

ICL-Router: In-Context Learned Model Representations for LLM Routing

Chenxu Wang^{1,2*†}, Hao Li^{2,3*†}, Yiqun Zhang^{2,4*}, Linyao Chen^{2,5*},
Jianhao Chen^{2,6*}, Ping Jian⁷, Qiaosheng Zhang², Shuyue Hu^{2‡}

¹School of Computer Science, Fudan University

²Shanghai Artificial Intelligence Laboratory

³School of Cybersecurity, Northwestern Polytechnical University

⁴Northeastern University

⁵The University of Tokyo

⁶Nanjing University

⁷Beijing Institute of Technology

{wangchenxu,zhangqiaosheng,hushuyue}@pjlab.org.cn, li.hao@mail.nwpu.edu.cn, yiqunzhang@stumail.neu.edu.cn,
chen-linyao217@g.ecc.u-tokyo.ac.jp, jhchen.nju@gmail.com, pjian@bit.edu.cn

Abstract

Large language models (LLMs) often exhibit complementary strengths. Model routing harnesses these strengths by dynamically directing each query to the most suitable model, given a candidate model pool. However, routing performance relies on accurate model representations, and adding new models typically requires retraining, limiting scalability. To address these challenges, we propose a novel routing method using in-context vectors to represent model capabilities. The method proceeds in two stages. First, queries are embedded and projected into vectors, with a projector and LLM-based router trained to reconstruct the original queries, aligning vector representations with the router’s semantic space. Second, each candidate model is profiled on a query set, and the router learns—based on in-context vectors of query and model performance—to predict whether each model can correctly answer new queries. Extensive experiments demonstrate that our method achieves state-of-the-art routing performance in both in-distribution and out-of-distribution tasks. Moreover, our method allows for seamless integration of new models without retraining the router.

Code — <https://github.com/lalalambf/ICL-Router>

Extended version — <https://arxiv.org/abs/2510.09719>

1 Introduction

Large language models (LLMs) have demonstrated exceptional capabilities performance across various tasks, including mathematical reasoning (Liu et al. 2024b; Chervonyi et al. 2025), code generation (DeepSeek-AI et al. 2024; Wang et al. 2025b), logical understanding (Liu et al.

2025a; Wang, Jian, and Yang 2025), and STEM problem-solving (Wu et al. 2025; Ma et al. 2025b). Despite these advances, *no* single model consistently outperforms others in every domain. Instead, models often exhibit complementary strengths and weaknesses, shaped by differences in their training data, architecture, and optimization.

An emerging line of research explores *routing* methods. The intuition is straightforward: if models excel in different tasks, then dynamically assigning the most capable model in a model pool to each query might maximize overall performance (Chen et al. 2024b; Lu et al. 2024; Zhuang et al. 2024a; Zhang, Zhan, and Ye 2025). However, in practice, achieving this goal is *far* from trivial. A key challenge is that *effective routing critically depends on an accurate understanding of each model’s capabilities*. For example, given two general-purpose models, it is difficult to tell which tasks LLaMA (Grattafiori et al. 2024) excels at versus those where Gemma (Team et al. 2025) performs better, unless both are *extensively* evaluated across a wide range of tasks. This challenge is further exacerbated by the rapid pace of LLM development. As new models are frequently released from time to time, routing methods must be able to *incrementally* adapt to these new models. Otherwise, the cost of repeating large-scale evaluations and retraining routers will quickly become prohibitive, ultimately limiting the ability of routing methods to benefit from model scaling.

Although some recent efforts have begun to address this challenge, existing approaches fall short of addressing scalability and evaluation efficiency. For instance, RouterDC (Chen et al. 2024b) uses dual contrastive learning to jointly train query and model embeddings, aligning each query with its optimal model. Similarly, EmbedLLM (Zhuang et al. 2024a) applies binary cross-entropy loss to train an embedding-based router that predicts query–model compatibility. However, both RouterDC and EmbedLLM operate on a fixed set of LLMs; incorporating a new model requires retraining the router. More recently, MODEL-SAT (Zhang, Zhan, and Ye 2025) represents mod-

*This work was done during his internship at Shanghai Artificial Intelligence Laboratory.

†These authors contributed equally.

‡Corresponding Author

Copyright © 2026, Association for the Advancement of Artificial Intelligence (www.aaai.org). All rights reserved.

els using a hand-crafted “capability instruction” based on performance on the MMLU benchmark (Hendrycks et al. 2021). This removes the need to retrain the router when adding new models, but requires manual instruction design for other benchmarks and prior knowledge of the capabilities they measure.

To this end, in this paper, we propose **ICL-Router**, a novel method that leverages in-context learned representations to characterize model capabilities. We hypothesize that a model’s performance on diverse queries can serve as in-context exemplars, as the way a model responds to various queries naturally reflects its unique capability profile. However, including hundreds or thousands of queries directly as in-context information would result in an unmanageably long context window. Inspired by the recent concept of *in-context vectors* (Zhuang et al. 2024b; Liu et al. 2024a), we instead transform these exemplars into compact in-context vectors, substantially reducing context length. At inference time, the router uses these vectors—compact representations of each model’s capability profile—to estimate the probability that each model can correctly answer a given query. When a new LLM is introduced, it is quickly evaluated on a small set of queries to generate in-context vectors (i.e. its capability profile), which is then fed to the router, enabling immediate inclusion without retraining.

More specifically, our ICL-Router consists of three main components: an embedding model, a projector, and an LLM-based router. It is trained in two stages, as shown in Figure 1. In the first stage, queries are embedded and projected into vector representations; the projector and the LLM-based router are co-trained to align their semantic spaces, allowing the router to interpret the information encoded in the vectors effectively. In the second stage, each candidate LLM in the model pool is evaluated on a small set of queries. The resulting performance is paired with the vector representations of the queries to form the model’s capability profile. Using these query-performance pairs as in-context vectors, the router is then trained to predict whether a given model can successfully handle new queries.

We evaluate the performance of ICL-Router on 10 widely used benchmarks (AGIEval, AIME, BBH, HumanEval, KORBench, LogicBench, MBPP, MMLU-CF, MMLUPro, and OlympiadBench) across both in-distribution and out-of-distribution scenarios. Experimental results show that ICL-Router achieves state-of-the-art (SOTA) routing performance. Specifically, on in-distribution tasks, ICL-Router outperforms the best-performing single LLM in the model pool (Deepseek-R1-distill-Qwen-7B) by 7.2 absolute points (Table 1); it also surpasses RouterDC by 3.9 points, EmbedLLM by 2.2 points, and MODEL-SAT by 4.6 points. Moreover, unlike EmbedLLM and RouterDC—which assume a fixed model pool and require retraining to incorporate new models—the ICL-Router seamlessly benefits from model scaling. As more models are added, it significantly outperforms MODEL-SAT (Figures 2 and 3), demonstrating superior scalability and adaptability. Beyond model scalability, the ICL-Router also benefits from incorporating a modest number of in-context exemplars, which further enhances routing performance as shown in Figures 4 and 5.

