61.85	65.93	39.58	49.17	71.25	55.13
GraphRouter [12]	✗	47.41	51.48	68.89	70.37	51.67	50.42	87.92	61.17
RouterDC [7]	✗	44.81	50.37	67.04	68.89	40.00	48.33	77.50	56.71
KNN Router [19]	✗	46.67	52.22	68.15	71.48	52.08	46.25	86.25	60.44
MF [33, 58]	✗	46.30	**52.59**	68.89	71.48	49.17	50.42	82.92	60.25
RAGRouter (Ours)	✓	**48.52**	**52.59**	**71.48**	**74.44**	**56.67**	**56.67**	**90.83**	**64.46**
Oracle	-	54.44	57.41	91.85	90.37	69.17	77.92	96.25	76.77

assignment; **Weighted** routing to different LLMs according to RAGRouter's empirical probability distribution. To show the ideal upper bound of routing performance, we introduce the **Oracle** baseline by routing each query to its optimal LLM using ground-truth performance data.

Implementation Details. For the RAGRouter architecture, we use all-mpnet-base-v2 [37] as the encoder for both queries and documents, and ms-marco-MiniLM-L12-v2 [46] as the cross-encoder, resulting in a total parameter size of approximately 136M. Both the knowledge representation vector and the RAG capability vector are set to a dimensionality of 768. To mitigate overfitting, all but the last two transformer layers in the query/document encoder and the cross-encoder are frozen during training. The classification loss weight λ is set to 2.0, and the contrastive learning temperature τ to 0.2. The router is optimized using AdamW [30] with a learning rate of 5e-5, batch size of 64, for 10 epochs. All experiments are conducted on a single NVIDIA RTX 4090D GPU.

5.2 Main Results

Comparison with Single Non-Routed LLMs. Table 2 presents the test accuracy across a variety of knowledge-intensive tasks and retrieval settings. RAGRouter consistently achieves the highest performance, with an average accuracy of 64.46%. It surpasses the best-performing single RAG-enabled LLM, LLaMA-3.3-70B-Instruct (60.85%), by +3.61%, demonstrating the effectiveness of routing in RAG. Notably, RAGRouter also outperforms the Oracle Single Best baseline (61.22%)—which selects the optimal single model for each dataset—by +3.24%, indicating that routing enables the integration of multiple RAG-enhanced LLMs to achieve performance beyond any individual model.

Comparison with Non-RAG-Aware and Other Baselines. RAGRouter significantly outperforms all non-retrieval-aware routing baselines, including GraphRouter (61.17%, +3.29%), MF (60.25%, +4.21%), KNN Router (60.44%, +4.02%), and RouterDC (56.71%, +7.75%). These results underscore the limitations of methods that do not explicitly model the interaction between LLMs and retrieved

knowledge. Without capturing retrieval-induced capability shifts, such approaches struggle to make effective routing decisions in RAG. RAGRouter also surpasses rule-based strategies such as Random (46.13%, +18.33%) and Weighted (59.53%, +4.93%). Together, these findings support our core claim: modeling RAG-specific capabilities and knowledge shift is essential for accurate LLM routing.

Inference Cost of RAGRouter. The inference characteristics are evaluated on a single NVIDIA RTX 4090D GPU. Peak GPU memory usage is 4147 MiB. With a batch size of 64, RAGRouter processes 270 instances in 3 seconds across 5 batches, yielding an average inference time of 0.011 seconds per instance. These results clearly show that RAGRouter is lightweight and efficient during inference.

5.3 Routing Performance Involving Closed-Source LLMs

To verify that RAGRouter can still deliver performance gains with strong closed-source LLMs, we augment the candidate LLMs set—comprising Qwen2.5-32B-Instruct, Qwen2.5-72B-Instruct, and Llama-3.3-70B-Instruct—with closed-source models Qwen2.5-Max [5], GPT-4o [6], and reasoning-based DeepSeek-R1 [7].

As shown in Table 3, RAGRouter outperforms the strongest model GPT-4o by +1.67%, while invoking GPT-4o in only 44.79% of samples. It also surpasses Qwen2.5-Max (+7.71%) and the reasoning-based DeepSeek-R1 (+4.17%). Furthermore, RAGRouter significantly outperforms non–RAG-aware baselines such as KNN Router (+4.17%) and MF (+2.92%). These results demonstrate that RAGRouter can effectively integrate closed-source models, leveraging their complementarity to achieve simultaneous improvements in both performance and efficiency.

Table 3: Testing accuracy (%) of RAGRouter and baselines with the inclusion of closed-source LLMs (Qwen2.5-Max, GPT-4o, and DeepSeek-R1) on NQ and WebQ, **bold** indicates best results.

Method	NQ	WQ	Avg
Qwen2.5-32B-Instruct	44.58	50.42	47.50
Qwen2.5-72B-Instruct	40.00	48.75	44.38
Llama-3.3-70B-Instruct	51.67	50.42	51.05
Qwen2.5-Max	51.25	52.08	51.67
GPT-4o	58.33	57.08	57.71
DeepSeek-R1	56.67	53.75	55.21
KNN Router	58.75	51.67	55.21
MF	57.50	55.42	56.46
RAGRouter	**59.58**	**59.17**	**59.38**

5.4 Extended Results on Latency-Aware Routing

Metrics. Three metrics are used for evaluate latency-aware routing: **Area**, which measures the proportion of the area under the accuracy–latency curve within 1 second, reflecting overall time-efficiency; **Peak Acc**, the highest accuracy achieved within 1 second; and **Latency Gap-to-Match**, defined as the latency of the best-performing single LLM minus the minimum latency required by a method to match its accuracy, indicating the efficiency margin obtained through routing.

Table 4: Area (%), Peak Accuracy (PA, %), and Latency Gap-to-Match (G, s) for RAGRouter and baselines using score-threshold-based routing on MedMCQA (Local/Online) and TriviaQA. "–" indicates failure to match the best single-LLM performance; **bold** denotes the best result.

Method	MedMCQA (Local)			MedMCQA (Online)			TQA		
	Area ↑	PA ↑	G (s) ↑	Area ↑	PA ↑	G (s) ↑	Area ↑	PA ↑	G (s) ↑
RouterDC	55.87	62.22	-	57.77	65.56	-	70.99	75.83	-
MF	46.42	59.63	0.10	54.85	65.56	0.45	66.84	80.00	-
GraphRouter	47.96	59.30	0.18	57.82	64.67	0.32	65.42	73.15	0.01
KNN Router	52.26	62.30	-	60.18	66.10	0.33	72.61	81.65	-
RAGRouter	**57.12**	**62.59**	**0.24**	**63.12**	**67.78**	**0.51**	**73.78**	**87.50**	**0.76**

Quantitative Results and Visualization. We apply the score-threshold-based routing mechanism to RAGRouter and baselines across all tasks. Results on MedMCQA (Local/Online) and TriviaQA are reported in Table 4 and Figure 4, with supplementary results in Appendix C. RAGRouter consistently achieves the highest Area and Peak Accuracy across all tasks. Specifically, it outperforms baselines

