RouterEval: A Comprehensive Benchmark for Routing LLMs to Explore Model-level Scaling Up in LLMs

Zhongzhan Huang[1], Guoming Ling[1], Yupei Lin[1], Yandong Chen[1], Shanshan Zhong[1], Hefeng Wu[1], Liang Lin[†1]
[1]Sun Yat-sen University [†]Corresponding author

Abstract

Routing large language models (LLMs) is a new paradigm that uses a router to recommend the best LLM from a pool of candidates for a given input. In this paper, our comprehensive analysis with more than 8,500 LLMs reveals a novel model-level scaling up phenomenon in Routing LLMs, i.e., a capable router can significantly enhance the performance of this paradigm as the number of candidates increases. This improvement can even surpass the performance of the best single model in the pool and many existing strong LLMs, confirming it a highly promising paradigm. However, the lack of comprehensive and open-source benchmarks for Routing LLMs has hindered the development of routers. In this paper, we introduce RouterEval , a benchmark tailored for router research, which includes over 200,000,000 performance records for 12 popular LLM evaluations across various areas such as commonsense reasoning, semantic understanding, etc., based on over 8,500 various LLMs. Using RouterEval, extensive evaluations of existing Routing LLM methods reveal that most still have significant room for improvement. See project page for all data, code and tutorial.

1 Introduction

Routing LLMs (Jitkrittum et al., 2025; Lu et al., 2023a; Zhao et al., 2024a; Shnitzer et al., 2023a; Chen et al., 2024a) aims to establish an efficient router capable of selecting an LLM from a pool of candidates with varying abilities to effectively handle a given input, in order to achieve specific goals such as high overall accuracy, low computational cost, or minimal hallucination, as shown in Fig. 1. Since this paradigm only involves input assignment, it is naturally compatible with heterogeneous LLMs of different structures, as well as most model enhancement methods, including fine-tuning (Hu et al., 2022; Zhang et al., 2023b), Mixture-of-Experts (Jiang et al., 2024; Zhong et al., 2024a), model merging (Goddard et al., 2024; Yang et al., 2024), etc. This allows for the creation of a sufficiently large LLM candidate pool. In this paper, through extensive experiments, we find a model-level scaling up phenomenon in LLMs, i.e., using some capable router, the rapid performance improvement as the number of candidates in the LLM pool increases, can even easily surpass the performance of the best single model in the pool and most existing strong LLMs. Moreover, the majority of LLMs in the candidate pool are relatively small and open-source models whose individual capabilities are far below those of commercialized LLMs, like GPT-4 (Achiam et al., 2023), which demonstrates Routing LLM is a highly promising paradigm (See Section 3 for details).

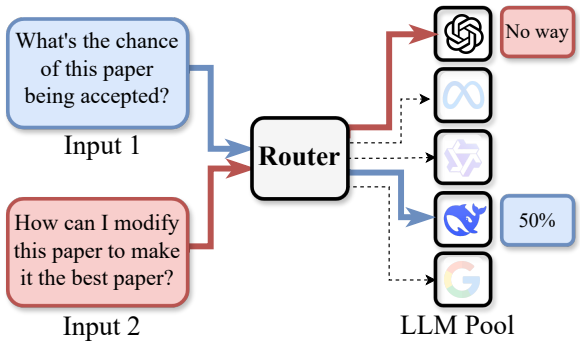

Figure 1: **The Overview of Routing LLMs**. For each given input, the router distributes it to the appropriate LLM to achieve specific objectives, such as high accuracy, low computational cost, reduced hallucinations, etc. We find that Routing LLMs is a promising paradigm for LLMs to achieve model-level scaling up.

However, current Routing LLMs methods are still in the early stages of rapid development and lack comprehensive and open-source benchmarks specifically designed for routers, which hinders their progress. For example, some existing benchmarks (Lu et al., 2024a; Hu et al., 2024a) still have the issues of insufficient LLM candidates, limited evaluation diversity, closed-source or inad-

Findings of the Association for Computational Linguistics: EMNLP 2025, pages 3860–3887
November 4-9, 2025

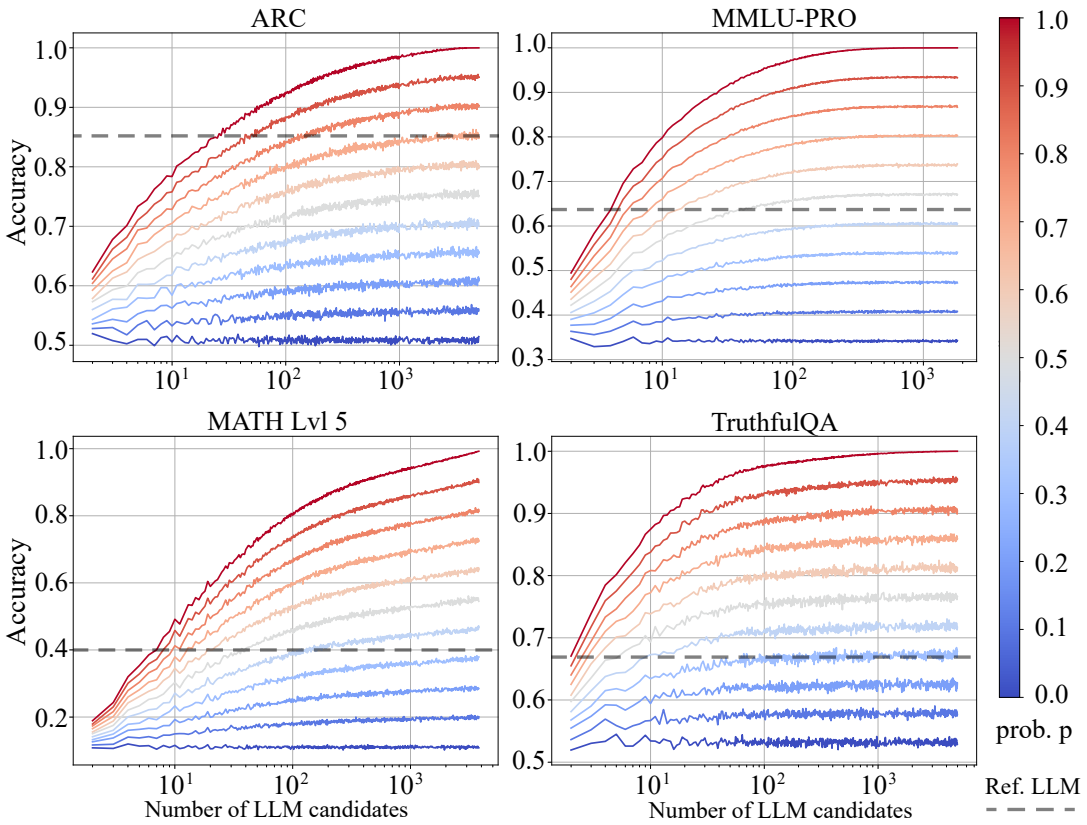

Figure 2: **The Model-level Scaling Up Phenomenon in Routing LLMs**. As shown in Section 3, the Prob. p indicates the performance of the router, with values closer to 1 representing greater similarity to the oracle router's capability. If $p \to 0$, then $r_o(p)$ degenerates into a random sampler. When the router $r_o(p)$ reaches a certain level of capability, it induces a scaling up phenomenon in the Routing LLMs paradigm. Specifically, as the number of LLM candidates increases, performance rapidly improves. "Ref. LLM" denotes a representative LLM with strong performance on given benchmark, such as GPT-4. Further details are provided in Section 5.1. For more examples of model-level scaling up phenomenon, please refer to the Appendix G.

equate performance records, etc. To address these challenges, in this paper, we collect and clean 12 popular LLM evaluations covering fields such as knowledge-based Q&A, commonsense inference, semantic understanding, etc. From these evaluations, we use performance records from over 8,500 LLMs, amounting to more than 200,000,000 entries, to construct a comprehensive Router benchmark named RouterEval in Section 4. Utilizing RouterEval, in Section 5, we conduct a comprehensive exploration of several existing router construction methods across various settings and find that these methods still have considerable room for improvement. We show the related works in Appendix C, and summarize the contributions of this paper as follows:

- We find the model-level scaling up phenomenon in LLMs, i.e., capable router in Routing LLMs can significantly enhance the performance as the number of candidates increases.
- Based on over 8,500 LLMs and their performance records exceeding 200 million entries across various LLM evaluations, we construct a comprehensive benchmark tailored for router design, named RouterEval, to re-examined a wide range of existing router methods.