Our key contributions are summarized as follows:

- We propose characterizing model capabilities through vector representations of query-performance pairs, offering a new perspective on model representation.
- We propose a novel routing method, ICL-Router, that decouples model capability profiling from routing, enabling scalable integration of new models without retraining.
- Empirical results demonstrate that ICL-Router achieves SOTA routing performance across multiple domains in both in-distribution and OOD tasks.

2 Related Work

In-Context Learning. In-context learning (ICL) enables LLMs to perform new tasks using only demonstrations provided in the prompt. This capability was first demonstrated effectively by GPT-3 (Brown et al. 2020), which achieved strong few-shot and even zero-shot performance across diverse tasks. With LLMs supporting longer context windows, ICL performance has further improved, especially when hundreds or thousands of examples are provided (Li et al. 2023a; Chen et al. 2023; Bertsch et al. 2025). However, this also increases context length and inference cost. To address these challenges, recent work explores in-context vectors as a more efficient way to encode contextual information. For example, Zhuang et al. (2024b) propose Vector-ICL, which maps continuous inputs from various modalities into the model’s embedding space, enabling LLMs to learn from vectorized demonstrations. Similarly, Liu et al. (2024a) improve context selection and demonstration efficiency, reducing inference cost while preserving strong ICL performance. Building on these advances, our work investigates using in-context vectors as model representations for model routing.

Model Routing. Model routing aims to efficiently allocate queries to the optimal model without invoking all available models. Recent research generally falls into two categories: methods balancing performance and computational cost (Ong et al. 2024; Feng, Shen, and You 2024; Wang et al. 2025a; Jitkrittum et al. 2025; Zhang, Feng, and You 2025), and methods prioritizing model performance (Jiang, Ren, and Lin 2023; Lu et al. 2024; Chen et al. 2024b; Zhuang et al. 2024a; Zhang, Zhan, and Ye 2025; Zhang et al. 2025c). For the former, GraphRouter (Feng, Shen, and You 2024) adopts a graph-based strategy to reduce computational overhead while preserving accuracy, and Router-R1 (Zhang, Feng, and You 2025) frames routing as a sequential decision process, using an LLM-based router that iteratively “thinks” and “routes” to balance cost and performance. For the latter, EmbedLLM (Zhuang et al. 2024a) uses an encoder–decoder architecture to leverage compact model and query embeddings for routing prediction. MODEL-SAT (Zhang, Zhan, and Ye 2025) employs aptitude-based instruction tuning to characterize model capabilities for dynamic routing. However, existing performance-oriented approaches still face two main limitations: their model representations are often overly simplistic, and adding new models remains inefficient, typically requiring costly retraining and large-scale inference.

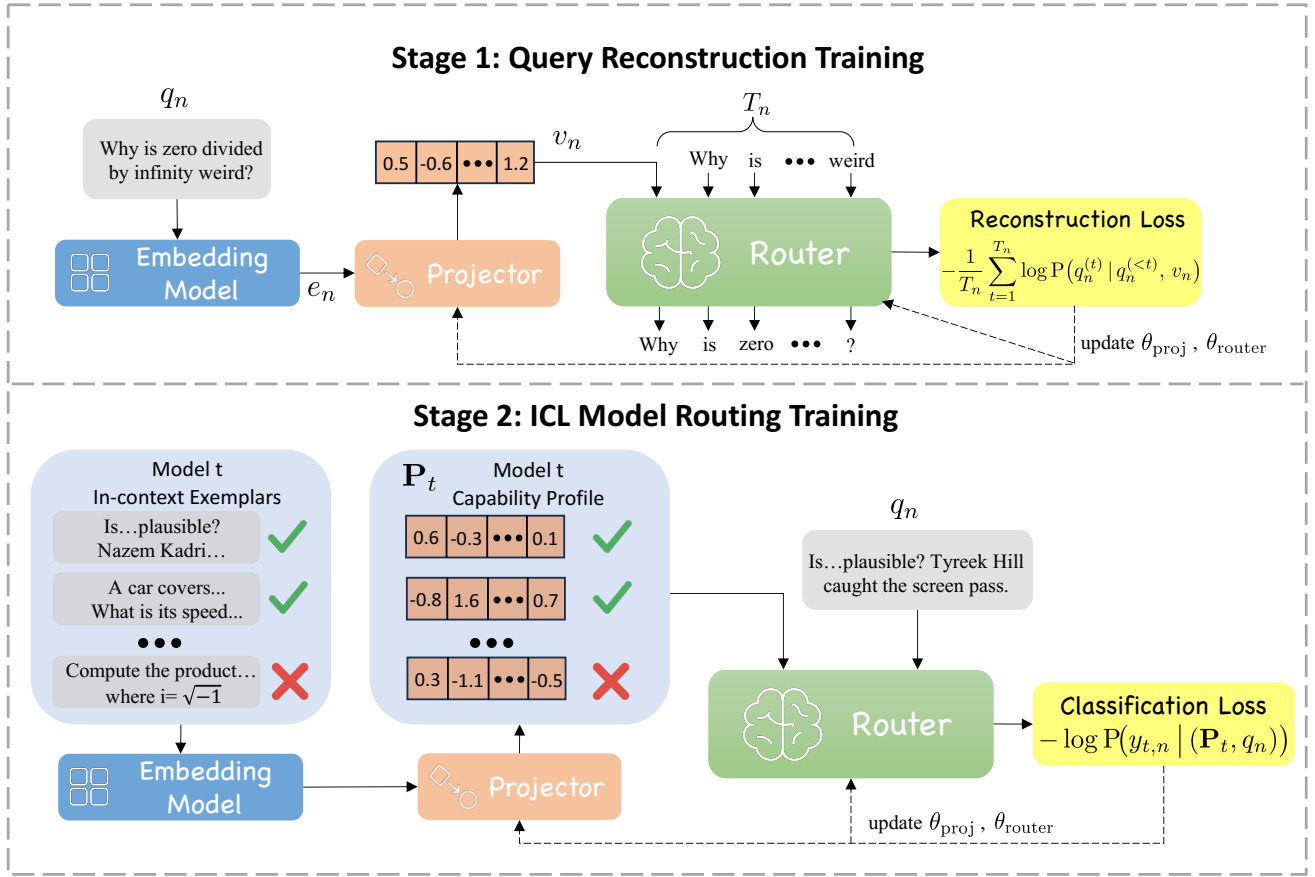


Figure 1: The two-stage ICL-Router framework. (1) Query Reconstruction Training: The projector is trained to align the embedding model and router dimensions, while the router reconstructs queries from projected vectors to learn their semantics. (2) ICL Model Routing Training: Each model’s capabilities are encoded as in-context vectors, and the router is trained to predict whether a given model can handle a specific query.