[5] `https://chat.qwen.ai`

[6] `https://platform.openai.com/docs/models/gpt-4o`

[7] `https://www.deepseek.com`

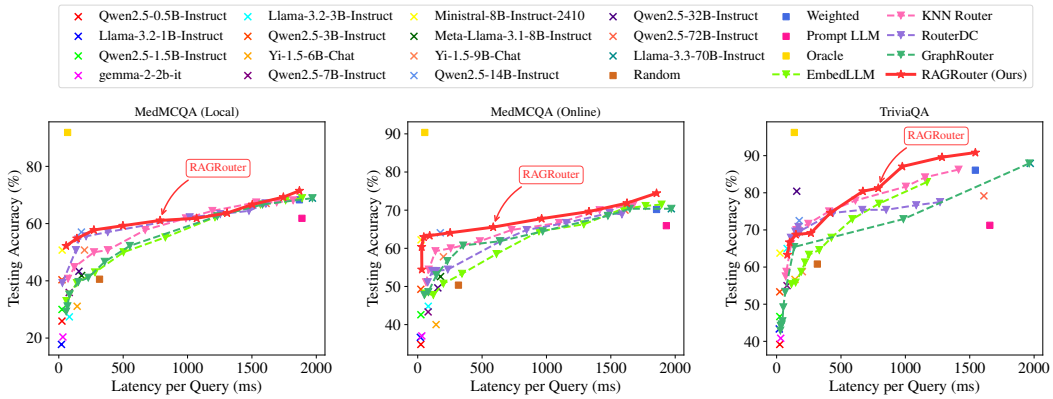

Figure 4: Accuracy–latency curves on MedMCQA (Local), MedMCQA (Online), and TriviaQA.

by 4.86%–10.7%, 2.94%–8.27%, and 1.17%–8.36% in Area, and by 0.29%–3.29%, 1.68%–3.11%, and 5.85%–14.35% in Peak Accuracy on MedMCQA (Local/Online) and TriviaQA, respectively. As shown in Figure 4, its accuracy–latency curve consistently dominates those of baselines, indicating superior accuracy under low-latency constraints. RAGRouter also achieves the largest positive Latency Gap-to-Match margins, demonstrating it can match the accuracy of the best single LLM with substantially lower latency. These results highlight RAGRouter's ability to exploit the optimization space enabled by retrieval augmentation, efficiently leveraging smaller, faster models to achieve the performance of larger ones without incurring unnecessary latency.

5.5 Sensitivity Analysis and Ablation Study

Table 5: Ablation study of Cross Encoder and RAG Capability Embedding Layer.

Configuration	PopQA		MedMCQA		NQ	WQ	TQA	Avg	Δ
	Local	Online	Local	Online					
RAGRouter	48.52	52.59	71.48	74.44	56.67	56.67	90.83	64.46	0.00
w/o Cross Encoder	47.78	52.22	70.74	74.44	55.42	55.00	88.75	63.48	-0.98
w/o RAG Capability Embedding Layer	48.52	52.22	71.11	74.44	55.42	54.17	90.00	63.70	-0.76

Table 6: Ablation study of contrastive learning objectives.

ISC	CSC	PopQA		MedMCQA		NQ	WQ	TQA	Avg	Δ
		Local	Online	Local	Online					
✓	✓	48.52	52.59	71.48	74.44	56.67	56.67	90.83	64.46	0.00
✓	✗	48.15	52.22	71.11	73.33	53.75	56.25	89.58	63.49	-0.97
✗	✓	48.52	51.85	71.48	74.07	55.00	56.25	89.58	63.82	-0.64
✗	✗	47.78	51.48	69.26	70.74	53.75	53.75	89.17	62.28	-2.18

Ablation Study on RAGRouter Architecture. We conduct an ablation study to evaluate the contributions of the Cross Encoder and the RAG Capability Embedding Layer in RAGRouter. As shown in Table 5, removing the Cross Encoder reduces performance by 0.98%, highlighting the importance of query-document interactions for deriving fused knowledge representations. Removing the RAG Capability Embedding Layer reduces performance by 0.76%; in this configuration, we removed the explicit modeling of RAG capabilities and instead assigned each LLM a single embedding capturing its overall behavior, without distinguishing between parametric knowledge and RAG capabilities. This result demonstrates that explicitly modeling RAG capabilities is essential for effective routing that accounts for each LLM's capacity to exploit external information.

Effects of Positive and Negative Sample Selection in Contrastive Learning. To assess the effectiveness of contrastive learning, we perform an ablation study on the two positive–negative sample construction strategies used in our method. As shown in Table 6, removing either Intra-Setting Contrast or Cross-Setting Contrast results in performance drops of 0.64% and 0.97%, respectively. When both components are removed—effectively disabling contrastive learning—accuracy decreases

Table 7: Effects of different candidate LLMs sets.

Candidate Set	Method	PopQA		MedMCQA		NQ	WQ	TQA	Avg	Δ
		Local	Online	Local	Online					
Small	Oracle Single Best	45.56	51.48	40.37	49.26	36.67	45.42	65.00	47.68	0.00
	RAGRouter	45.56	51.85	40.37	51.48	36.67	47.92	65.00	48.41	+0.73
Large	Oracle Single Best	46.30	50.37	68.89	70.37	51.67	50.42	87.92	60.85	0.00
	RAGRouter	47.41	50.74	71.11	73.33	54.58	53.33	90.42	62.99	+2.14
Small & Large	Oracle Single Best	46.30	51.48	68.89	70.37	51.67	50.42	87.92	61.01	0.00
	RAGRouter	48.15	52.22	71.48	73.33	53.33	55.00	89.58	63.30	+2.29

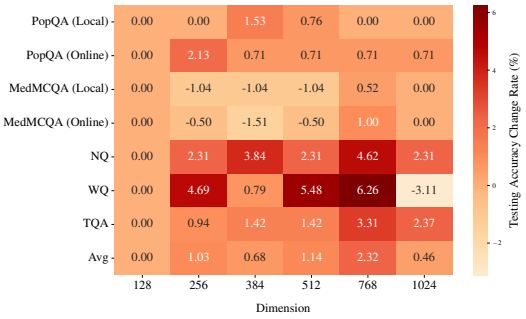

Figure 5: Effects of dimension (baseline: 128).

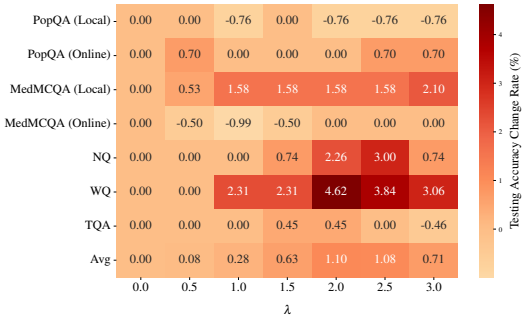

Figure 6: Effects of λ (baseline: $\lambda = 0$).

by 2.18%. These findings underscore the importance of contrastive learning in capturing knowledge representation shifts and LLM heterogeneity, which is essential for effective routing in RAG settings.

Effects of the Dimension of LLM Knowledge and RAG Capability Embeddings. We investigate the impact of the dimensionality of both knowledge representation and RAG capability vectors. As shown in Figure 5, we observe that performance improves as the dimensionality increases, peaking at 768, after which it declines. Accordingly, we adopt a dimensionality of 768 in the main experiments. Detailed results are provided in Appendix D.

Effects of λ. We investigate the impact of the classification loss weight λ in the overall loss function (Eq. 9) on test accuracy. As shown in Figure 6, we observe that combining contrastive and classification losses yields better performance than using contrastive loss alone (i.e., $\lambda = 0$, achieving only 63.76% on average). As λ increases, accuracy improves, peaking at $\lambda = 2$ with 64.46% on average, before experiencing a slight decline. Based on this, we set $\lambda = 2$ in all experiments. Detailed results are provided in Appendix D.