2 Preliminary

Notation. Assume there is a LLM pool containing m LLMs $\{\ell_i\}_{i=1}^m$ and input set $\{s_j\}_{j=1}^n$. For a given input s_j, its representation $\kappa(s_j)$ can be obtained using an encoder κ. Based on the performance records of these LLMs on s_j, we can obtain a m-dimensional one-hot selection vector v_j, where the dimension marked as 1 indicates that the corresponding LLM indexed by that dimension achieves the dominant performance among $\{\ell_i\}_{i=1}^m$ on the input s, otherwise it is marked as 0. The vector v_j indicates the optimal choice for the router on the input s_j. Specifically, taking $m = 3$ as ex-

ample, if the performance metric is correctness or incorrectness, the dimensions of v_0 corresponding to the LLMs that provide correct answers can be simultaneously marked as 1, such as $v_0 = [1, 1, 0]$ or even $v_0 = [1, 1, 1]$, since these LLMs can all be considered as the optimal choices for the router. Moreover, If the metric is continuous, the index of the LLM whose performance is within 95% of the optimal performance can be marked as 1, otherwise, it is marked as 0.

Router. In practice, the task of the router r_θ with learnable parameters θ is to establish the following mapping using the given input set $\{s_j\}_{j=1}^n$ and the corresponding selection vectors $\{v_j\}_{j=1}^n$, and possible external data $\mathcal{D}$ as the training set:

$$r_\theta[\kappa(s_j)|\mathcal{D}] \rightarrow v_j. \tag{1}$$

If there is no external data, then $\mathcal{D} = \phi$. After training, for an unseen input s', the trained router is required to generate a selection vector v', and based on this vector, determine the optimal LLM from the LLM pool for the current input. Thus, the acquisition of the router can be regarded as a classic classification problem (Deng et al., 2009; Hu et al., 2018; Zhong et al., 2023b,a; Liang et al., 2020; Huang et al., 2020).

Given that the development of routers is still in its nascent stage, in this paper, we prioritize the performance corresponding to each benchmark when selecting LLMs, without considering factors such as computational cost, hallucination rate, etc. See Section 6 for more analysis.

3 Model-level Scaling Up Phenomenon

In this section, we comprehensively explore the relationship between the number of LLM candidates and the final performance under different router capabilities, to demonstrate the immense potential of Routing LLMs paradigm. Specifically, we consider the four well-known LLM benchmarks ARC (Clark et al., 2018), MMMU-PRO (Wang et al., 2024), MATH Lvl 5 (Hendrycks et al., 2021), TruthfulQA (Lin et al., 2021), and collect performance records of thousands of LLMs on these benchmarks. Using collected performance records, we can construct an oracle router r_o, which can select the optimal LLM to process a given input s from any m LLM candidates. Furthermore, we can construct other routers with different capabilities by defining $r_o(p)$ as

$$r_o(p) = \begin{cases} r_o, & \text{with probability } p, \\ \omega_m, & \text{with probability } 1-p, \end{cases} \tag{2}$$

where ω_m is a router that samples uniformly from the m candidate LLMs with probability $1/m$. As $p \rightarrow 1$, $r_o(p)$ approaches the oracle router r_o, resulting in strongest classification performance on m LLM candidates. Conversely, as $p \rightarrow 0$, $r_o(p)$ degenerates into a random sampler.

Subsequently, for a given m, we repeatedly sample m LLMs from the LLM pool with equal probability, 100 times, forming 100 candidate subsets each containing m LLM candidates. We then calculate the average performance of different routers $r_o(p)$ on these subsets. The statistical results are shown in Fig. 2, and we have following finding:

> (1) Model-level Scaling Up for LLMs.

As the number of candidates increases, most routers $r_o(p)$ can rapidly enhance performance, especially when $p \geq 0.5$. This implies that, given sufficiently well-constructed routers, Routing LLM is an effective paradigm for scaling up LLMs at the model-level. In fact, this phenomenon aligns with the classical neural scaling law (Kaplan et al., 2020), as the increase in the number of candidates can be regarded as an implicit increase in the number of parameters, which is manifested in a sparse manner under the Routing LLM paradigm.

> (2) Weak Candidates Can Also be Promising.

In fact, the vast majority of LLMs discussed in this paper are open-source and can be deployed locally (see Appendix B for details). The individual performance of these models is not particularly remarkable and generally falls significantly short of mainstream LLMs. However, as shown in Fig. 2, even relatively weak candidates can achieve complementary performance among multiple heterogeneous models under the Routing LLMs paradigm. They can obtain fine-grained division of labor for each input and outperform mainstream LLMs. This shows that relatively weak candidates can also be promising for obtaining high performance within this paradigm. See more details in Section 4.2.

> (3) Small Number of Candidates is Enough.

In Fig. 2, we not only observe that a sufficiently large number of candidates can achieve excellent performance, but also note that even with only 3 $\sim$

10 candidates, the performance can obtain strong performance, and even surpass that of the strong reference model. Therefore, when only moderately good performance is required, a small number of candidates is sufficient, which is highly advantageous for users with limited resources to adopt the Routing LLMs paradigm. Coupled with the aforementioned observation that weak candidates also hold great potential, the Routing LLMs paradigm demonstrates significant utility and applicability.

4 The Construction of RouterEval

In this section, we elaborate on the construction of our comprehensive benchmark for Routing LLMs, named **RouterEval**. We collected over 200,000,000 performance records for 12 popular LLM evaluations spanning areas such as knowledge-based Q&A, commonsense reasoning, semantic understanding, and instruction following, based on more than 8,500 LLMs. The benchmarks (Zellers et al., 2019; Clark et al., 2018; Wang et al., 2024; Hendrycks et al., 2020; Lin et al., 2021; Sakaguchi et al., 2021; Cobbe et al., 2021; Zhou et al., 2023; Suzgun et al., 2022; Rein et al., 2024; Sprague et al., 2023) involved include ARC, HellaSwag, MMLU, TruthfulQA, Winogrande, GSM8k, IFEval, BBH, GPQA, MUSR, MATH Lvl 5, and MMLU-PRO. See Appendix B, A and D for more details.

4.1 Data format

As shown in Eq. (1), our objective is to train a router r_θ with learnable parameters θ, which is essentially an m-way classifier. Given the representation $\kappa(s_j)$ of an input s, the router fits a selection vector $v_j \in \{0,1\}^m$ and identifies the optimal LLM for the input s, corresponding to the position of 1 in v_j. Accordingly, the input $\mathcal{X}$ and label $\mathcal{Y}$ format of RouterEval is

$$(\mathcal{X}, \mathcal{Y}) = \{\kappa(s_j), v_j\}_{j=1}^{n}, \tag{3}$$

where s_j encompasses all test samples from the 12 LLM benchmarks considered in this study. We also provide versions using four different pre-trained models (Beltagy et al., 2020; Reimers and Gurevych, 2019; Liu et al., 2019) as encoders κ: Sentence BERT, RoBERTa, the last layer of RoBERTa, and Longformer. The training, validation, and test sets are split in a ratio of 8:1:1. In all experiments, Roberta is used as an example. Researchers can select any embedding in our code or design their own embedding based on input.

4.2 The Construction of LLM Candidates

The selection vector $v_j \in \{0,1\}^m$ in Eq. (3) depends on the choice of m LLM candidates. In this paper, we set two difficulty levels: an easy level with $m \in \{3,5\}$ and a hard level with $m \in \{10, 100, 1000\}$. We focus primarily on the easy level, as the analysis in Fig. 2 and Section 3 shows that performance grows most rapidly when $2 \leq m \leq 10$. Moreover, the deployment cost of Routing LLMs is low in this range, making these values of m the most cost-effective and worthy of attention. The hard level of m is mainly used to explore the limits of router design, in preparation for the strong performance demonstrated by the scaling up phenomenon in Section 3.

For each given benchmark and m, we construct three types of LLMs candidate G, and the final model performance is the average of the results from these three candidates. Specifically, we sort all N LLMs with performance between 0.1 and 0.9 based on their individual performance on the given benchmark, obtaining $\{\ell'_i\}_{i=1}^{N}$. We then consider the following optimization problem regarding the performance of G and its corresponding oracle r_o, as shown in Eq. (4),

$$\hat{G} = \max_G \text{Perf.}(r_o, G). \tag{4}$$

When the m LLMs in G are all selected from $\{\ell'_i\}_{i=1}^{\lfloor 0.2N \rfloor}$ and $\{\ell'_i\}_{i=\lfloor 0.8N \rfloor}^{N}$ respectively, $\hat{G}$ forms the "all-strong" group and the "all-weak" group. And $\hat{G}$ forms the "strong-to-weak" group, when the j-th LLM in G is selected from $\{\ell'_i\}_{i=(j-1)m}^{\min(jm,N)}$. Through this process, we can explore the router from multiple perspectives and investigate the potential of Routing LLMs. For instance, in MMLU and $m = 10$, while the individual performance of each LLM in the "all-weak" group does not exceed 0.3, its oracle performance can reach 0.95. See Appendix D for more details. Additionally, through validation, we ensure that in all candidate groups, at least 10^{30} routers can achieve oracle performance, showing the learnability of the routers.