3 ICL-Router

3.1 Overview

We introduce ICL-Router, a framework for constructing in-context learned representations to support LLM routing. It comprises three main components: an embedding model, a projector, and an LLM-based router. Consider a set of candidate LLMs. At inference time, each query is first encoded using the embedding model and projector to obtain its vector representation. The router then takes this query vector along with the in-context vectors representing each candidate LLM’s capabilities as input, and selects the model best suited to handle the query. As illustrated in Figure 1, our approach features a two-stage training strategy: (1) **Query Reconstruction Training**, which aims to enable the projector align the dimension between the embedding model and the router, while also allowing the router to interpret the vectors generated by the embedding model. (2) **ICL Model Routing Training**, which utilizes the in-context model performance vectors as input to train the router to select the most suitable LLM for each query.

3.2 Query Reconstruction Training

The core objective of this stage is to co-train the projector and the router. The projector learns to map vectors from an embedding model into the routing model’s semantic space, while the router simultaneously learns to understand and interpret these incoming vector representations.

Given a set of queries $\mathcal{Q} = \{q_n \mid n = 1, \dots, N\}$, each query q_n is first encoded into an embedding by the embedding model f_{emb} . The embedding e_n corresponding to query q_n is defined as:

$$e_n = f_{emb}(q_n) \in \mathbb{R}^{d_{Emb}}. \quad (1)$$

Next, each embedding e_n is projected into a vector v_n by a projector f_{proj} , which is trained to align the embedding dimension with the LLM-based router’s input space:

$$v_n = f_{proj}(e_n) \in \mathbb{R}^{d_{Router}}. \quad (2)$$

The router is trained to reconstruct the original query q_n from its projected vector v_n in an autoregressive manner. Specifically, at each step, it maximizes the conditional probability of the current token $q_n^{(t)}$ given the previously generated tokens $q_n^{(<t)}$ and the projected vector v_n . The training

objective is thus defined by minimizing the reconstruction loss:

$$\mathcal{L}_{\text{rec}}(\theta_{\text{proj}}, \theta_{\text{router}}) = -\frac{1}{NT_n} \sum_{n=1}^N \sum_{t=1}^{T_n} \log P(q_n^{(t)} | q_n^{(<t)}, v_n), \quad (3)$$

where $q_n^{(t)}$ denotes the t -th token, $q_n^{(<t)}$ represents all preceding tokens, and T_n is the number of tokens. By jointly training the projector and router to minimize this reconstruction loss, we encourage the projection module to produce vectors aligned with the router’s semantic space, ensuring that projected queries are interpretable by the router for downstream routing.

3.3 ICL Model Routing Training

Building upon the vector representation learned during query reconstruction, this stage trains the router to assign each query to the most appropriate LLM. By leveraging in-context vectors that represent each LLM’s capabilities as input, the router learns to match queries with the LLM that is most likely to provide correct answers.

Consider a set of LLMs, denoted as $\mathbb{M} = \{\mathcal{M}_t : t = 1, \dots, T\}$, and a set of queries $\mathcal{Q} = \{q_k | k = 1, \dots, K\}$. For each model \mathcal{M}_t , we construct a capability profile \mathbf{P}_t by evaluating the model’s responses to each query in \mathcal{Q} . The query set \mathcal{Q} consists of challenging queries, specifically those that only a few LLMs are able to answer correctly. Formally, for each model \mathcal{M}_t , we define its capability profile \mathbf{P}_t as:

$$\mathbf{P}_t = ((v_1, c_1), (v_2, c_2), \dots, (v_K, c_K)), \quad (4)$$

where v_k is the vector representation of each query $q_k \in \mathcal{Q}$, which is encoded through the embedding model and the projector, and c_k indicates whether \mathcal{M}_t answered query q_k correctly, with ‘Yes’ and ‘No’ denoting correct and incorrect responses, respectively.

These capability profiles provide a detailed characterization of model capabilities. We jointly train the projector and the router to predict whether a model can accurately handle a query, conditioned on its capability profile. Formally, the projector and the router are optimized together using a cross-entropy loss:

$$\mathcal{L}_{\text{ce}}(\theta_{\text{proj}}, \theta_{\text{router}}) = -\frac{1}{TN} \sum_{t=1}^T \sum_{n=1}^N \log P(y_{t,n} | (\mathbf{P}_t, q_n)), \quad (5)$$

where T is the number of LLMs, N is the number of queries, and $y_{t,n}$ is the ground-truth label indicating whether model \mathcal{M}_t answers query q_n correctly. Conditioning on these vectors allows the router to reason about the relative strengths and weaknesses of each candidate model, without requiring an excessively long context window.

3.4 Inference and Scalable Model Incorporation

During inference, for each new query q' , the router combines it with each model’s capability profile \mathbf{P}_t and outputs the probability that model \mathcal{M}_t will answer correctly. The model with the highest predicted probability is selected:

$$\mathcal{M}^* = \arg \max_{t=1, \dots, T} p(\text{‘Yes’} | \mathcal{M}_t, q'), \quad (6)$$

where $p(\text{‘Yes’} | \mathcal{M}_t, q')$ denotes the router-estimated probability that \mathcal{M}_t will produce a correct answer for the query. When a new model \mathcal{M}_{T+1} is introduced, we simply evaluate it on the same query set \mathcal{Q} to construct its capability profile \mathbf{P}_{T+1} with the embedding model and the projector. The capability profile of this new model can then be incorporated into the routing process without any additional retraining. Therefore, our approach allows for rapid, plug-and-play integration of new LLMs in dynamic environments.