Effects of Different Candidate LLMs Sets. We investigate how the composition of candidate LLMs affects routing performance. To this end, we form two subsets of models—Small ($\leq$3B parameters) and Large ($\geq$32B)—and evaluate three configurations: Small only, Large only, and a heterogeneous set combining both. As shown in Table 7, RAGRouter consistently outperforms the Oracle Single Best in all settings, demonstrating its ability to coordinate models and achieve cumulative gains. Two key insights emerge. First, the routing upper bound is largely determined by model strength: the Large set yields a significantly higher Oracle Single Best average (60.85%) than the Small set (47.68%), with RAGRouter following the same trend (62.99% vs. 48.41%). Second, combining heterogeneous models yields the best performance: the Small & Large setting achieves the highest RAGRouter average (63.30%) and the largest gain over its Oracle Single Best (+2.29%), suggesting that model diversity improves complementarity and enables more effective routing.

6 Conclusion

In this paper, we have studied the problem of LLM routing in Retrieval-Augmented Generation (RAG) for the first time and propose RAGRouter, the first RAG-aware routing method. By leveraging contrastive learning, RAGRouter captures knowledge representation shifts induced by external documents, enabling effective routing decisions. Experiments on diverse knowledge-intensive tasks demonstrate that RAGRouter outperforms existing non-RAG-aware methods and achieves strong performance-efficiency trade-offs under low-latency constraints.

Acknowledgments and Disclosure of Funding

This work was supported in part by National Key R&D Program of China (No. 2022ZD0119100), China NSF grant No. 62025204, No. 62202296, No. 62272293, No. 62441236, No. U24A20326, and No. 62572299, Tencent WeChat Research Program, and SJTU-Huawei Explore X Gift Fund. The opinions, findings, conclusions, and recommendations expressed in this paper are those of the authors and do not necessarily reflect the views of the funding agencies or the government.

References

[1] Q Jiang Albert, Alexandre Sablayrolles, Arthur Mensch, Chris Bamford, and Devendra Singh Chaplot. Mistral 7b. *arXiv*, 2023.

[2] Jonathan Berant, Andrew Chou, Roy Frostig, and Percy Liang. Semantic parsing on freebase from question-answer pairs. In *Proceedings of the 2013 conference on empirical methods in natural language processing*, pages 1533–1544, 2013.

[3] Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901, 2020.

[4] Sébastien Bubeck, Varun Chadrasekaran, 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, 2023.

[5] Deng Cai, Yan Wang, Lemao Liu, and Shuming Shi. Recent advances in retrieval-augmented text generation. In *Proceedings of the 45th international ACM SIGIR conference on research and development in information retrieval*, pages 3417–3419, 2022.

[6] Jiawei Chen, Hongyu Lin, Xianpei Han, and Le Sun. Benchmarking large language models in retrieval-augmented generation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pages 17754–17762, 2024.

[7] Shuhao Chen, Weisen Jiang, Baijiong Lin, James T Kwok, and Yu Zhang. Routerdc: Query-based router by dual contrastive learning for assembling large language models. *arXiv preprint arXiv:2409.19886*, 2024.

[8] Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. A simple framework for contrastive learning of visual representations. In *International conference on machine learning*, pages 1597–1607. PmLR, 2020.

[9] Zhijun Chen, Jingzheng Li, Pengpeng Chen, Zhuoran Li, Kai Sun, Yuankai Luo, Qianren Mao, Dingqi Yang, Hailong Sun, and Philip S Yu. Harnessing multiple large language models: A survey on llm ensemble. *arXiv preprint arXiv:2502.18036*, 2025.

[10] Dujian Ding, Ankur Mallick, Chi Wang, Robert Sim, Subhabrata Mukherjee, Victor Ruhle, Laks VS Lakshmanan, and Ahmed Hassan Awadallah. Hybrid llm: Cost-efficient and quality-aware query routing. *arXiv preprint arXiv:2404.14618*, 2024.

[11] Feiteng Fang, Yuelin Bai, Shiwen Ni, Min Yang, Xiaojun Chen, and Ruifeng Xu. Enhancing noise robustness of retrieval-augmented language models with adaptive adversarial training. *arXiv preprint arXiv:2405.20978*, 2024.

[12] Tao Feng, Yanzhen Shen, and Jiaxuan You. Graphrouter: A graph-based router for llm selections. In *The Thirteenth International Conference on Learning Representations*, 2024.

[13] Yunfan Gao, Yun Xiong, Xinyu Gao, Kangxiang Jia, Jinliu Pan, Yuxi Bi, Yi Dai, Jiawei Sun, Haofen Wang, and Haofen Wang. Retrieval-augmented generation for large language models: A survey. *arXiv preprint arXiv:2312.10997*, 2, 2023.

[14] Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Alex Vaughan, et al. The llama 3 herd of models. *arXiv preprint arXiv:2407.21783*, 2024.

[15] Kelvin Guu, Kenton Lee, Zora Tung, Panupong Pasupat, and Mingwei Chang. Retrieval augmented language model pre-training. In *International conference on machine learning*, pages 3929–3938. PMLR, 2020.

[16] Hiroaki Hayashi, Prashant Budania, Peng Wang, Chris Ackerson, Raj Neervannan, and Graham Neubig. Wikiasp: A dataset for multi-domain aspect-based summarization. *Transactions of the Association for Computational Linguistics*, 9:211–225, 2021.

[17] Hangfeng He, Hongming Zhang, and Dan Roth. Rethinking with retrieval: Faithful large language model inference. *arXiv preprint arXiv:2301.00303*, 2022.

[18] Kaiming He, Haoqi Fan, Yuxin Wu, Saining Xie, and Ross Girshick. Momentum contrast for unsupervised visual representation learning. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 9729–9738, 2020.

[19] 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.

[20] Gautier Izacard and Edouard Grave. Leveraging passage retrieval with generative models for open domain question answering. *arXiv preprint arXiv:2007.01282*, 2020.

[21] Gautier Izacard, Patrick Lewis, Maria Lomeli, Lucas Hosseini, Fabio Petroni, Timo Schick, Jane Dwivedi-Yu, Armand Joulin, Sebastian Riedel, and Edouard Grave. Few-shot learning with retrieval augmented language models. *arXiv preprint arXiv:2208.03299*, 1(2):4, 2022.

[22] Ziwei Ji, Nayeon Lee, Rita Frieske, Tiezheng Yu, Dan Su, Yan Xu, Etsuko Ishii, Ye Jin Bang, Andrea Madotto, and Pascale Fung. Survey of hallucination in natural language generation. *ACM computing surveys*, 55(12):1–38, 2023.

[23] Mandar Joshi, Eunsol Choi, Daniel S Weld, and Luke Zettlemoyer. Triviaqa: A large scale distantly supervised challenge dataset for reading comprehension. *arXiv preprint arXiv:1705.03551*, 2017.

[24] Vladimir Karpukhin, Barlas Oguz, Sewon Min, Patrick SH Lewis, Ledell Wu, Sergey Edunov, Danqi Chen, and Wen-tau Yih. Dense passage retrieval for open-domain question answering. In *EMNLP (1)*, pages 6769–6781, 2020.

[25] Tom Kwiatkowski, Jennimaria Palomaki, Olivia Redfield, Michael Collins, Ankur Parikh, Chris Alberti, Danielle Epstein, Illia Polosukhin, Jacob Devlin, Kenton Lee, et al. Natural questions: a benchmark for question answering research. *Transactions of the Association for Computational Linguistics*, 7:453–466, 2019.