4.3 Extra Training Data

Note that, the direct training set in RouterEval typically ranges from several hundred to tens of thousands. This poses a significant challenge for training a reliable router, especially for the hard level m discussed in Section 4.2. Therefore, similar to what is shown in Eq. (1), we provide an extra dataset $\mathcal{D}$ to aid in the training of the router.

Specifically, we open-source over 200,000,000 performance records constructed from various LLMs and benchmarks involved in this study. Utilizing these data, researchers can explore various data augmentation techniques (Shorten and Khoshgoftaar, 2019; Qin et al., 2024), few-shot learning (Vinyals et al., 2016; Huang et al., 2024), regularization method (Srivastava et al., 2014; Huang et al., 2021, 2023), pre-training approaches (Zoph et al., 2020; Zhong et al., 2024b), and recommendation systems (Zhao et al., 2025; Zhong et al., 2024c), etc. Additionally, to further enhance the capabilities of the router, our proposed RouterEval encourages the use of any external data and pre-trained models.

5 Experiments

In this section, we present the performance of various existing routers on our proposed RouterEval. Specifically, in these experiments, we do not consider the extra data mentioned in Section 4.3, since the utilization of such data is highly diverse and can be considered from multiple perspectives.

5.1 Metrics

We establish three metrics to investigate the performance of different routers r_θ: (1) **Original metric** $\mu_o(r_\theta)$, i.e., the overall performance of the LLMs selected by router r_θ on the given benchmark. (2) **Reference value** V_R. For each benchmark, we select a representative LLM with strong performance as the reference, such as GPT-4, with its performance denoted as Perf.(ref.). Then,

$$V_R = \mu_o(r_\theta)/\text{Perf.(ref.)}. \quad (5)$$

See details of reference LLMs in Appendix A. (3) **Best single model value** V_B. In addition to the global metric V_G, we also need a local metric to explore the potential of the router. Specifically, given a set of candidate models, let the best performance of a single model in the candidate set be Perf.(BSM). Then,

$$V_B = \mu_o(r_\theta)/\text{Perf.(BSM)}. \quad (6)$$

(4) **Classification bias** E_p. As mentioned in Section 4.3, under the current settings, the classification of the router is prone to bias if an effective training method for the router is not proposed. Therefore, we use entropy to measure the diversity of the classifier's prediction distribution. If the router always select same LLM, the entropy of its prediction distribution will be low, indicating a lack of diversity. Specifically,

$$E_p = -\frac{1}{n}\sum_{j=1}^{n}\sum_{i=1}^{m} P_i^{(j)} \log P_i^{(j)}, \quad (7)$$

where $P_i^{(j)} \in [0,1]^m$ represents the probability that the router selects the LLM with index i for the j-th test sample.

5.2 Baselines

In this paper, we consider two kind of baselines for RouterEval. Specifically, these include: (1) **Strong router**. To directly evaluate the performance of routers, we introduce the oracle router r_o and $r_o(0.5)$ mentioned in Section 3 . (2) **Existing router**. We also compile some recently router methods, such as LinearR, MLPR, C-RoBERTa, MLC, and PRknn (Hu et al., 2024a; Srivatsa et al., 2024; Zhao et al., 2024a). See Appendix E or our code for detailed implementation.

5.3 Result

In this section, we present the results of the baselines shown in Section 5.2, evaluated using the metrics described in Section 5.1. The results under easy level settings, $m \in \{3, 5\}$, are shown in Table 1 and Table 2. See more results under hard level settings $m \in \{10, 100, 1000\}$ in Appendix F.

The experimental results show that most existing routers have some classification capability. However, the selected LLMs still lag significantly behind the best single models in most settings and the strong reference model in terms of performance, i.e., $V_R \leq 1$ and $V_B \leq 1$. Moreover, no single router consistently outperforms other router across different benchmarks. Additionally, some routers exhibit lower E_p values, where low entropy E_p suggests potential overfitting, resulting in biases when selecting LLMs. For a detailed analysis, please refer to Section 6.

6 Analysis

In this Section, We conduct a more comprehensive analysis of the proposed RouterEval.

> (1) The differences between Routing LLM and existing paradigms.

In fact, several existing paradigms in other fields are related to Routing LLMs. This section aims to analyze their relationships and differences. See related works in Appendix C for more details.

	Router	ARC				HellaSwag				MMLU				TruthfulQA			
		$\mu_o\uparrow$	$V_R\uparrow$	$V_B\uparrow$	E_p	$\mu_o\uparrow$	$V_R\uparrow$	$V_B\uparrow$	E_p	$\mu_o\uparrow$	$V_R\uparrow$	$V_B\uparrow$	E_p	$\mu_o\uparrow$	$V_R\uparrow$	$V_B\uparrow$	E_p
$m=3$	Oracle r_o	0.80	0.94	1.34	1.02	0.80	0.84	1.08	1.32	0.89	1.03	1.35	1.00	0.85	1.27	1.21	1.05
	$r_o(0.5)$	0.67	0.79	1.11	1.47	0.74	0.78	1.00	1.53	0.75	0.87	1.11	1.47	0.74	1.10	1.04	1.47
	LinearR	0.61	0.71	0.96	1.42	**0.75**	**0.79**	**1.00**	1.43	0.74	0.85	1.04	1.30	**0.72**	**1.08**	**1.00**	1.36
	MLPR	0.61	0.71	0.96	1.42	0.75	0.78	1.00	1.43	**0.74**	**0.86**	**1.04**	1.26	0.71	1.06	0.96	1.30
	C-RoBERTa	0.62	0.73	1.00	1.03	**0.75**	**0.79**	**1.00**	0.29	0.73	0.84	1.02	0.62	0.71	1.06	0.96	0.31
	MLC	**0.63**	**0.74**	**1.00**	0.81	0.75	0.78	1.00	1.01	0.73	0.85	1.02	0.79	0.70	1.05	0.95	0.49
	PRknn	0.60	0.71	0.97	1.56	0.72	0.76	0.97	1.57	0.70	0.81	0.98	1.55	0.70	1.04	0.95	1.55
	Random	0.54	0.64	0.89	1.59	0.68	0.71	0.91	1.59	0.62	0.71	0.88	1.59	0.62	0.93	0.86	1.59
$m=5$	Oracle r_o	0.85	1.00	1.34	1.57	0.81	0.85	1.10	2.00	0.92	1.07	1.63	1.49	0.89	1.33	1.27	1.72
	$r_o(0.5)$	0.70	0.82	1.09	2.16	0.74	0.78	1.00	2.25	0.75	0.87	1.24	2.14	0.75	1.12	1.05	2.19
	LinearR	0.64	0.75	0.93	2.15	0.75	0.79	1.00	2.19	0.69	0.80	1.01	2.04	**0.72**	**1.08**	**0.97**	2.15
	MLPR	0.64	0.75	0.93	2.13	**0.75**	**0.79**	**1.01**	2.20	**0.70**	**0.81**	**1.02**	2.00	0.71	1.05	0.93	2.11
	C-RoBERTa	**0.66**	**0.78**	**0.97**	0.82	0.75	0.79	1.00	0.52	0.68	0.79	0.98	1.02	0.70	1.04	0.92	0.84
	MLC	0.63	0.74	0.90	1.28	0.75	0.78	1.01	1.65	0.69	0.79	0.99	1.11	0.68	1.02	0.91	1.04
	PRknn	0.63	0.74	0.95	2.30	0.71	0.74	0.95	2.31	0.64	0.74	0.94	2.30	0.70	1.04	0.95	2.29
	Random	0.55	0.65	0.83	2.32	0.67	0.71	0.91	2.32	0.58	0.67	0.86	2.32	0.61	0.92	0.83	2.32
		Winogrande				GSM8k				IFEval				BBH			
$m=3$	Oracle r_o	0.95	1.09	1.22	1.20	0.87	0.95	1.29	1.10	0.79	1.02	1.33	1.04	0.82	0.99	1.42	0.97
	$r_o(0.5)$	0.86	0.98	1.09	1.51	0.76	0.82	1.10	1.49	0.67	0.87	1.08	1.47	0.68	0.82	1.15	1.46
	LinearR	0.76	0.87	0.95	1.45	**0.71**	**0.77**	**0.97**	1.37	0.70	0.91	1.08	1.10	0.63	0.76	1.04	1.34
	MLPR	**0.78**	**0.89**	**0.98**	1.30	0.69	0.75	0.95	1.33	0.70	0.91	1.08	0.94	**0.63**	**0.76**	**1.05**	1.30
	C-RoBERTa	**0.78**	**0.89**	**0.98**	0.60	0.69	0.75	0.94	0.61	**0.70**	**0.91**	**1.09**	0.79	0.60	0.72	0.98	0.80
	MLC	0.76	0.87	0.96	1.56	0.70	0.76	0.97	0.74	0.68	0.88	0.98	0.40	0.62	0.74	1.02	0.38
	PRknn	0.74	0.84	0.92	1.57	0.70	0.76	0.99	1.56	0.69	0.90	1.04	1.55	0.61	0.73	1.00	1.56
	Random	0.77	0.88	0.96	1.59	0.64	0.70	0.90	1.59	0.54	0.71	0.82	1.59	0.53	0.64	0.88	1.59
$m=5$	Oracle r_o	0.98	1.12	1.31	1.77	0.89	0.96	1.33	1.67	0.81	1.06	1.36	1.63	0.88	1.06	1.69	1.43
	$r_o(0.5)$	0.85	0.97	1.12	2.21	0.74	0.81	1.09	2.19	0.67	0.87	1.06	2.17	0.70	0.84	1.29	2.13
	LinearR	0.75	0.85	0.96	2.15	0.72	0.78	0.98	2.01	0.67	0.87	0.95	1.86	**0.63**	**0.75**	**1.08**	2.11
	MLPR	**0.80**	**0.91**	**1.03**	2.08	0.72	0.78	0.98	1.99	**0.67**	**0.87**	**0.96**	1.80	0.62	0.74	1.05	2.05
	C-RoBERTa	0.76	0.87	0.97	0.83	**0.72**	**0.78**	**0.99**	0.82	0.67	0.87	0.92	1.02	0.59	0.71	0.99	1.03
	MLC	0.74	0.84	0.93	2.21	0.71	0.78	0.96	1.11	0.53	0.69	0.75	0.57	0.60	0.72	1.00	0.41
	PRknn	0.72	0.83	0.93	2.30	0.71	0.77	1.00	2.30	0.62	0.80	0.91	2.29	0.58	0.70	1.00	2.29
	Random	0.72	0.82	0.93	2.32	0.60	0.65	0.85	2.32	0.53	0.68	0.76	2.32	0.52	0.62	0.89	2.32