4 Experiments

4.1 Datasets

We evaluate our method on 10 benchmarks, including OlympiadBench (Chervonyi et al. 2025), AIME, MBPP (Austin et al. 2021), HumanEval (Zhong et al. 2023), BBH (Suzgun et al. 2022), LogicBench (Parmar et al. 2024), KORBench (Ma et al. 2025a), MMLUPro (Wang et al. 2024), AGIEval (Zhong et al. 2023), and MMLU-CF (Zhao et al. 2024). To evaluate the generalization of ICL-Router, we partition these benchmarks into in-distribution and out-of-distribution (OOD) sets. We designate AIME, HumanEval, KORBench, AGIEval and MMLU-CF as OOD datasets, reserving them exclusively for testing. Referring to the settings in RouterDC (Chen et al. 2024b), the other benchmarks are considered in-distribution and split into training and test sets at a 7:3 ratio.

4.2 Implementation Settings

Our framework is composed of three main components: an embedding model, a projector, and an LLM-based router. We adopt *Qwen3-Embedding-8B* (Zhang et al. 2025d) as the embedding model; notably, this model is used solely to generate query vectors and does not participate in the training process. The projector is implemented as a two-layer multi-layer perceptron (MLP) that facilitates the transformation between the embedding space and the router. For the router, we employ *Qwen2.5-7B-Instruct*, given its good performance and versatility. For the LLM pool, we consider eight widely-used open-source LLMs.

Our training process consists of two stages. In the first stage, we begin by training only the projector for one epoch with a learning rate of $2e-5$. This is followed by two epochs of joint training, where the router is introduced with a learning rate of $5e-6$. To construct the set of 500 challenging queries, we selected 125 queries each where exactly 1, 2, 3, or 4 models (given a pool of 8 models) produced the correct answer. The intuition is that all-correct or all-wrong queries fail to differentiate model capability. We jointly train the projector and router for five epochs with learning rates of $1e-5$ and $2e-6$, respectively. The training is repeated three times using different random seeds. For both stages, we set the batch size to 32. To ensure stable results during evaluation, we sample each routed LLM 10 times, setting the temperature to 0.3 and top- p to 1.0, and then report the average accuracy.

4.3 Baselines

We compare **ICL-Router** with the following baselines:

Method	OlympiadBench	BBH	LogicBench	MMLUPro	MBPP	Avg.
DeepSeek-R1-Distill-Qwen-7B	66.24	72.87	78.03	58.84	69.52	69.10
Llama-3.1-8B-Instruct	16.93	59.47	69.47	49.38	60.03	51.06
Llama-3.1-Nemotron-Nano-8B-v1	74.26	53.24	65.43	50.40	79.21	64.51
Qwen2.5-7B-Instruct	38.02	71.54	71.93	57.49	71.99	62.19
cogito-v1-preview-llama-8B	16.58	75.62	68.43	58.04	60.82	55.90
Gemma-2-9B-IT	13.71	62.19	68.27	55.42	60.31	51.98
Internlm3-8B-Instruct	33.12	68.64	72.83	56.49	56.88	57.59
GLM-4-9B-Chat	16.88	51.14	69.53	48.58	61.10	49.65
Random Router	34.47	64.34	71.41	54.33	64.98	57.91
LLM Router	46.03	69.54	73.19	55.87	66.55	62.36
Max Expert	74.26	75.62	78.03	58.84	79.21	73.19
RouterDC	73.56	73.49	77.24	58.20	<u>79.11</u>	72.32
EmbedLLM	71.45	<u>79.02</u>	<u>78.92</u>	<u>64.06</u>	<u>77.34</u>	<u>74.16</u>
MODEL-SAT	73.02	71.14	74.80	63.61	76.00	71.71
ICL-Router (ours)	<u>74.16</u>	80.52	79.03	67.53	80.53	76.30

Table 1: Comparison of ICL-Router with baselines on in-distribution tasks. The best is in **bold** and the second-best is underlined.

Method	AIME	KORBench	MMLU-CF	AGIEval	HumanEval	Avg.
DeepSeek-R1-Distill-Qwen-7B	45.50	50.30	56.70	62.56	83.78	59.77
Llama-3.1-8B-Instruct	3.67	43.67	62.16	54.40	65.43	45.87
Llama-3.1-Nemotron-Nano-8B-v1	58.00	30.91	49.76	47.64	91.10	55.48
Qwen2.5-7B-Instruct	8.00	38.31	64.98	62.23	84.27	57.58
cogito-v1-preview-llama-8B	2.67	44.99	62.11	58.56	67.80	50.83
Gemma-2-9B-IT	10.00	37.16	64.33	60.60	63.23	51.82
Internlm3-8B-Instruct	5.83	38.53	64.58	64.01	67.98	52.99
GLM-4-9B-Chat	1.67	35.67	60.76	59.11	67.68	50.18
Random Router	15.79	37.80	60.67	59.02	73.91	49.44
LLM Router	30.33	40.67	60.48	58.08	75.20	52.95
Max Expert	58.00	<u>50.30</u>	64.98	64.01	91.10	<u>65.68</u>
RouterDC	58.00	46.44	59.50	61.49	<u>90.20</u>	63.13
EmbedLLM	53.72	46.90	65.22	64.99	83.29	62.82
MODEL-SAT	<u>56.84</u>	44.53	64.45	60.08	89.02	62.99
ICL-Router (ours)	58.00	53.03	<u>64.99</u>	67.29	89.04	66.47

Table 2: Comparison of ICL-Router with baselines on OOD tasks. The best is in **bold** and the second-best is underlined.

- **Random Router**: Selects a model at random from the pool of candidate LLMs for each query.
- **LLM Router**: A prompt-based approach that leverages an LLM (Qwen2.5-7B-Instruct) to choose models based on natural-language descriptions of their performance profiles.
- **Max Expert**: Serves as a strong baseline by selecting the best-performing model for each dataset.
- **RouterDC** (Chen et al. 2024b): Trains query and model embeddings using dual contrastive learning, pulling queries closer to suitable models and clustering semantically similar queries within the representation space.
- **EmbedLLM** (Zhuang et al. 2024a): An encoder-decoder framework that employs compact model and query embeddings to predict model-query compatibility, with the router trained via binary cross-entropy loss.

- **MODEL-SAT** (Zhang, Zhan, and Ye 2025): Leverages capability instruction tuning by converting candidate model performance into textual descriptions, which are embedded and passed to a trainable LLM that dynamically predicts the most suitable model for each query.