[26] Woosuk Kwon, Zhuohan Li, Siyuan Zhuang, Ying Sheng, Lianmin Zheng, Cody Hao Yu, Joseph E. Gonzalez, Hao Zhang, and Ion Stoica. Efficient memory management for large language model serving with pagedattention. In *Proceedings of the ACM SIGOPS 29th Symposium on Operating Systems Principles*, 2023.

[27] Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen-tau Yih, Tim Rocktäschel, et al. Retrieval-augmented generation for knowledge-intensive nlp tasks. *Advances in neural information processing systems*, 33:9459–9474, 2020.

[28] Kuan Li, Liwen Zhang, Yong Jiang, Pengjun Xie, Fei Huang, Shuai Wang, and Minhao Cheng. Lara: Benchmarking retrieval-augmented generation and long-context llms-no silver bullet for lc or rag routing. *arXiv preprint arXiv:2502.09977*, 2025.

[29] Xianzhi Li, Samuel Chan, Xiaodan Zhu, Yulong Pei, Zhiqiang Ma, Xiaomo Liu, and Sameena Shah. Are chatgpt and gpt-4 general-purpose solvers for financial text analytics? a study on several typical tasks. *arXiv preprint arXiv:2305.05862*, 2023.

[30] Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101*, 2017.

[31] 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. *arXiv preprint arXiv:2311.08692*, 2023.

[32] Alex Mallen, Akari Asai, Victor Zhong, Rajarshi Das, Daniel Khashabi, and Hannaneh Hajishirzi. When not to trust language models: Investigating effectiveness of parametric and non-parametric memories. *arXiv preprint arXiv:2212.10511*, 2022.

[33] 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.

[34] Ankit Pal, Logesh Kumar Umapathi, and Malaikannan Sankarasubbu. Medmcqa: A large-scale multi-subject multi-choice dataset for medical domain question answering. In *Conference on health, inference, and learning*, pages 248–260. PMLR, 2022.

[35] Alec Radford, Karthik Narasimhan, Tim Salimans, Ilya Sutskever, et al. Improving language understanding by generative pre-training. *OpenAI blog*, 2018.

[36] Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9, 2019.

[37] Nils Reimers and Iryna Gurevych. Sentence-bert: Sentence embeddings using siamese bert-networks. *arXiv preprint arXiv:1908.10084*, 2019.

[38] Marija Šakota, Maxime Peyrard, and Robert West. Fly-swat or cannon? cost-effective language model choice via meta-modeling. In *Proceedings of the 17th ACM International Conference on Web Search and Data Mining*, pages 606–615, 2024.

[39] Freda Shi, Xinyun Chen, Kanishka Misra, Nathan Scales, David Dohan, Ed H Chi, Nathanael Schärli, and Denny Zhou. Large language models can be easily distracted by irrelevant context. In *International Conference on Machine Learning*, pages 31210–31227. PMLR, 2023.

[40] 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.

[41] Devendra Singh, Siva Reddy, Will Hamilton, Chris Dyer, and Dani Yogatama. End-to-end training of multi-document reader and retriever for open-domain question answering. *Advances in Neural Information Processing Systems*, 34:25968–25981, 2021.

[42] Huatong Song, Jinhao Jiang, Yingqian Min, Jie Chen, Zhipeng Chen, Wayne Xin Zhao, Lei Fang, and Ji-Rong Wen. R1-searcher: Incentivizing the search capability in llms via reinforcement learning. *arXiv preprint arXiv:2503.05592*, 2025.

[43] Dimitris Stripelis, Zijian Hu, Jipeng Zhang, Zhaozhuo Xu, Alay Dilipbhai Shah, Han Jin, Yuhang Yao, Salman Avestimehr, and Chaoyang He. Tensoropera router: A multi-model router for efficient llm inference. *arXiv preprint arXiv:2408.12320*, 2024.

[44] Gemma Team, Morgane Riviere, Shreya Pathak, Pier Giuseppe Sessa, Cassidy Hardin, Surya Bhupatiraju, Léonard Hussenot, Thomas Mesnard, Bobak Shahriari, Alexandre Ramé, et al. Gemma 2: Improving open language models at a practical size. *arXiv preprint arXiv:2408.00118*, 2024.

[45] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

[46] Wenhui Wang, Furu Wei, Li Dong, Hangbo Bao, Nan Yang, and Ming Zhou. Minilm: Deep self-attention distillation for task-agnostic compression of pre-trained transformers. *Advances in neural information processing systems*, 33:5776–5788, 2020.

[47] Xiaohua Wang, Zhenghua Wang, Xuan Gao, Feiran Zhang, Yixin Wu, Zhibo Xu, Tianyuan Shi, Zhengyuan Wang, Shizheng Li, Qi Qian, et al. Searching for best practices in retrieval-augmented generation. *arXiv preprint arXiv:2407.01219*, 2024.

[48] Zilong Wang, Zifeng Wang, Long Le, Huaixiu Steven Zheng, Swaroop Mishra, Vincent Perot, Yuwei Zhang, Anush Mattapalli, Ankur Taly, Jingbo Shang, et al. Speculative rag: Enhancing retrieval augmented generation through drafting. *arXiv preprint arXiv:2407.08223*, 2024.

[49] Zhepei Wei, Wei-Lin Chen, and Yu Meng. Instructrag: Instructing retrieval-augmented generation via self-synthesized rationales. *arXiv preprint arXiv:2406.13629*, 2024.

[50] Shitao Xiao, Zheng Liu, Peitian Zhang, Niklas Muennighoff, Defu Lian, and Jian-Yun Nie. C-pack: Packed resources for general chinese embeddings. In *Proceedings of the 47th international ACM SIGIR conference on research and development in information retrieval*, pages 641–649, 2024.

[51] Ran Xu, Wenqi Shi, Yue Yu, Yuchen Zhuang, Yanqiao Zhu, May D Wang, Joyce C Ho, Chao Zhang, and Carl Yang. Bmretriever: Tuning large language models as better biomedical text retrievers. *arXiv preprint arXiv:2404.18443*, 2024.

[52] An Yang, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chengyuan Li, Dayiheng Liu, Fei Huang, Haoran Wei, et al. Qwen2. 5 technical report. *arXiv preprint arXiv:2412.15115*, 2024.

[53] Haoran Yang, Yumeng Zhang, Jiaqi Xu, Hongyuan Lu, Pheng Ann Heng, and Wai Lam. Unveiling the generalization power of fine-tuned large language models. *arXiv preprint arXiv:2403.09162*, 2024.

[54] Zhilin Yang, Peng Qi, Saizheng Zhang, Yoshua Bengio, William W Cohen, Ruslan Salakhutdinov, and Christopher D Manning. Hotpotqa: A dataset for diverse, explainable multi-hop question answering. *arXiv preprint arXiv:1809.09600*, 2018.

[55] Alex Young, Bei Chen, Chao Li, Chengen Huang, Ge Zhang, Guanwei Zhang, Guoyin Wang, Heng Li, Jiangcheng Zhu, Jianqun Chen, et al. Yi: Open foundation models by 01. ai. *arXiv preprint arXiv:2403.04652*, 2024.

[56] Peitian Zhang, Shitao Xiao, Zheng Liu, Zhicheng Dou, and Jian-Yun Nie. Retrieve anything to augment large language models. *arXiv preprint arXiv:2310.07554*, 2023.