Table 1: **The Results on RouterEval (part1)**.See Section 5.1 for detials of various metrics. Red area and blue area highlights indicate the "Strong router" and "Existing router" mentioned in Section 5.2, respectively. The best results in existing router methods are highlighted with underlines and on bold. The values in the table are rounded to two decimal places. The results on hard level settings are shown in Appendix F.

Recommender system. In fact, Routing LLMs are a specialized type of Recommender System (RS), where the input can be seen as the user in the RS, LLM candidates as the items to be recommended, and the performance record as the interaction history between items and users. Given the input, the router needs to recommend an appropriate LLM to achieve various objectives, such as high accuracy, low computational cost, or minimal hallucination. However, compared to traditional RS, Routing LLMs have very limited "user information" that can be collected, and labeling this data is challenging. Most of the data is private and rarely open-source. The over 200,000,000 performance records we have organized and open-sourced represent only a small step toward building a router with strong recommendation capabilities.

LLMs ensemble. For a given input, the LLMs ensemble paradigm focuses more on requiring all LLM candidates to perform inference, then aggregating and organizing their results, with the final output achieved through strategies such as majority voting. In contrast, the Routing LLMs paradigm performs an efficient assignment before the candidates' inference, making it a more computationally efficient approach. Of course, since different paradigms can borrow from each other, there is still some technical overlap between these two paradigms in many improvement methods.

LLMs fusion. The paradigm of LLM fusion, which combines different LLMs to achieve superior performance, is a promising technology. It typically requires the LLMs being fused to have the same structure. However, Routing LLMs can

	Router	GPQA				MUSR				MATH Lvl 5				MMLU-PRO			
		$\mu_o\uparrow$	$V_R\uparrow$	$V_B\uparrow$	E_p	$\mu_o\uparrow$	$V_R\uparrow$	$V_B\uparrow$	E_p	$\mu_o\uparrow$	$V_R\uparrow$	$V_B\uparrow$	E_p	$\mu_o\uparrow$	$V_R\uparrow$	$V_B\uparrow$	E_p
$m=3$	Oracle r_o	0.68	1.71	2.00	0.72	0.74	1.05	1.74	0.81	0.56	1.39	1.33	1.14	0.67	1.05	1.45	1.05
	$r_o(0.5)$	0.49	1.24	1.43	1.41	0.56	0.81	1.32	1.43	0.45	1.14	1.06	1.49	0.55	0.86	1.13	1.48
	LinearR	**0.39**	**0.99**	**1.14**	1.48	0.43	0.61	0.99	1.43	0.44	1.11	0.95	1.37	**0.54**	**0.85**	**1.01**	1.37
	MLPR	0.36	0.91	1.05	1.29	0.43	0.62	0.99	1.40	0.43	1.08	0.92	1.32	**0.54**	**0.85**	**1.01**	1.36
	C-RoBERTa	0.33	0.82	0.92	0.60	**0.44**	**0.63**	**1.01**	1.02	**0.45**	**1.13**	**0.99**	0.65	0.54	0.84	0.97	0.33
	MLC	0.30	0.75	0.93	0.00	0.39	0.56	0.91	0.00	0.42	1.06	0.88	0.53	0.53	0.84	0.95	0.49
	PRknn	0.32	0.81	0.93	1.56	0.40	0.57	0.92	1.56	0.40	1.01	0.88	1.56	0.50	0.79	0.92	1.56
	Random	0.30	0.76	0.87	1.59	0.39	0.56	0.90	1.59	0.35	0.88	0.79	1.59	0.43	0.67	0.82	1.59
$m=5$	Oracle r_o	0.87	2.18	2.52	0.84	0.82	1.17	1.77	1.31	0.61	1.51	1.52	1.60	0.72	1.14	1.69	1.50
	$r_o(0.5)$	0.58	1.47	1.69	1.99	0.61	0.87	1.30	2.10	0.46	1.16	1.13	2.16	0.56	0.88	1.23	2.14
	LinearR	**0.40**	**1.00**	**1.15**	2.19	**0.46**	**0.65**	**0.96**	2.24	0.43	1.08	0.97	2.12	0.54	0.85	1.01	2.06
	MLPR	0.36	0.92	1.05	2.04	0.45	0.65	0.95	2.24	**0.44**	**1.10**	**0.99**	2.06	**0.54**	**0.86**	**1.02**	2.05
	C-RoBERTa	0.33	0.84	0.95	0.82	0.46	0.65	0.93	1.54	0.42	1.05	0.91	0.65	0.53	0.84	0.97	0.33
	MLC	0.30	0.76	0.86	0.00	0.40	0.58	0.85	0.09	0.41	1.04	0.87	0.77	0.54	0.85	1.00	0.74
	PRknn	0.32	0.80	0.91	2.30	0.43	0.61	0.90	2.30	0.39	0.97	0.88	2.30	0.49	0.76	0.93	2.30
	Random	0.30	0.75	0.85	2.32	0.40	0.57	0.84	2.32	0.32	0.80	0.73	2.32	0.40	0.62	0.78	2.32

Table 2: **The Results on RouterEval (part2)**. See Section 5.1 for detials of various metrics. Red area and blue area highlights indicate the "Strong router" and "Existing router" mentioned in Section 5.2, respectively. The best results in existing router methods are highlighted with underlines and on bold. The values in the table are rounded to two decimal places. The results on hard level settings are shown in Appendix F.

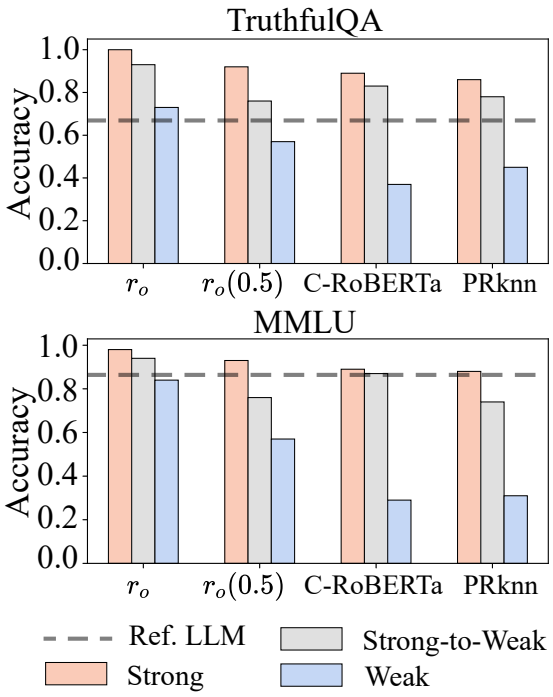

Figure 3: **The Results on Different Candidate Group**.

integrate not only homogenous models but also heterogeneous ones, offering greater flexibility from an application perspective.