4.4 Main Results

Table 1 presents the performance comparison across five in-distribution datasets. **ICL-Router achieves the highest average accuracy of 76.30%, outperforming RouterDC by 4.08%, EmbedLLM by 2.14% and MODEL-SAT by 4.59%.** Notably, ICL-Router attains the best results on 4 out of the 5 datasets, demonstrating its consistent superiority across a diverse range of tasks. Furthermore, ICL-Router surpasses the Max Expert—which selects the best-performing model for each dataset—by an average of 3.11%. This improvement is primarily driven by substantial

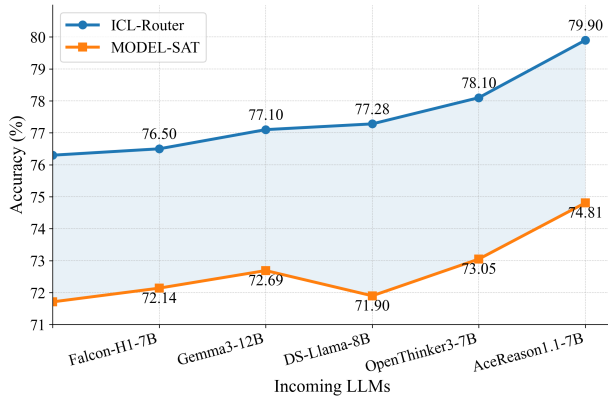


Figure 2: Effects of integrating new LLMs on in-distribution routing performance.

accuracy gains on several tasks: MMLUPro (+8.69%), BBH (+4.9%), LogicBench (+1%), and MBPP (+1.32%). These results indicate that ICL-Router can match each query to the most suitable model based on its specific characteristics, rather than relying solely on domain-level or task-based selection.

Table 2 shows the performance comparison across five OOD tasks. Remarkably, **ICL-Router achieves an average accuracy of 66.47%, surpassing RouterDC by 3.34%, EmbedLLM by 3.65% and MODEL-SAT by 3.48%**. This result highlights ICL-Router’s robust performance across diverse OOD scenarios, confirming its effectiveness beyond in-distribution settings. Moreover, ICL-Router significantly outperforms the Max Expert baseline on KORBench (+2.73%) and AGIEval (+3.28%), while achieving nearly identical results on AIME, MMLU-CF, and HumanEval. This demonstrates that ICL-Router not only generalizes well across OOD tasks, but also reliably matches or surpasses the performance of the strongest individual model on each benchmark.

Overall, the results demonstrate that ICL-Router consistently outperforms all baseline methods on both in-distribution and OOD tasks. This underscores the strength of our approach in introducing accurate and semantically rich model representations, which effectively capture subtle differences in the capabilities of candidate models and enable more precise routing. The robust and stable performance observed across benchmarks further validates the effectiveness and generalization ability of our method.

4.5 Scalability to New Models

We conduct experiments to assess the scalability of our method on a set of recently released and competitive LLMs, including Falcon-H1-7B-Instruct (Zuo et al. 2025), Gemma3-12B-IT (Team et al. 2025), DeepSeek-R1-Llama-8B (Guo et al. 2025), OpenThinker3-7B (Guha et al. 2025), and AceReason-Nemotron-1.1-7B (Liu et al. 2025b). As illustrated in Figures 2 and 3, ICL-Router demonstrates a consistent upward trajectory in accuracy for both in-distribution

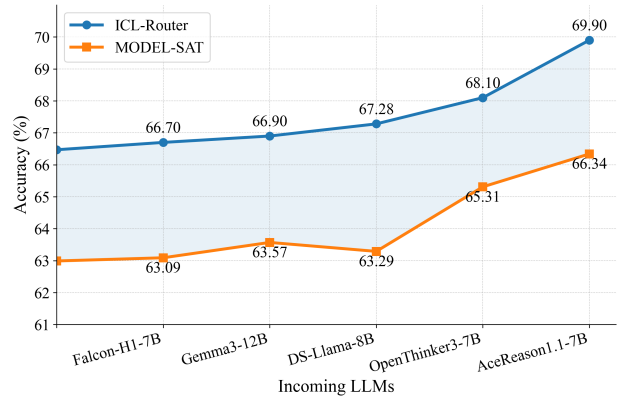


Figure 3: Effects of integrating new LLMs on out-of-distribution (OOD) routing performance.

Embed.	Para./Dim.	ID	OOD
mxbai-embed-large-v1	0.3B/1024	74.45	64.55
bge-m3	0.6B/1024	75.13	65.14
stella-en-1.5B-v5	1.5B/2048	75.51	65.45
gte-Qwen2-7B-instruct	7B/3584	<u>76.03</u>	<u>66.06</u>
Qwen3-8B-Embedding	8B/4096	76.30	66.47

Table 3: Embedding model comparison in terms of parameter size and embedding dimension. Performance is evaluated on both in-distribution (ID) and out-of-distribution (OOD) scenarios.

and out-of-distribution (OOD) scenarios. For example, in the in-distribution setting, ICL-Router’s accuracy improves from 76.3% to 79.9% as more models are integrated; a similar trend is observed on OOD tasks, where performance rises from 66.4% to 69.9%. In contrast, MODEL-SAT exhibits less stable gains, with accuracy improvements that are inconsistent and sometimes fluctuate as additional models are introduced.

Importantly, at higher accuracy levels, where further improvements are typically more difficult, **ICL-Router consistently maintains and even expands its performance lead as more models are added**. This steady and reliable progress demonstrates that our method effectively leverages the strengths of an expanding model pool. These results show that our approach remains robust and dependable, even as new models are continually integrated in real-world settings.

4.6 Analysis

To better understand the effectiveness of ICL-Router, we analyze how its performance is influenced by three key elements — the embedding model, the number of in-context exemplars, and the query reconstruction training stage.

Embedding Model. The choice of embedding model plays a pivotal role in determining the quality of vector representations used for downstream routing. Spe-

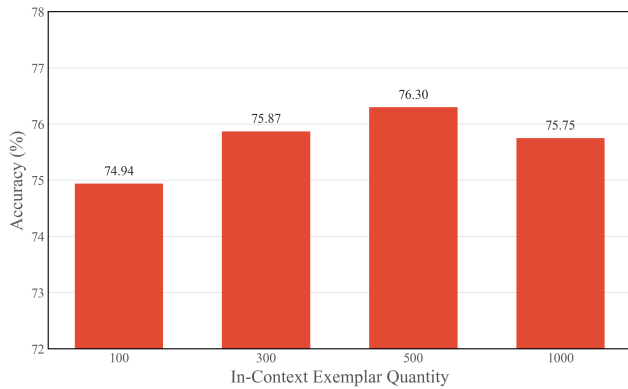


Figure 4: Effects of in-Context exemplar quantity on in-distribution performance.

cially, we evaluate the impact of various embedding models on both in-distribution and OOD accuracy by comparing five embedding models that differ in parameter size and embedding dimensions, including *mxbai-embed-large-v1* (Lee et al. 2024), *bge-m3* (Chen et al. 2024a), *stella-en-1.5B-v5* (Zhang et al. 2025a), *gte-Qwen2-7B-instruct* (Li et al. 2023b), *Qwen3-8B-Embedding* (Zhang et al. 2025d). As shown in Table 3, a clear trend emerges: as model size increases—from the smallest, *mxbai-embed-large-v1* (0.3B/1024), to the largest, *Qwen3-8B-Embedding* (8B/4096)—in-distribution accuracy improves from 74.45% to 76.30%, while out-of-distribution (OOD) accuracy rises from 64.55% to 66.47%. This stepwise improvement across all five evaluated models indicates that **stronger embedding models consistently produce more robust and generalizable vector representations**, leading to better downstream routing performance.