[57] Yue Zhang, Ming Zhang, Haipeng Yuan, Shichun Liu, Yongyao Shi, Tao Gui, Qi Zhang, and Xuanjing Huang. Llmeval: A preliminary study on how to evaluate large language models. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pages 19615–19622, 2024.

[58] Richard Zhuang, Tianhao Wu, Zhaojin Wen, Andrew Li, Jiantao Jiao, and Kannan Ramchandran. Embedllm: Learning compact representations of large language models. *arXiv preprint arXiv:2410.02223*, 2024.

NeurIPS Paper Checklist

The checklist is designed to encourage best practices for responsible machine learning research, addressing issues of reproducibility, transparency, research ethics, and societal impact. Do not remove the checklist: **The papers not including the checklist will be desk rejected.** The checklist should follow the references and follow the (optional) supplemental material. The checklist does NOT count towards the page limit.

Please read the checklist guidelines carefully for information on how to answer these questions. For each question in the checklist:

- You should answer [Yes] , [No] , or [NA] .
- [NA] means either that the question is Not Applicable for that particular paper or the relevant information is Not Available.
- Please provide a short (1–2 sentence) justification right after your answer (even for NA).

The checklist answers are an integral part of your paper submission. They are visible to the reviewers, area chairs, senior area chairs, and ethics reviewers. You will be asked to also include it (after eventual revisions) with the final version of your paper, and its final version will be published with the paper.

The reviewers of your paper will be asked to use the checklist as one of the factors in their evaluation. While "[Yes] " is generally preferable to "[No] ", it is perfectly acceptable to answer "[No] " provided a proper justification is given (e.g., "error bars are not reported because it would be too computationally expensive" or "we were unable to find the license for the dataset we used"). In general, answering "[No] " or "[NA] " is not grounds for rejection. While the questions are phrased in a binary way, we acknowledge that the true answer is often more nuanced, so please just use your best judgment and write a justification to elaborate. All supporting evidence can appear either in the main paper or the supplemental material, provided in appendix. If you answer [Yes] to a question, in the justification please point to the section(s) where related material for the question can be found.

IMPORTANT, please:

- **Delete this instruction block, but keep the section heading "NeurIPS Paper Checklist"**,
- **Keep the checklist subsection headings, questions/answers and guidelines below.**
- **Do not modify the questions and only use the provided macros for your answers**.

1. **Claims**

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

   Answer: [Yes]

   Justification: Yes, main claims made in the abstract and introduction accurately reflect the paper's contributions and scope.

   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: Yes, the paper discuss the limitations of the work performed by the authors.

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: [NA]

   Justification: The paper does not include theoretical results.

   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: We have provided detailed training settings and datasets information in the Experiments section and Appendix.

   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: We use public dataset to conduct experiments, and our data and source code will be released.

   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 have specified all the experimental setting and details in the Experiments section and Appendix.

   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: [No]

   Justification: Error bars are not reported because it would be too computationally expensive.

   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: All experiments are conducted on a single NVIDIA RTX 4090D GPU.

   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: Yes, the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics.

   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: Yes, the broader impacts are discussed in the conclusion of this paper.

    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: [NA]

Justification: No such risks.

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: We have cited all referenced papers and open-source assets.

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 construct data suitable for LLM routing under RAG based on public datasets, and relevant details are presented in the Experiments section and Appendix.

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: Not involved.

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: Not involved.

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: [Yes]

Justification: Yes, the paper describes the usage of LLMs for LLMs are important component of experiments.

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.

Appendix Contents

A Experimental Setup Details

A.1 Details of Candidate LLMs

The responses of Qwen2.5-72B-Instruct, Llama-3-70B-Instruct and closed-source LLMs were obtained via API calls, while other open-source LLMs were locally deployed using the vLLM framework [26] for high-speed inference from Huggingface[8]. Notably, latency calculations for API-based models incorporated both average network latency and inference time, whereas latency measurements for locally deployed models exclusively accounted for inference time.

A.2 Details of Datasets

Table 8: The statistics results for different tasks.

Query Num	PopQA		MedMCQA		NQ	WQ	TQA
	Local	Online	Local	Online			
Train	2000	2000	2000	2000	1000	1000	1000
Test	270	270	270	270	240	240	240

As shown in Table 8, we randomly sampled queries from five knowledge-intensiv tasks (PopQA [32], MedMCQA [34], NQ [25], WebQ [2], and TriviaQA [23]) and partitioned them into training and test sets. For PopQA and MedMCQA, we retrieved documents through both local and online search engines following [42], while for NQ, WebQ, and TriviaQA, we constructed artificially noised documents based on [11], with an equal proportion of four types: Golden Context, Relevant Retrieval Noise, Irrelevant Retrieval Noise, and Counterfactual Retrieval Noise. Each query-document pair was processed by the 15 LLMs described in Section 5.1 to generate responses. These responses were then evaluated against ground-truth answers to derive binary scores.

B Impact of RAG on LLM Performance

Performance Shift with and without RAG. We analyze how LLM performance changes before and after RAG across different models and settings. As shown in Figure 7, on real retrieval settings such as PopQA (Online), MedMCQA (Local), and MedMCQA (Online), RAG improves overall performance and reduces the accuracy gap among models. Notably, small-scale models benefit more (e.g., Qwen2.5-0.5B-Instruct achieves a 17.57% improvement on MedMCQA (Online)) compared to larger models (e.g., Llama-3.3-70B-Instruct improves by only 0.88%). In contrast, under noisy retrieval settings (NQ, WebQ, TriviaQA), the impact of RAG varies—some models improve while others degrade (e.g., on NQ, Llama-3.3-70B-Instruct performs worse, while Qwen2.5-0.5B-Instruct performs better). These results highlight inconsistent performance shifts across models, indicating that RAG significantly alters the distribution of response quality, which undermines the assumptions of non-RAG-aware routing strategies.

Analysis of Response Quality Reversal. We further investigate how RAG causes quality reversals for the same query. To quantify these reversals, we define two metrics at the task level. The positive gain rate is the proportion of queries that were unanswerable without retrieval but became answerable with retrieved documents, calculated as:

$$\text{Positive Gain Rate} = \frac{\#(\text{Incorrect w/o RAG} \rightarrow \text{Correct w/ RAG})}{\#(\text{Incorrect w/o RAG})} \tag{10}$$

The negative interference rate measures the opposite effect—queries that were answerable without retrieval but became unanswerable due to retrieved content:

$$\text{Negative Interference Rate} = \frac{\#(\text{Correct w/o RAG} \rightarrow \text{Incorrect w/ RAG})}{\#(\text{Correct w/o RAG})} \tag{11}$$

Figures 8 and 9 report these metrics across 15 LLMs and multiple settings. On average, across all models and tasks, the positive gain rate is 30.86%, negative interference rate is 31.46%. These results confirm that response quality reversals induced by external documents are common suggests that RAG-induced knowledge shifts are widespread among LLMs.