Mixture-of-Experts (MoE). Traditional MoE mainly focuses on the mixture of local parameters, such as FFN, within a single LLM to achieve better performance like a larger model by activating only a subset of parameters. Similarly, Routing LLMs can be viewed as a larger-granularity MoE, specifically a Model-level "MoE", where the experts refer to the candidate LLMs. Both are effective methods for improving LLM performance.

In summary, although the Routing LLMs paradigm shares similarities with many existing approaches, it is important to note that Routing LLMs can actually be compatible with these paradigms. This compatibility allows the integration of heterogeneous models from these paradigms into the candidate pool, leading to further enhancements in performance.

> (2) How do different types of candidate combinations affect performance?

In Section 4.2, we constructed three types of candidate combinations, namely "all-strong," "all-weak," and "strong-to-weak" groups. In this part, we attempt to analyze how the selection of these different groups affects the performance of Routing LLMs. As shown in Fig. 3, taking $m = 5$ as example, we consider two types of strong routers, r_o and $r_o(0.5)$, as well as two existing routers, C-RoBERTa and PRknn.

We observe that, overall, the "all-strong" group typically exhibits the best performance, while the "all-weak" group lags behind. However, when the router has sufficiently strong classification capabilities, like r_o, even the "all-weak" group can achieve satisfactory performance. For instance, in MMLU result in Fig. 3, the performance of each individual LLM in the "all-weak" group does not exceed

0.3, but under the capable router r_o, it can approximatively approach the performance of reference model, i.e., GPT-4. This indicates that heterogeneous models can effectively complement each other under the guidance of Routing LLMs, provided that the router is sufficiently capable.

	Router	all-strong	all-weak	strong-to-weak
$m=3$	Oracle r_o	1.39	0.77	0.96
	$r_o(0.5)$	1.55	1.42	1.45
	LinearR	1.54	1.54	0.81
	MLPR	1.50	1.52	0.76
	C-RoBERTa	0.93	0.94	0.00
	MLC	1.52	0.34	0.52
	PRknn	1.58	1.56	1.52
	Random	1.59	1.59	1.59
$m=5$	Oracle r_o	2.09	0.90	1.49
	$r_o(0.5)$	2.27	2.00	2.15
	LinearR	2.27	2.28	1.58
	MLPR	2.26	2.25	1.50
	C-RoBERTa	1.53	1.53	0.00
	MLC	2.25	0.03	1.06
	PRknn	2.31	2.30	2.28
	Random	2.32	2.32	2.32

Table 3: **The E_p on Various Candidate Groups**. Some router methods suffer from the classification bias, i.e., E_p is close to 0.

(3) Classification bias in routers.

In Section 5.1, we proposed using E_p to analyze the classification bias of routers and found that existing router methods may exhibit classification bias, as shown in Tables 1 and 2. In this part, taking MMLU benchmark as an example, we conduct a more detailed analysis. As illustrated in Table 3, the bias is severe in several settings and router methods, where strong models are more likely to be selected with higher probability.

In fact, if $E_p \to 0$, the router degrades into an individual router, which is actually the best-performing router in the training set. While this strategy can still achieve decent performance, it fails to leverage the advantages of the Routing LLMs paradigm. From the results in Table 3, even in the "strong-to-weak" group, a strong router like r_o requires diverse selections to integrate the complementary strengths of different candidates for better performance. Thus, debiasing is crucial for further enhancing the router's capability.

(4) How to boost the performance of router?

As shown in Table 1 and 2, the current router has significant room for improvement, with potential directions including the following. First, fully leveraging the extra data of performance records in Section 4.3 may be crucial. For instance, data augmentation, pre-training methods, and few-shot learning techniques could be constructed based on this data. Additionally, as mentioned in Section 6 (1), if we regard Routing LLMs as a recommender system (RS), future research could focus on utilizing extra data to design effective representation learning for LLMs and inputs (Zhuang et al., 2024a), addressing the cold start (Schein et al., 2002; Lam et al., 2008; Wei et al., 2021) problem of the router, and employing causal inference techniques (Liang et al., 2016; Wang et al., 2020; Huang et al., 2025), which are classic methods in RS research, to improve the router, especially for debiasing. This aligns with the objectives outlined in Section 6 (3).

(5) Why not consider other routing objective, like computational cost, temporarily?

In fact, the paradigm of RouterEval can be extended to consider additional objectives (Chen et al., 2023; Wang et al., 2025; Šakota et al., 2024) such as computational cost and hallucination rate through multi-objective optimization approaches (Deb and Gupta, 2005; Giagkiozis and Fleming, 2015). However, as shown in the experiments in Section 5, even when focusing solely on performance metrics across different benchmarks, the current router methods still have significant room for improvement. Under these circumstances, it is advisable to temporarily defer the exploration of other objectives. Otherwise, with the current limited data availability, adding more learning targets may further compromise performance. Therefore, it is recommended to first concentrate on enhancing performance. Once a certain level of exploration has been achieved, more data can then be utilized to expand to other optimization objectives.

7 Conclusion

In this paper, we comprehensively explored the potential of the Routing LLMs paradigm through extensive experiments and identified the scaling-up phenomenon of LLMs at the model level. Furthermore, given that the development of routers is hindered by the lack of comprehensive benchmarks, we introduce RouterEval, a benchmark based on 200 million performance records across 12 LLM evaluations. The evaluations reveal that existing routing methods still have room for improvement.

Limitations

Although we observed that the Routing LLMs paradigm can trigger the model-level scaling-up phenomenon of LLMs in this paper, this implies that a large number of LLMs may cause deployment challenges. Fortunately, our experiments found that the cost-effectiveness of this paradigm is highest when there are approximately 3 to 10 LLM candidates, that is, the performance growth rate is the fastest with fewer LLMs. Therefore, if we do not pursue extremely outstanding performance, it is still possible to achieve lower computational requirements in deployment. Additionally, for industrial deployment, when there are sufficiently many inputs as a batch of requests, in fact, as long as the routing infrastructure is well-developed, the average computational cost of inputs is not high. Furthermore, despite considering a sufficient number of LLMs and their corresponding performance records, the current data volume still cannot produce an excellent router. Increasing the data volume will be one of the most important research directions in the future, as most benchmark performance record data are not open-source and really expensive, which requires the cooperation of the entire community.

Acknowledgments

This work was supported by National Science and Technology Major Project (No.2021ZD0111601), National Natural Science Foundation of China under Grants No. 623B2099, 62272494, Guangdong Basic and Applied Basic Research Foundation (No.2023A1515011374, 2023A1515012845), and Guangzhou Science and Technology Program (No.2024A04J6365).

References

Josh Achiam, Steven Adler, et al. 2023. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*.

Pranjal Aggarwal, Aman Madaan, Ankit Anand, Srividya Pranavi Potharaju, Swaroop Mishra, Pei Zhou, Aditya Gupta, Dheeraj Rajagopal, Karthik Kappaganthu, Yiming Yang, Shyam Upadhyay, Manaal Faruqui, and Mausam . 2024. Automix: Automatically mixing language models. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*.

Armen Aghajanyan, Lili Yu, Alexis Conneau, Wei-Ning Hsu, Karen Hambardzumyan, Susan Zhang, Stephen Roller, Naman Goyal, Omer Levy, and Luke Zettlemoyer. 2023. Scaling laws for generative mixed-modal language models. In *International Conference on Machine Learning*, pages 265–279. PMLR.

Samuel K Ainsworth, Jonathan Hayase, and Siddhartha Srinivasa. 2022. Git re-basin: Merging models modulo permutation symmetries. *arXiv preprint arXiv:2209.04836*.

Takuya Akiba, Makoto Shing, Yujin Tang, Qi Sun, and David Ha. 2024a. Evolutionary optimization of model merging recipes.

Takuya Akiba, Makoto Shing, Yujin Tang, Qi Sun, and David Ha. 2024b. Evolutionary optimization of model merging recipes. *CoRR*, abs/2403.13187.

Dario Amodei, Sundaram Ananthanarayanan, Rishita Anubhai, Jingliang Bai, Eric Battenberg, Carl Case, Jared Casper, Bryan Catanzaro, Qiang Cheng, Guoliang Chen, et al. 2016. Deep speech 2: End-to-end speech recognition in english and mandarin. In *International conference on machine learning*, pages 173–182. PMLR.