Effects of In-Context Exemplar Quantity. To further investigate the impact of the number of in-context exemplars on routing performance, we conduct ablation experiments by varying the quantity of exemplars used to construct each model’s capability profile. Specifically, we assess performance with 100, 300, 500, and 1000 in-context exemplars, analyzing both in-distribution and out-of-distribution (OOD) scenarios. As shown in Figures 4 and 5, increasing the number of in-context exemplars consistently improves both in-distribution and out-of-distribution routing accuracy up to a certain point. For in-distribution tasks, accuracy increases from 74.94% with 100 exemplars to a peak of 76.30% at 500 exemplars, before slightly dropping to 75.75% at 1000 exemplars. A similar trend is observed for out-of-distribution performance, where accuracy rises from 65.36% (100 exemplars) to 66.47% (500 exemplars), but then plateaus or slightly declines to 65.71% with 1000 exemplars. These results suggest that while **adding more exemplars generally enriches model capability profiles and enhances routing, there is a point of diminishing returns—likely due to redundancy or increased noise as the exemplar set grows too large**. This observation is consistent with findings from recent studies on many-shot in-context learning (Agarwal et al. 2024; Li et al. 2024; Zhang

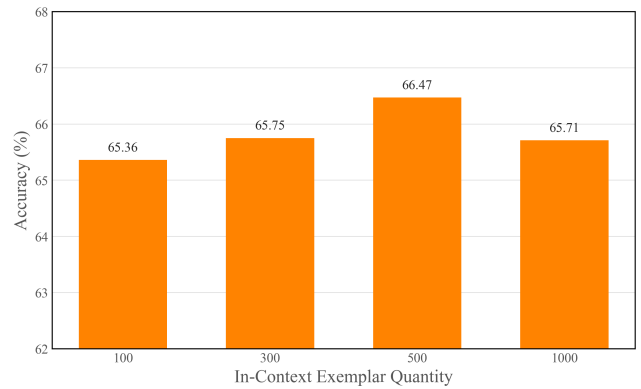


Figure 5: Effects of in-context exemplar quantity on OOD performance.

Method	ID	OOD
w/o QRT	74.01	64.06
ICL-Router	76.30	66.47

Table 4: Ablation study of query reconstruction training (QRT) on in-distribution (ID) and OOD scenarios.

et al. 2025b), which report that including too many context examples can sometimes degrade performance rather than improve it. Overall, using a moderate number of exemplars strikes the best balance between informativeness and efficiency, supporting robust and generalizable routing performance.

Effects of Query Reconstruction Training. Table 4 demonstrates that **removing query reconstruction training (QRT) leads to a clear decline in performance on both in-distribution and OOD tasks, with accuracy dropping by 2.29% and 2.41%**, respectively. This result highlights the importance of QRT for aligning the embedding and router spaces, enabling the router to more effectively leverage query representations during downstream routing. Without this training stage, the router is less able to capture and differentiate the relevant semantics required for accurate model selection.

5 Conclusion

In this paper, we propose a new insight on model representation for LLM routing by characterizing model capabilities through in-context vectors. By leveraging compact vector-based profiles that summarize how each LLM performs on challenging queries, our approach constructs semantically rich model representations that capture the diverse capabilities of LLMs. ICL-Router decouples capability profiling from routing, allowing plug-and-play integration of new models without retraining the router. Empirical results on 10 benchmarks demonstrate that ICL-Router achieves state-of-the-art performance across both in-distribution and OOD settings, demonstrating the effectiveness and scalability of in-context learned model representations for LLM routing.

Acknowledgments

This work was supported by the National Natural Science Foundation of China (Grant No. 6250076060). This work was supported by the Shanghai Municipal Science and Technology Major Project.