[8] https://huggingface.co

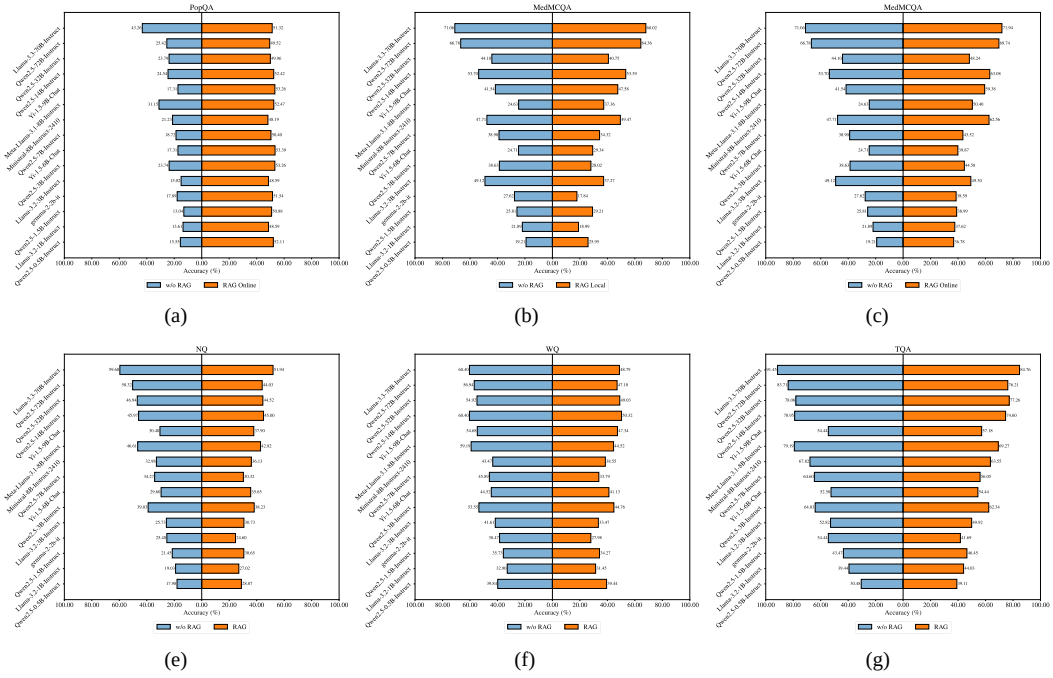

Figure 7: Accuracy of 15 LLMs before and after RAG under various tasks and retrieval settings: (a) PopQA (Online), (b) MedMCQA (Local), (c) MedMCQA (Online), (d) NQ, (e) WebQ, (f) TQA.

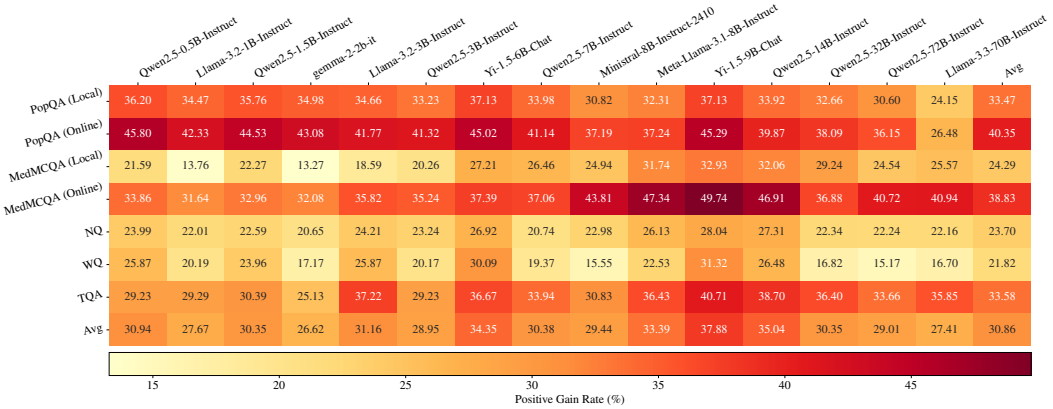

Figure 8: Positive Gain Rates of 15 candidate LLMs across tasks.

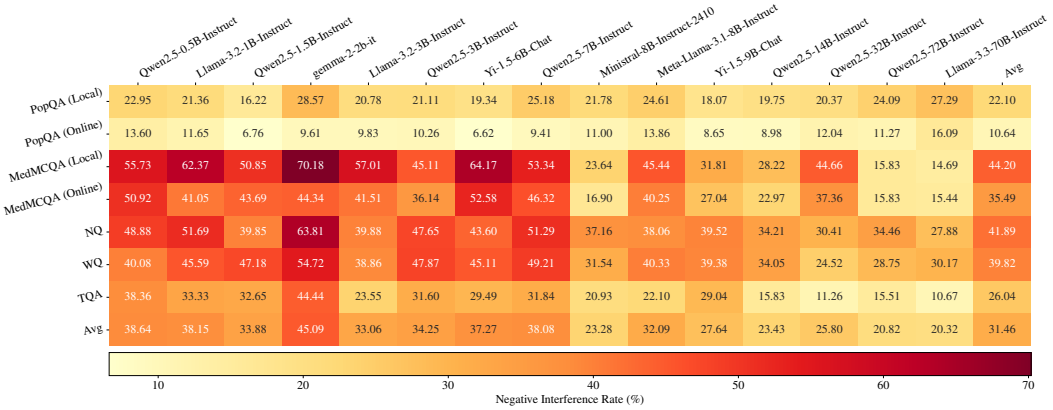

Figure 9: Negative Interference Rates of 15 candidate LLMs across tasks.

C Full Results on Latency-Aware Routing

Setting Details. Based on LLMs' profiles, the 15 candidate models in Section 5.4 were ranked as follows in ascending order: Qwen2.5-0.5B-Instruct, Llama-3.2-1B-Instruct, Qwen2.5-1.5B-Instruct, gemma-2-2b-it, Llama-3.2-3B-Instruct, Qwen2.5-3B-Instruct, Yi-1.5-6B-Chat, Qwen2.5-7B-Instruct, Ministral-8B-Instruct-2410, Meta-Llama-3.1-8B-Instruct, Yi-1.5-9B-Chat, Qwen2.5-14B-Instruct, Qwen2.5-32B-Instruct, Qwen2.5-72B-Instruct, Llama-3.3-70B-Instruct. Substitution models were selected from immediate predecessors of the highest-routing-score model within the threshold. For quantitative analysis, we discretized the threshold parameter θ over [0, 1] with a step size of 1e-4 to generate complete high-precision accuracy–latency trade-off curves.

Results. Figure 10 illustrates the accuracy–latency curves of RAGRouter and baseline methods on PopQA (Local/Online), NQ, and WebQ, while Tables 9 present their quantitative results on Area, Peak Acc, and Latency Gap-to-Match metrics. RAGRouter achieves the highest scores in Area and Peak Acc, with its accuracy–latency curve mostly surpassing the baselines, demonstrating strong performance-efficiency trade-offs under low-latency constraints.