Newsha Ardalani, Carole-Jean Wu, Zeliang Chen, Bhargav Bhushanam, and Adnan Aziz. 2022. Understanding scaling laws for recommendation models. *arXiv preprint arXiv:2208.08489*.

Michele Banko and Eric Brill. 2001. Scaling to very very large corpora for natural language disambiguation. In *Proceedings of the 39th annual meeting of the Association for Computational Linguistics*, pages 26–33.

Yamini Bansal, Behrooz Ghorbani, Ankush Garg, Biao Zhang, Colin Cherry, Behnam Neyshabur, and Orhan Firat. 2022. Data scaling laws in nmt: The effect of noise and architecture. In *International Conference on Machine Learning*, pages 1466–1482. PMLR.

Iz Beltagy, Matthew E Peters, and Arman Cohan. 2020. Longformer: The long-document transformer. *arXiv preprint arXiv:2004.05150*.

Christopher M Bishop and Nasser M Nasrabadi. 2006. *Pattern recognition and machine learning*, volume 4. Springer.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. 2020. Language models are few-shot learners. *NeurIPS*.

Ruisi Cai, Saurav Muralidharan, Greg Heinrich, Hongxu Yin, Zhangyang Wang, Jan Kautz, and Pavlo Molchanov. 2024. Flextron: Many-in-one flexible large language model. In *International Conference on Machine Learning (ICML)*.

Chong Chen, Min Zhang, Yiqun Liu, and Shaoping Ma. 2018. Neural attentional rating regression with review-level explanations. In *Proceedings of the 2018 world wide web conference*, pages 1583–1592.

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

Shuhao Chen, Weisen Jiang, Baijiong Lin, James Kwok, and Yu Zhang. 2024a. Routerdc: Query-based router by dual contrastive learning for assembling large language models. *Advances in Neural Information Processing Systems*, 37:66305–66328.

Shuhao Chen, Weisen Jiang, Baijiong Lin, James Kwok, and Yu Zhang. 2024b. RouterDC: Query-based router by dual contrastive learning for assembling large language models. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*.

Mehdi Cherti, Romain Beaumont, Ross Wightman, Mitchell Wortsman, Gabriel Ilharco, Cade Gordon, Christoph Schuhmann, Ludwig Schmidt, and Jenia Jitsev. 2023. Reproducible scaling laws for contrastive language-image learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2818–2829.

Aidan Clark, Diego de Las Casas, Aurelia Guy, Arthur Mensch, Michela Paganini, Jordan Hoffmann, Bogdan Damoc, Blake Hechtman, Trevor Cai, Sebastian Borgeaud, et al. 2022. Unified scaling laws for routed language models. In *International conference on machine learning*, pages 4057–4086. PMLR.

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

Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, et al. 2021. Training verifiers to solve math word problems. *arXiv preprint arXiv:2110.14168*.

Nico Daheim, Thomas Möllenhoff, Edoardo Maria Ponti, Iryna Gurevych, and Mohammad Emtiyaz Khan. 2023. Model merging by uncertainty-based gradient matching. *arXiv preprint arXiv:2310.12808*.

Christoph Dann, Yishay Mansour, Teodor Vanislavov Marinov, and Mehryar Mohri. 2024. Domain adaptation for robust model routing. In *Adaptive Foundation Models: Evolving AI for Personalized and Efficient Learning*.

Kalyanmoy Deb and Himanshu Gupta. 2005. Searching for robust pareto-optimal solutions in multi-objective optimization. In *International conference on evolutionary multi-criterion optimization*, pages 150–164. Springer.

Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. 2009. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pages 248–255. Ieee.

Tyler Derr, Yao Ma, Wenqi Fan, Xiaorui Liu, Charu Aggarwal, and Jiliang Tang. 2020. Epidemic graph convolutional network. In *Proceedings of the 13th International Conference on Web Search and Data Mining (WSDM)*, pages 160–168.

Fnu Devvrit, Sneha Kudugunta, Aditya Kusupati, Tim Dettmers, Kaifeng Chen, Inderjit S Dhillon, Yulia Tsvetkov, Hannaneh Hajishirzi, Sham M. Kakade, Ali Farhadi, and Prateek Jain. 2024. MatFormer: Nested transformer for elastic inference. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*.

Dujian Ding, Ankur Mallick, Chi Wang, Robert Sim, Subhabrata Mukherjee, Victor Rühle, Laks V. S. Lakshmanan, and Ahmed Hassan Awadallah. 2024a. Hybrid LLM: Cost-efficient and quality-aware query routing. In *The Twelfth International Conference on Learning Representations*.

Ning Ding, Yulin Chen, Ganqu Cui, Xingtai Lv, Weilin Zhao, Ruobing Xie, Bowen Zhou, Zhiyuan Liu, and Maosong Sun. 2024b. Mastering text, code and math simultaneously via fusing highly specialized language models.

Swati Dongre and Jitendra Agrawal. 2023. Deep learning-based drug recommendation and adr detection healthcare model on social media. *IEEE Transactions on Computational Social Systems*.

Arthur Douillard, Qixuan Feng, Andrei A. Rusu, Adhiguna Kuncoro, Yani Donchev, Rachita Chhaparia, Ionel Gog, Marc'Aurelio Ranzato, Jiajun Shen, and Arthur Szlam. 2024. DiPaCo: Distributed path composition. *Preprint*, arXiv:2403.10616.

Wenqi Fan, Tyler Derr, Yao Ma, Jianping Wang, Jiliang Tang, and Qing Li. 2019a. Deep adversarial social recommendation. In *28th International Joint Conference on Artificial Intelligence (IJCAI-19)*, pages 1351–1357. International Joint Conferences on Artificial Intelligence.

Wenqi Fan, Qing Li, and Min Cheng. 2018. Deep modeling of social relations for recommendation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 32.

Wenqi Fan, Xiaorui Liu, Wei Jin, Xiangyu Zhao, Jiliang Tang, and Qing Li. 2022a. Graph trend filtering networks for recommendation. In *Proceedings of the 45th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 112–121.

Wenqi Fan, Yao Ma, Qing Li, Jianping Wang, Guoyong Cai, Jiliang Tang, and Dawei Yin. 2020. A graph neural network framework for social recommendations. *IEEE Transactions on Knowledge and Data Engineering*.

Wenqi Fan, Yao Ma, Dawei Yin, Jianping Wang, Jiliang Tang, and Qing Li. 2019b. Deep social collaborative filtering. In *Proceedings of the 13th ACM Conference on Recommender Systems*, pages 305–313.

Wenqi Fan, Xiangyu Zhao, Xiao Chen, Jingran Su, Jingtong Gao, Lin Wang, Qidong Liu, Yiqi Wang, Han Xu, Lei Chen, et al. 2022b. A comprehensive survey on trustworthy recommender systems. *arXiv preprint arXiv:2209.10117*.

William Fedus, Barret Zoph, and Noam Shazeer. 2022a. Switch transformers: Scaling to trillion parameter models with simple and efficient sparsity. *Journal of Machine Learning Research*, 23(120):1–39.

William Fedus, Barret Zoph, and Noam Shazeer. 2022b. Switch Transformers: Scaling to trillion parameter models with simple and efficient sparsity. *Preprint*, arXiv:2101.03961.

Tao Feng, Yanzhen Shen, and Jiaxuan You. 2024. GraphRouter: A graph-based router for llm selections. *Preprint*, arXiv:2410.03834.

Elias Frantar, Carlos Riquelme, Neil Houlsby, Dan Alistarh, and Utku Evci. 2023. Scaling laws for sparsely-connected foundation models. *arXiv preprint arXiv:2309.08520*.

Evan Frick, Connor Chen, Joseph Tennyson, Tianle Li, Wei-Lin Chiang, Anastasios N Angelopoulos, and Ion Stoica. 2025. Prompt-to-leaderboard. *arXiv preprint arXiv:2502.14855*.

Leo Gao, John Schulman, and Jacob Hilton. 2023. Scaling laws for reward model overoptimization. In *International Conference on Machine Learning*, pages 10835–10866. PMLR.

Ioannis Giagkiozis and Peter J Fleming. 2015. Methods for multi-objective optimization: An analysis. *Information Sciences*, 293:338–350.

Charles Goddard, Shamane Siriwardhana, Malikeh Ehghaghi, Luke Meyers, Vladimir Karpukhin, Brian Benedict, Mark McQuade, and Jacob Solawetz. 2024. Arcee's mergekit: A toolkit for merging large language models. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing: Industry Track*, pages 477–485.