References

- Agarwal, R.; Singh, A.; Zhang, L.; Bohnet, B.; Rosias, L.; Chan, S.; Zhang, B.; Anand, A.; Abbas, Z.; Nova, A.; et al. 2024. Many-shot in-context learning. *Advances in Neural Information Processing Systems*, 37: 76930–76966.
- Austin, J.; Odena, A.; Nye, M.; Bosma, M.; Michalewski, H.; Dohan, D.; Jiang, E.; Cai, C.; Terry, M.; Le, Q.; and Sutton, C. 2021. Program Synthesis with Large Language Models. arXiv:2108.07732.
- Bertsch, A.; Ivgi, M.; Xiao, E.; Alon, U.; Berant, J.; Gormley, M. R.; and Neubig, G. 2025. In-Context Learning with Long-Context Models: An In-Depth Exploration. In *Proceedings of the 2025 Conference of the Nations of the Americas Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, 12119–12149.
- Brown, T.; Mann, B.; Ryder, N.; Subbiah, M.; Kaplan, J. D.; Dhariwal, P.; Neelakantan, A.; Shyam, P.; Sastry, G.; Askell, A.; et al. 2020. Language models are few-shot learners. *Advances in neural information processing systems*, 33: 1877–1901.
- Chen, J.; Chen, L.; Zhu, C.; and Zhou, T. 2023. How Many Demonstrations Do You Need for In-context Learning? In Bouamor, H.; Pino, J.; and Bali, K., eds., *Findings of the Association for Computational Linguistics: EMNLP 2023*, 11149–11159. Singapore: Association for Computational Linguistics.
- Chen, J.; Xiao, S.; Zhang, P.; Luo, K.; Lian, D.; and Liu, Z. 2024a. BGE M3-Embedding: Multi-Lingual, Multi-Functionality, Multi-Granularity Text Embeddings Through Self-Knowledge Distillation. arXiv:2402.03216.
- Chen, S.; Jiang, W.; Lin, B.; Kwok, J.; and Zhang, Y. 2024b. Routerdc: Query-based router by dual contrastive learning for assembling large language models. *Advances in Neural Information Processing Systems*, 37: 66305–66328.
- Chervonyi, Y.; Trinh, T. H.; Olšák, M.; Yang, X.; Nguyen, H.; Menegali, M.; Jung, J.; Verma, V.; Le, Q. V.; and Luong, T. 2025. Gold-medalist Performance in Solving Olympiad Geometry with AlphaGeometry2. arXiv:2502.03544.
- DeepSeek-AI; Zhu, Q.; Guo, D.; Shao, Z.; Yang, D.; Wang, P.; Xu, R.; Wu, Y.; Li, Y.; Gao, H.; Ma, S.; Zeng, W.; Bi, X.; Gu, Z.; Xu, H.; Dai, D.; Dong, K.; Zhang, L.; Piao, Y.; Gou, Z.; Xie, Z.; Hao, Z.; Wang, B.; Song, J.; Chen, D.; Xie, X.; Guan, K.; You, Y.; Liu, A.; Du, Q.; Gao, W.; Lu, X.; Chen, Q.; Wang, Y.; Deng, C.; Li, J.; Zhao, C.; Ruan, C.; Luo, F.; and Liang, W. 2024. DeepSeek-Coder-V2: Breaking the Barrier of Closed-Source Models in Code Intelligence. arXiv:2406.11931.
- Feng, T.; Shen, Y.; and You, J. 2024. Graphrouter: A graph-based router for llm selections. arXiv preprint arXiv:2410.03834.
- Grattafiori, A.; Dubey, A.; Jauhri, A.; Pandey, A.; Kadian, A.; Al-Dahle, A.; Letman, A.; Mathur, A.; Schelten, A.; Vaughan, A.; et al. 2024. The llama 3 herd of models. arXiv preprint arXiv:2407.21783.
- Guha, E.; Marten, R.; Keh, S.; Raoof, N.; Smyrnis, G.; Bansal, H.; Nezhurina, M.; Mercat, J.; Vu, T.; Sprague, Z.; Suvama, A.; Feuer, B.; Chen, L.; Khan, Z.; Frankel, E.; Grover, S.; Choi, C.; Muennighoff, N.; Su, S.; Zhao, W.; Yang, J.; Pimpalgaonkar, S.; Sharma, K.; Ji, C. C.-J.; Deng, Y.; Pratt, S.; Ramanujan, V.; Saad-Falcon, J.; Li, J.; Dave, A.; Albalak, A.; Arora, K.; Wulfe, B.; Hegde, C.; Durrett, G.; Oh, S.; Bansal, M.; Gabriel, S.; Grover, A.; Chang, K.-W.; Shankar, V.; Gokaslan, A.; Merrill, M. A.; Hashimoto, T.; Choi, Y.; Jitsev, J.; Heckel, R.; Sathiamoorthy, M.; Dimakis, A. G.; and Schmidt, L. 2025. OpenThoughts: Data Recipes for Reasoning Models. arXiv:2506.04178.
- Guo, D.; Yang, D.; Zhang, H.; Song, J.; Zhang, R.; Xu, R.; Zhu, Q.; Ma, S.; Wang, P.; Bi, X.; et al. 2025. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning. arXiv preprint arXiv:2501.12948.
- Hendrycks, D.; Burns, C.; Basart, S.; Zou, A.; Mazeika, M.; Song, D.; and Steinhardt, J. 2021. Measuring Massive Multitask Language Understanding. *Proceedings of the International Conference on Learning Representations (ICLR)*.
- Jiang, D.; Ren, X.; and Lin, B. Y. 2023. LLM-Blender: Ensembling Large Language Models with Pairwise Comparison and Generative Fusion. In *Proceedings of the 61th Annual Meeting of the Association for Computational Linguistics (ACL 2023)*.
- Jitkrittum, W.; Narasimhan, H.; Rawat, A. S.; Juneja, J.; Wang, Z.; Lee, C.-Y.; Shenoy, P.; Panigrahy, R.; Menon, A. K.; and Kumar, S. 2025. Universal model routing for efficient llm inference. arXiv preprint arXiv:2502.08773.
- Lee, S.; Shakir, A.; Koenig, D.; and Lipp, J. 2024. Open Source Strikes Bread - New Fluffy Embeddings Model.
- Li, M.; Gong, S.; Feng, J.; Xu, Y.; Zhang, J.; Wu, Z.; and Kong, L. 2023a. In-Context Learning with Many Demonstration Examples. arXiv:2302.04931.
- Li, T.; Zhang, G.; Do, Q. D.; Yue, X.; and Chen, W. 2024. Long-context llms struggle with long in-context learning. arXiv preprint arXiv:2404.02060.
- Li, Z.; Zhang, X.; Zhang, Y.; Long, D.; Xie, P.; and Zhang, M. 2023b. Towards general text embeddings with multi-stage contrastive learning. arXiv preprint arXiv:2308.03281.
- Liu, S.; Ye, H.; Xing, L.; and Zou, J. 2024a. In-context vectors: making in context learning more effective and controllable through latent space steering. In *Proceedings of the 41st International Conference on Machine Learning*, 32287–32307.
- Liu, T.; Xu, W.; Huang, W.; Zeng, Y.; Wang, J.; Wang, X.; Yang, H.; and Li, J. 2025a. Logic-of-Thought: Injecting Logic into Contexts for Full Reasoning in Large Language Models. arXiv:2409.17539.
- Liu, Z.; Chen, Y.; Shoeybi, M.; Catanzaro, B.; and Ping, W. 2024b. AceMath: Advancing Frontier Math Reasoning with Post-Training and Reward Modeling. arXiv preprint.