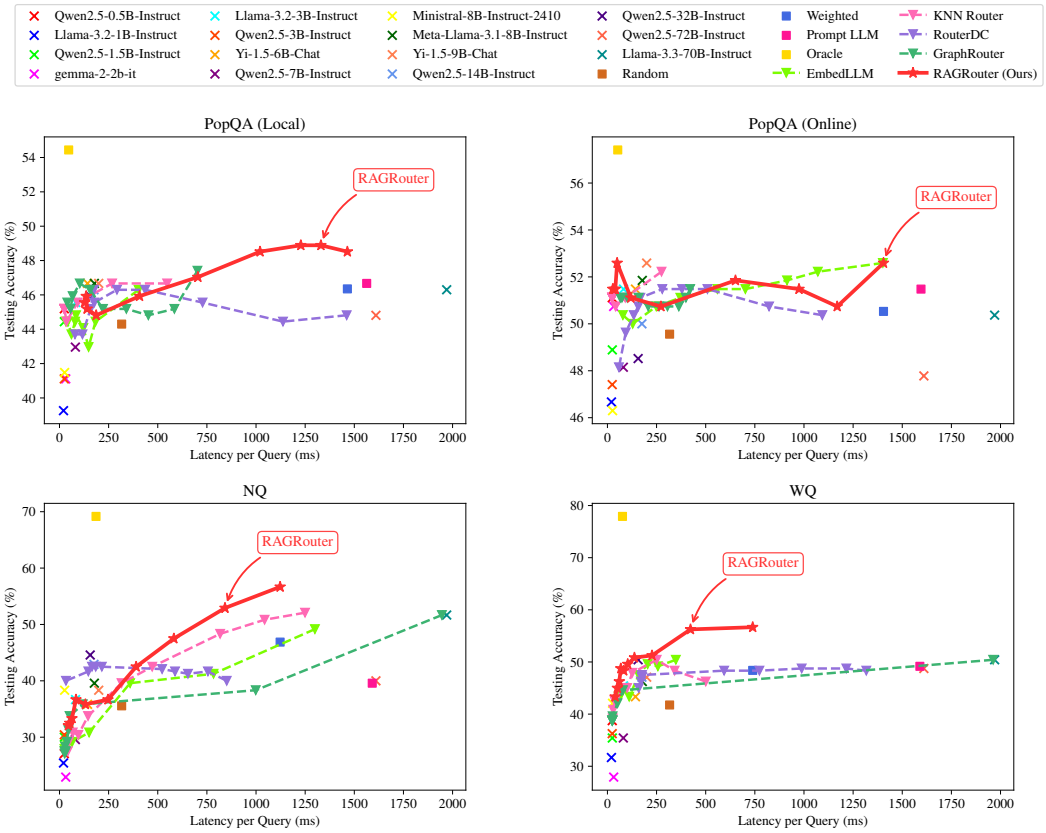

Figure 10: Accuracy–latency curves on PopQA (Local/Online), NQ and WebQ.

Table 9: Area (%), Peak Accuracy (PA, %), and Latency Gap-to-Match (G, s) for RAGRouter and baselines using score-threshold-based routing on PopQA (Local/Online), NQ and WebQ. "-" in Latency Gap-to-Match indicates failure to match the best single-LLM performance; "-" in Area denotes maximum routing latency below 1s, excluded from comparison; **bold** denotes the best result.

Method	PopQA (Local)			PopQA (Online)			NQ			WQ		
	Area ↑	PA ↑	G (s) ↑	Area ↑	PA ↑	G (s) ↑	Area ↑	PA ↑	G (s) ↑	Area ↑	PA ↑	G (s) ↑
RouterDC	44.58	47.04	**0.02**	49.80	51.85	-	33.10	42.92	-	46.39	49.17	-
MF	-	46.30	-	49.96	52.22	-0.91	37.34	44.17	-	-	50.42	1.66
GraphRouter	-	47.41	-0.50	-	51.48	-	35.68	38.35	0.02	45.67	48.18	0.01
KNN Router	-	46.67	-0.06	-	52.22	-	41.13	50.63	0.72	-	50.42	1.72
RAGRouter	**45.13**	**48.52**	-0.48	**50.13**	**52.59**	**0.15**	**43.63**	**55.83**	**1.13**	-	**58.33**	**1.84**

D Full Results of Sensitivity Analysis and Ablation Study

Table 10 presents the impact of the dimensionality of LLM knowledge embeddings and RAG capability embeddings on test accuracy, the best average accuracy is observed at dimension 768.

Table 10: Effects of dimension.

Dimension	PopQA		MedMCQA		NQ	WQ	TQA	Avg
	Local	Online	Local	Online				
128	48.52	52.22	71.11	73.70	54.17	53.33	87.92	63.00
256	48.52	53.33	70.37	73.33	55.42	55.83	88.75	63.65
384	49.26	52.59	70.37	72.59	56.25	53.75	89.17	63.43
512	48.89	52.59	70.37	73.33	55.42	56.25	89.17	63.72
768	48.52	52.59	71.48	74.44	56.67	56.67	90.83	64.46
1024	48.52	52.59	71.11	73.70	55.42	51.67	90.00	63.29

Table 11 presents the impact of the loss weight λ on test accuracy, the best average accuracy is observed at $\lambda = 2$.

Table 11: Effects of λ.

λ	PopQA		MedMCQA		NQ	WQ	TQA	Avg
	Local	Online	Local	Online				
0.0	48.89	52.59	70.37	74.44	55.42	54.17	90.42	63.76
0.5	48.89	52.96	70.74	74.07	55.42	54.17	90.42	63.81
1.0	48.52	52.59	71.48	73.70	55.42	55.42	90.42	63.94
1.5	48.89	52.59	71.48	74.07	55.83	55.42	90.83	64.16
2.0	48.52	52.59	71.48	74.44	56.67	56.67	90.83	64.46
2.5	48.52	52.96	71.48	74.44	57.08	56.25	90.42	64.45
3.0	48.52	52.96	71.85	74.44	55.83	55.83	90.00	64.21

E Case Studies of Failures and Successes

E.1 Quantitative Failure Case Studies of Routing Decisions

We conduct a quantitative analysis of cases where RAGRouter and non-RAG-aware routing methods fail to select the correct LLM or route correctly on the PopQA (Local), MedMCQA (Local), and NQ datasets. In particular, we categorize non-trivially unsolvable failures (i.e., excluding cases where all LLMs fail) into four types as follows:

- F1: Failure to perceive performance degradation caused by RAG (i.e., cases where LLM's performance decreases after RAG is applied).
- F2: Inherent task difficulty, where the majority of LLMs (e.g., >80%) fail, leading to routing confusion.
- F3: Overconfident selection of stronger LLMs for cases where high-capacity LLMs are chosen but still fail, outside the conditions of F1 and F2.
- F4: Other factors, such as outliers or ambiguous inputs.

Table 12: Type-wise failure rates (as a percentage of all cases) and the overall failure rate (FR).

Method	F1 (%)	F2 (%)	F3 (%)	F4 (%)	FR (%)
KNN Router	5.00	8.33	**1.28**	1.54	16.15
MF	5.51	8.21	1.41	1.79	16.92
RAGRouter	**3.85**	**7.05**	1.41	**0.64**	**12.95**

As shown in Table 12, we can observe that compared with non-RAG-aware routing methods, including KNN Router and MF, our RAGRouter achieves the lowest overall failure rate and the lowest F1-type failure rate. This suggests that RAGRouter more correctly captures performance shifts in LLMs introduced by RAG.

We additionally analyze cases where RAGRouter fails while non-RAG-aware routing methods succeed. We find that such cases are less than 2% of all the evaluated cases. Notably, more than 50% of these cases fall into the F2 type, where the performance gap between candidate LLMs is inherently small.

E.2 Qualitative Success Case Studies of RAGRouter

We further illustrate RAGRouter's ability to perceive changes in LLM knowledge states under RAG conditions through the two case studies shown in Table 13 and Table 14. In Table 13, the retrieved document contains both correct answer information and certain distracting content. In this case, RAGRouter identifies that the document provides significant performance gains for Qwen2.5-14B-Instruct and accordingly selects it as the responder. Although this model produces an incorrect response in the non-RAG setting due to confusion between two French departments, it successfully corrects the answer when the document is incorporated. In contrast, Meta-Llama-3.1-8B-Instruct exhibits greater sensitivity to distracting content in the document, where the introduction of RAG leads to reverse interference and impairs its reasoning, and thus it is not prioritized by RAGRouter. However, traditional routing strategies without RAG awareness, such as MF, rely solely on static model performance and fail to perceive the document-induced performance shifts, ultimately routing to Meta-Llama-3.1-8B-Instruct and resulting in routing failure for this sample.