Albert Gu and Tri Dao. 2023. Mamba: Linear-time sequence modeling with selective state spaces. *arXiv preprint arXiv:2312.00752*.

Hasan Abed Al Kader Hammoud, Umberto Michieli, Fabio Pizzati, Philip Torr, Adel Bibi, Bernard Ghanem, and Mete Ozay. 2024. Model merging and safety alignment: One bad model spoils the bunch. *arXiv preprint arXiv:2406.14563*.

Surya Narayanan Hari and Matt Thomson. 2023a. Tryage: Real-time, intelligent routing of user prompts to large language models. *Preprint*, arXiv:2308.11601.

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

Tatsunori Hashimoto. 2021. Model performance scaling with multiple data sources. In *International Conference on Machine Learning*, pages 4107–4116. PMLR.

Trevor Hastie, Robert Tibshirani, Jerome H Friedman, and Jerome H Friedman. 2009. *The elements of statistical learning: data mining, inference, and prediction*, volume 2. Springer.

Xiangnan He, Lizi Liao, Hanwang Zhang, Liqiang Nie, Xia Hu, and Tat-Seng Chua. 2017. Neural collaborative filtering. In *Proceedings of the 26th international conference on world wide web*, pages 173–182.

Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. 2020. Measuring massive multitask language understanding. *arXiv preprint arXiv:2009.03300*.

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

Amr Hendy, Mohamed Abdelrehim, Amr Sharaf, Vikas Raunak, Mohamed Gabr, Hitokazu Matsushita, Young Jin Kim, Mohamed Afify, and Hany Hassan Awadalla. 2023. How good are GPT models at machine translation? a comprehensive evaluation. *Preprint*, arXiv:2302.09210.

Tom Henighan, Jared Kaplan, Mor Katz, Mark Chen, Christopher Hesse, Jacob Jackson, Heewoo Jun, Tom B Brown, Prafulla Dhariwal, Scott Gray, et al. 2020. Scaling laws for autoregressive generative modeling. *arXiv preprint arXiv:2010.14701*.

Danny Hernandez, Jared Kaplan, Tom Henighan, and Sam McCandlish. 2021. Scaling laws for transfer. *arXiv preprint arXiv:2102.01293*.

Joel Hestness, Sharan Narang, Newsha Ardalani, Gregory Diamos, Heewoo Jun, Hassan Kianinejad, Md Mostofa Ali Patwary, Yang Yang, and Yanqi Zhou. 2017. Deep learning scaling is predictable, empirically. *arXiv preprint arXiv:1712.00409*.

Jacob Hilton, Jie Tang, and John Schulman. 2023. Scaling laws for single-agent reinforcement learning. *arXiv preprint arXiv:2301.13442*.

Jordan Hoffmann, Sebastian Borgeaud, Arthur Mensch, Elena Buchatskaya, Trevor Cai, Eliza Rutherford, Diego de Las Casas, Lisa Anne Hendricks, Johannes Welbl, Aidan Clark, et al. 2022. Training compute-optimal large language models. *arXiv preprint arXiv:2203.15556*.

Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, Weizhu Chen, et al. 2022. Lora: Low-rank adaptation of large language models. *ICLR*, 1(2):3.

Jie Hu, Li Shen, and Gang Sun. 2018. Squeeze-and-excitation networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 7132–7141.

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

Qitian Jason Hu, Jacob Bieker, Xiuyu Li, Nan Jiang, Benjamin Keigwin, Gaurav Ranganath, Kurt Keutzer, and Shriyash Kaustubh Upadhyay. 2024b. RouterBench: A benchmark for multi-LLM routing system. In *Agentic Markets Workshop at ICML 2024*.

Shengding Hu, Yuge Tu, Xu Han, Chaoqun He, Ganqu Cui, Xiang Long, Zhi Zheng, Yewei Fang, Yuxiang Huang, Weilin Zhao, et al. 2024c. Minicpm: Unveiling the potential of small language models with scalable training strategies. *arXiv preprint arXiv:2404.06395*.

Zhongzhan Huang, Mingfu Liang, Senwei Liang, and Wei He. 2021. Altersgd: Finding flat minima for continual learning by alternative training. *arXiv preprint arXiv:2107.05804*.

Zhongzhan Huang, Mingfu Liang, Shanshan Zhong, and Liang Lin. 2024. Attns: Attention-inspired numerical solving for limited data scenarios. In *International Conference on Machine Learning*, pages 19929–19948. PMLR.

Zhongzhan Huang, Senwei Liang, Mingfu Liang, and Haizhao Yang. 2020. Dianet: Dense-and-implicit attention network. In *Proceedings of the AAAI conference on artificial intelligence*, volume 34, pages 4206–4214.

Zhongzhan Huang, Shanshan Zhong, Pan Zhou, Shanghua Gao, Marinka Zitnik, and Liang Lin. 2025. A causality-aware paradigm for evaluating creativity of multimodal large language models. *IEEE Transactions on Pattern Analysis and Machine Intelligence*.

Zhongzhan Huang, Pan Zhou, Shuicheng Yan, and Liang Lin. 2023. Scalelong: Towards more stable training of diffusion model via scaling network long skip connection. *Advances in Neural Information Processing Systems*, 36:70376–70401.

Gabriel Ilharco, Marco Tulio Ribeiro, Mitchell Wortsman, Suchin Gururangan, Ludwig Schmidt, Hannaneh Hajishirzi, and Ali Farhadi. 2023a. Editing models with task arithmetic.

Gabriel Ilharco, Marco Túlio Ribeiro, Mitchell Wortsman, Ludwig Schmidt, Hannaneh Hajishirzi, and Ali Farhadi. 2023b. Editing models with task arithmetic. In *The Eleventh International Conference on Learning Representations, ICLR 2023, Kigali, Rwanda, May 1-5, 2023*. OpenReview.net.

Robert A. Jacobs, Michael I. Jordan, Steven J. Nowlan, and Geoffrey E. Hinton. 1991. Adaptive mixtures of local experts. *Neural Computation*, 3(1):79–87.

Dong-Hwan Jang, Sangdoo Yun, and Dongyoon Han. 2024. Model stock: All we need is just a few fine-tuned models.

Joel Jang, Seungone Kim, Seonghyeon Ye, Doyoung Kim, Lajanugen Logeswaran, Moontae Lee, Kyungjae Lee, and Minjoon Seo. 2023. Exploring the benefits of training expert language models over instruction tuning. In *Proceedings of the 40th International Conference on Machine Learning*, ICML'23. JMLR.org.

Albert Q Jiang, Alexandre Sablayrolles, Antoine Roux, Arthur Mensch, Blanche Savary, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Emma Bou Hanna, Florian Bressand, et al. 2024. Mixtral of experts. *arXiv preprint arXiv:2401.04088*.

Dongfu Jiang, Xiang Ren, and Bill Yuchen Lin. 2023a. Llm-blender: Ensembling large language models with pairwise ranking and generative fusion. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 14165–14178, Toronto, Canada. Association for Computational Linguistics.

Dongfu Jiang, Xiang Ren, and Bill Yuchen Lin. 2023b. LLM-Blender: Ensembling large language models with pairwise ranking and generative fusion. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 14165–14178, Toronto, Canada. Association for Computational Linguistics.

Xisen Jin, Xiang Ren, Daniel Preotiuc-Pietro, and Pengxiang Cheng. 2023. Dataless knowledge fusion by merging weights of language models.

Wittawat Jitkrittum, Harikrishna Narasimhan, Ankit Singh Rawat, Jeevesh Juneja, Zifeng Wang, Chen-Yu Lee, Pradeep Shenoy, Rina Panigrahy, Aditya Krishna Menon, and Sanjiv Kumar. 2025. Universal model routing for efficient llm inference. *arXiv preprint arXiv:2502.08773*.

M.I. Jordan and R.A. Jacobs. 1993. Hierarchical mixtures of experts and the EM algorithm. In *Proceedings of 1993 International Conference on Neural Networks (IJCNN-93-Nagoya, Japan)*, volume 2, pages 1339–1344 vol.2.

Jared Kaplan, Sam McCandlish, Tom Henighan, Tom B. Brown, Benjamin Chess, Rewon Child, Scott Gray, Alec Radford, Jeffrey Wu, and Dario Amodei. 2020. Scaling laws for neural language models.

Seungone Kim, Juyoung Suk, Shayne Longpre, Bill Yuchen Lin, Jamin Shin, Sean Welleck, Graham Neubig, Moontae Lee, Kyungjae Lee, and Minjoon Seo. 2024. Prometheus 2: An open source language model specialized in evaluating other language models. *arXiv preprint arXiv:2405.01535*.