- Liu, Z.; Yang, Z.; Chen, Y.; Lee, C.; Shoeybi, M.; Catanzaro, B.; and Ping, W. 2025b. AceReason-Nemotron 1.1: Advancing Math and Code Reasoning through SFT and RL Synergy. *arXiv preprint arXiv:2506.13284*.
- Lu, K.; Yuan, H.; Lin, R.; Lin, J.; Yuan, Z.; Zhou, C.; and Zhou, J. 2024. Routing to the Expert: Efficient Reward-guided Ensemble of Large Language Models. In Duh, K.; Gomez, H.; and Bethard, S., eds., *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, 1964–1974. Mexico City, Mexico: Association for Computational Linguistics.
- Ma, K.; Du, X.; Wang, Y.; Zhang, H.; Wen, Z.; Qu, X.; Yang, J.; Liu, J.; Liu, M.; Yue, X.; Huang, W.; and Zhang, G. 2025a. KOR-Bench: Benchmarking Language Models on Knowledge-Orthogonal Reasoning Tasks. *arXiv:2410.06526*.
- Ma, X.; Liu, Q.; Jiang, D.; Zhang, G.; Ma, Z.; and Chen, W. 2025b. General-Reasoner: Advancing LLM Reasoning Across All Domains. *arXiv:2505.14652*.
- Ong, I.; Almahairi, A.; Wu, V.; Chiang, W.-L.; Wu, T.; Gonzalez, J. E.; Kadous, M. W.; and Stoica, I. 2024. RouteLLM: Learning to Route LLMs from Preference Data. In *The Thirteenth International Conference on Learning Representations*.
- Parmar, M.; Patel, N.; Varshney, N.; Nakamura, M.; Luo, M.; Mashetty, S.; Mitra, A.; and Baral, C. 2024. LogicBench: Towards Systematic Evaluation of Logical Reasoning Ability of Large Language Models. *arXiv:2404.15522*.
- Suzgun, M.; Scales, N.; Schärli, N.; Gehrmann, S.; Tay, Y.; Chung, H. W.; Chowdhery, A.; Le, Q. V.; Chi, E. H.; Zhou, D.; and Wei, J. 2022. Challenging BIG-Bench Tasks and Whether Chain-of-Thought Can Solve Them. *arXiv:2210.09261*.
- Team, G.; Kamath, A.; Ferret, J.; Pathak, S.; Vieillard, N.; Merhej, R.; Perrin, S.; Matejovicova, T.; Ramé, A.; Rivière, M.; et al. 2025. Gemma 3 technical report. *arXiv preprint arXiv:2503.19786*.
- Wang, C.; Jian, P.; and Yang, Z. 2025. Thought-Path Contrastive Learning via Premise-Oriented Data Augmentation for Logical Reading Comprehension. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 39, 25345–25352.
- Wang, X.; Liu, Y.; Cheng, W.; Zhao, X.; Chen, Z.; Yu, W.; Fu, Y.; and Chen, H. 2025a. MixLLM: Dynamic Routing in Mixed Large Language Models. In *Proceedings of the 2025 Conference of the Nations of the Americas Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, 10912–10922.
- Wang, Y.; Ma, X.; Zhang, G.; Ni, Y.; Chandra, A.; Guo, S.; Ren, W.; Arulraj, A.; He, X.; Jiang, Z.; Li, T.; Ku, M.; Wang, K.; Zhuang, A.; Fan, R.; Yue, X.; and Chen, W. 2024. MMLU-Pro: A More Robust and Challenging Multi-Task Language Understanding Benchmark. *arXiv:2406.01574*.
- Wang, Y.; Yang, L.; Tian, Y.; Shen, K.; and Wang, M. 2025b. Co-Evolving LLM Coder and Unit Tester via Reinforcement Learning. *arXiv:2506.03136*.
- Wu, X. J.; Zhang, Z.; Wen, Z.; Zhang, Z.; Ren, W.; Shi, L.; Chen, C.; Zhao, D.; Wang, Q.; Han, X.; et al. 2025. SHARP: Synthesizing High-quality Aligned Reasoning Problems for Large Reasoning Models Reinforcement Learning. *arXiv preprint arXiv:2505.14147*.
- Zhang, D.; Li, J.; Zeng, Z.; and Wang, F. 2025a. Jasper and Stella: distillation of SOTA embedding models. *arXiv:2412.19048*.
- Zhang, H.; Feng, T.; and You, J. 2025. Router-R1: Teaching LLMs Multi-Round Routing and Aggregation via Reinforcement Learning. *arXiv preprint arXiv:2506.09033*.
- Zhang, X.; Lv, A.; Liu, Y.; Sung, F.; Liu, W.; Luan, J.; Shang, S.; Chen, X.; and Yan, R. 2025b. More is not always better? enhancing many-shot in-context learning with differentiated and reweighting objectives. *arXiv preprint arXiv:2501.04070*.
- Zhang, Y.; Li, H.; Wang, C.; Chen, L.; Zhang, Q.; Ye, P.; Feng, S.; Wang, D.; Wang, Z.; Wang, X.; et al. 2025c. The Avengers: A Simple Recipe for Uniting Smaller Language Models to Challenge Proprietary Giants. *arXiv preprint arXiv:2505.19797*.
- Zhang, Y.; Li, M.; Long, D.; Zhang, X.; Lin, H.; Yang, B.; Xie, P.; Yang, A.; Liu, D.; Lin, J.; Huang, F.; and Zhou, J. 2025d. Qwen3 Embedding: Advancing Text Embedding and Reranking Through Foundation Models. *arXiv preprint arXiv:2506.05176*.
- Zhang, Y.-K.; Zhan, D.-C.; and Ye, H.-J. 2025. Capability instruction tuning: A new paradigm for dynamic llm routing. *arXiv preprint arXiv:2502.17282*.
- Zhao, Q.; Huang, Y.; Lv, T.; Cui, L.; Sun, Q.; Mao, S.; Zhang, X.; Xin, Y.; Yin, Q.; Li, S.; and Wei, F. 2024. MMLU-CF: A Contamination-free Multi-task Language Understanding Benchmark. *arXiv:2412.15194*.
- Zhong, W.; Cui, R.; Guo, Y.; Liang, Y.; Lu, S.; Wang, Y.; Saied, A.; Chen, W.; and Duan, N. 2023. AGIEval: A Human-Centric Benchmark for Evaluating Foundation Models. *arXiv:2304.06364*.
- Zhuang, R.; Wu, T.; Wen, Z.; Li, A.; Jiao, J.; and Ramchandran, K. 2024a. EmbedLLM: Learning Compact Representations of Large Language Models. *arXiv preprint arXiv:2410.02223*.
- Zhuang, Y.; Singh, C.; Liu, L.; Shang, J.; and Gao, J. 2024b. Vector-ICL: In-context Learning with Continuous Vector Representations. *arXiv preprint arXiv:2410.05629*.
- Zuo, J.; Velikanov, M.; Chahed, I.; Belkada, Y.; Rhayem, D. E.; Kunsch, G.; Hacid, H.; Yous, H.; Farhat, B.; Khadraoui, I.; Farooq, M.; Campesan, G.; Cojocaru, R.; Djilali, Y.; Hu, S.; Chaabane, I.; Khanna, P.; Seddik, M. E. A.; Huynh, N. D.; Khac, P. L.; AlQadi, L.; Mokeddem, B.; Chami, M.; Abubaker, A.; Lubinets, M.; Piskorski, K.; and Frikha, S. 2025. Falcon-H1: A Family of Hybrid-Head Language Models Redefining Efficiency and Performance. *arXiv preprint arXiv:2507.22448*.