Table 13: A case from PopQA (Online), demonstrating RAGRouter's ability to perceive LLMs' RAG capability and knowledge shift.

Query	What is Agen the capital of?	
Document	Agen, located in the Nouvelle-Aquitaine region of Southwestern France, serves as the prefecture of the Lot-et-Garonne department. Known for its geographical positioning along the river Garonne, the city lies approximately 135 kilometers southeast of Bordeaux. It has a rich cultural heritage, featuring various historical buildings such as the twelfth-century Agen Cathedral and numerous museums, including the Musée des Beaux Arts. Agen is also colloquially referred to as the c̈apital of the prune,ḧosting a popular prune festival every August. The town, with a population of 32,485 in 2021, has its own Roman Catholic diocese, adding to its historical significance within the region.	
Ground truth	Lot-et-Garonne	
Router	RAGRouter	MF
Selected LLM	Qwen2.5-14B-Instruct	Meta-Llama-3.1-8B-Instruct
Response w/o RAG	Lot department. (✗)	Lot-et-Garonne department. (✓)
Response w/ RAG	Lot-et-Garonne department. (✓)	The prune capital. (✗)

As shown in Table 14, the query itself is challenging, and neither Llama-3.3-70B-Instruct nor Qwen2.5-72B-Instruct is able to provide a correct answer without access to external documents. However, when the retrieved document containing key information is provided, Llama-3.3-70B-Instruct, benefiting from its stronger capabilities in information extraction and comprehension, successfully identifies "Poreotics" as the correct answer. In contrast, Qwen2.5-72B-Instruct fails to effectively utilize the document and still produces an incorrect response. In this scenario, RAGRouter is able to sense the differential capabilities of the candidate LLMs in leveraging retrieved content and routes the query to the model with stronger information extraction ability, leading to a correct response. By contrast, non-RAG-aware routing methods based on static modeling assumptions, such as RouterDC, fail to capture dynamic performance shifts induced by retrieval and result in incorrect routing and response failure. This case further highlights the advantage of RAGRouter in perceiving capability differences among LLMs in retrieval-augmented settings.

Table 14: A case from NQ, demonstrating RAGRouter's ability to perceive LLMs' RAG capability and knowledge shift.

Query	who are the dancers in the lazy song video?	
Document	tenth was the one chosen. The official video was directed by Mars and Cameron Duddy, produced by Nick Tabri and Dara Siegel, and features Poreotics wearing chimpanzee masks; it was released on April 15, 2011. The whole video is presented in as a lone continuous and uninterrupted shot, it begins with Mars singing and hanging out in a bedroom with five dancers, they all wear monkey masks and Mars dresses in black sunglasses and a flannel shirt. While Mars sings what he feels to do on a day off, he and the monkeys perform dance moves typical of a boy-band,	
Ground truth	Poreotics	
Router	RAGRouter	RouterDC
Selected LLM	Llama-3.3-70B-Instruct	Qwen2.5-72B-Instruct
Response w/o RAG	Bruno Mars and dancers. (✗)	Five dancers. (✗)
Response w/ RAG	Poreotics dancers. (✓)	Bruno Mars and his backup dancers. (✗)

F Routing Performance under Cross-Domain Settings

To evaluate RAGRouter's generalization across domains, we design two cross-domain settings:

- Setting 1: Trained on MedMCQA (Local) that falls into medical domain, but tested on a new dataset HotpotQA [54] for Wikipedia-based multi-hop QA.
- Setting 2: Trained on NQ that contains real-world search queries, but tested on MedMCQA (Local).

Table 15: Testing accuracy (%) of RAGRouter and baselines under two cross-domain settings.

Method	Setting 1 MedMCQA (Local)→HotpotQA	Setting 2 NQ→MedMCQA (Local)
KNN Router	18.60	63.33
MF	20.20	58.89
RAGRouter	**21.20**	**68.89**

As shown in Table 15, RAGRouter outperforms non–RAG-aware routing methods by 1.00% and 5.56% in two cross-domain settings, demonstrating strong generalization. This performance advantage can be attributed to RAGRouter's contrastive learning framework, which effectively captures relative RAG capability differences among candidate LLMs, even when transferred across domains.

G Routing Performance under Noisy Retrieval

We further partition the TriviaQA dataset into four subsets with manually injected retrieval noise—Golden Context, Relevant Noise, Irrelevant Noise, and Counterfactual Noise—to evaluate the routing effectiveness of RAGRouter under different noise conditions, in comparison with various baseline methods. As shown in Table 16, RAGRouter consistently achieves the best performance across all subsets, outperforming both the Oracle Single Best and other routing strategies, demonstrating strong robustness to different types of retrieval noise.

Notably, on the Relevant Noise, Irrelevant Noise, and Counterfactual Noise subsets, non-RAG-aware baselines exhibit significant performance gaps compared to RAGRouter, highlighting their limitations in high-noise retrieval scenarios. We hypothesize that this is due to knowledge shift induced by noisy retrieval, which affects different LLMs in heterogeneous ways. As a result, routing methods that rely on fixed model representations and ignore RAG capabilities struggle to accurately model routing strategies under such conditions.

Table 16: Test accuracy (%) of RAGRouter and baselines on TriviaQA subsets with different types of retrieval noise, **bold** indicates best results.

Method	Golden Context	Relevant Noise	Irrelevant Noise	Counterfactual Noise	Avg
Oracle Single Best	95.00	83.33	**90.00**	83.33	87.92
KNN Router	96.67	78.33	83.33	86.67	86.25
GraphRouter	95.00	83.33	**90.00**	83.33	87.92
RouterDC	80.00	78.33	73.33	78.33	77.50
MF	95.00	76.67	80.00	80.00	82.92
RAGRouter	**98.33**	**86.67**	**90.00**	**88.33**	**90.83**

H Routing Performance on Summarization Task

To verify that RAGRouter is also applicable to RAG tasks beyond question answering, we conduct experiments on WikiASP [16], a summarization task commonly used in RAG-related research. Specifically, WikiASP aims to generate aspect-based summaries for Wikipedia entities and naturally supports the use of retrieved evidence. We adopt ROUGE-L as the evaluation metric.

For continuous performance metrics such as ROUGE-L, we extend RAGRouter's contrastive learning by score-based probabilistic sampling. Given each candidate LLM's normalized score $s \in [0, 1]$, we label it as "can answer" (positive, label 1) with probability s, and "cannot answer" (negative, label 0) with probability $1 - s$. This strategy is applied in our WikiASP experiments, enabling contrastive training based on continuous performance signals rather than discrete correctness labels.

Table 17: Performance of RAGRouter and baselines on WikiASP measured by ROUGE-L.

Method	WikiASP
Qwen2.5-0.5B-Instruct	0.1388
Qwen2.5-1.5B-Instruct	0.1411
Llama-3.2-3B-Instruct	0.1768
Qwen2.5-3B-Instruct	0.1528
Qwen2.5-7B-Instruct	0.1897
Llama-3.1-8B-Instruct	0.1455
KNN Router	0.1881
MF	0.1865
RAGRouter	**0.1981**

As shown in Table 17, RAGRouter outperforms non-RAG-aware routing baselines by more than 5.3%. These findings demonstrate that RAGRouter generalizes well to different RAG tasks.