Xuan Nhat Lam, Thuc Vu, Trong Duc Le, and Anh Duc Duong. 2008. Addressing cold-start problem in recommendation systems. In *Proceedings of the 2nd international conference on Ubiquitous information management and communication*, pages 208–211.

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

Chia-Hsuan Lee, Hao Cheng, and Mari Ostendorf. 2024. OrchestraLLM: Efficient orchestration of language models for dialogue state tracking. In *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pages 1434–1445, Mexico City, Mexico. Association for Computational Linguistics.

Young-Suk Lee, Md Arafat Sultan, Yousef El-Kurdi, Tahira Naseem Asim Munawar, Radu Florian, Salim Roukos, and Ramón Fernandez Astudillo. 2023b. Ensemble-instruct: Generating instruction-tuning data with a heterogeneous mixture of lms. *arXiv preprint arXiv:2310.13961*.

Margaret Li, Sneha Kudugunta, and Luke Zettlemoyer. 2025. (mis) fitting: A survey of scaling laws. *arXiv preprint arXiv:2502.18969*.

Yang Li. 2025. LLM Bandit: Cost-efficient llm generation via preference-conditioned dynamic routing. *Preprint*, arXiv:2502.02743.

Dawen Liang, Laurent Charlin, and David M Blei. 2016. Causal inference for recommendation. In *Causation: Foundation to Application, Workshop at UAI. AUAI*, volume 6, page 108.

Senwei Liang, Zhongzhan Huang, Mingfu Liang, and Haizhao Yang. 2020. Instance enhancement batch normalization: An adaptive regulator of batch noise. In *Proceedings of the AAAI conference on artificial intelligence*, volume 34, pages 4819–4827.

Stephanie Lin, Jacob Hilton, and Owain Evans. 2021. Truthfulqa: Measuring how models mimic human falsehoods. *arXiv preprint arXiv:2109.07958*.

Junling Liu, Chao Liu, Renjie Lv, Kang Zhou, and Yan Zhang. 2023. Is chatgpt a good recommender? a preliminary study. *arXiv preprint arXiv:2304.10149*.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*.

Yixin Liu and Pengfei Liu. 2021. Simcls: A simple framework for contrastive learning of abstractive summarization. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, pages 1065–1072.

Yueyue Liu, Hongyu Zhang, Yuantian Miao, Van-Hoang Le, and Zhiqiang Li. 2024. Optllm: Optimal assignment of queries to large language models. In *2024 IEEE International Conference on Web Services (ICWS)*, pages 788–798. IEEE.

Jinliang Lu, Ziliang Pang, Min Xiao, Yaochen Zhu, Rui Xia, and Jiajun Zhang. 2024a. Merge, ensemble, and cooperate! a survey on collaborative strategies in the era of large language models. *arXiv preprint arXiv:2407.06089*.

Keming Lu, Hongyi Yuan, Runji Lin, Junyang Lin, Zheng Yuan, Chang Zhou, and Jingren Zhou. 2023a. Routing to the expert: Efficient reward-guided ensemble of large language models. *arXiv preprint arXiv:2311.08692*.

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

Keming Lu, Hongyi Yuan, Runji Lin, Junyang Lin, Zheng Yuan, Chang Zhou, and Jingren Zhou. 2024b. Routing to the expert: Efficient reward-guided ensemble of large language models. In *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pages 1964–1974, Mexico City, Mexico. Association for Computational Linguistics.

Yao Ma and Jiliang Tang. 2021. *Deep learning on graphs*. Cambridge University Press.

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

Michael Matena and Colin Raffel. 2022. Merging models with fisher-weighted averaging.

Alireza Mohammadshahi, Arshad Rafiq Shaikh, and Majid Yazdani. 2024. Routoo: Learning to route to large language models effectively. *arXiv preprint arXiv:2401.13979*.

Yannis Montreuil, Axel Carlier, Lai Xing Ng, and Wei Tsang Ooi. 2025. Adversarial robustness in two-stage learning-to-defer: Algorithms and guarantees. *Preprint*, arXiv:2502.01027.

Vaishnavh Nagarajan and J. Zico Kolter. 2021. Uniform convergence may be unable to explain generalization in deep learning.

Ganesh Nathan et al. 2024. Fisher mask nodes for language model merging. *arXiv preprint arXiv:2403.09891*.

Quang H Nguyen, Duy C Hoang, Juliette Decugis, Saurav Manchanda, Nitesh V Chawla, and Khoa D Doan. 2024. Metallm: A high-performant and cost-efficient dynamic framework for wrapping llms. *arXiv preprint arXiv:2407.10834*.

Isaac Ong, Amjad Almahairi, Vincent Wu, Wei-Lin Chiang, Tianhao Wu, Joseph E. Gonzalez, M Waleed Kadous, and Ion Stoica. 2024. RouteLLM: Learning to route LLMs with preference data. *Preprint*, arXiv:2406.18665.

Michael Poli, Armin W Thomas, Eric Nguyen, Pragaash Ponnusamy, Björn Deiseroth, Kristian Kersting, Taiji Suzuki, Brian Hie, Stefano Ermon, Christopher Ré, et al. 2024. Mechanistic design and scaling of hybrid architectures. *arXiv preprint arXiv:2403.17844*.

Jinghui Qin, Zhongzhan Huang, Ying Zeng, Quanshi Zhang, and Liang Lin. 2024. An introspective data augmentation method for training math word problem solvers. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*.

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

Guillem Ramírez, Alexandra Birch, and Ivan Titov. 2024. Optimising calls to large language models with uncertainty-based two-tier selection. *arXiv preprint arXiv:2405.02134*.

Guillem Ramírez, Matthias Lindemann, Alexandra Birch, and Ivan Titov. 2023. Cache & distil: Optimising api calls to large language models. *arXiv preprint arXiv:2310.13561*.

Sebastian Raschka. 2018. Model evaluation, model selection, and algorithm selection in machine learning. *arXiv preprint arXiv:1811.12808*.

Mathieu Ravaut, Shafiq Joty, and Nancy Chen. 2022a. Summareranker: A multi-task mixture-of-experts reranking framework for abstractive summarization. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 4504–4524.

Mathieu Ravaut, Shafiq Joty, and Nancy F Chen. 2022b. Summareranker: A multi-task mixture-of-experts reranking framework for abstractive summarization. *arXiv preprint arXiv:2203.06569*.

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

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

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

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

Marija Šakota, Maxime Peyrard, and Robert West. 2024. 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.

Andrew I Schein, Alexandrin Popescul, Lyle H Ungar, and David M Pennock. 2002. Methods and metrics for cold-start recommendations. In *Proceedings of the 25th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 253–260.

Thibault Sellam, Dipanjan Das, and Ankur Parikh. 2020. BLEURT: Learning robust metrics for text generation. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7881–7892.

Avital Shafran, Roei Schuster, Thomas Ristenpart, and Vitaly Shmatikov. 2025. Rerouting LLM routers. *Preprint*, arXiv:2501.01818.

Noam Shazeer, Azalia Mirhoseini, Krzysztof Maziarz, Andy Davis, Quoc Le, Geoffrey Hinton, and Jeff Dean. 2017. Outrageously large neural networks: The sparsely-gated mixture-of-experts layer.

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

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

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

Ken Shoemake. 1985. Animating rotation with quaternion curves. In *Proceedings of the 12th Annual Conference on Computer Graphics and Interactive Techniques*, SIGGRAPH '85, page 245–254, New York, NY, USA. Association for Computing Machinery.

Connor Shorten and Taghi M Khoshgoftaar. 2019. A survey on image data augmentation for deep learning. *Journal of big data*, 6(1):1–48.

Sidak Pal Singh and Martin Jaggi. 2020. Model fusion via optimal transport. In *Advances in Neural Information Processing Systems*, volume 33, pages 22045–22055. Curran Associates, Inc.

Ben Sorscher, Robert Geirhos, Shashank Shekhar, Surya Ganguli, and Ari Morcos. 2022. Beyond neural scaling laws: beating power law scaling via data pruning. *Advances in Neural Information Processing Systems*, 35:19523–19536.

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

Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. 2014. Dropout: a simple way to prevent neural networks from overfitting. *Journal of Machine Learning Research*, 15(1):1929–1958.

KV Srivatsa, Kaushal Kumar Maurya, and Ekaterina Kochmar. 2024. Harnessing the power of multiple minds: Lessons learned from llm routing. *arXiv preprint arXiv:2405.00467*.

George Stoica, Daniel Bolya, Jakob Bjorner, Pratik Ramesh, Taylor Hearn, and Judy Hoffman. 2024. Zipit! merging models from different tasks without training. *Preprint*, arXiv:2305.03053.

Dimitris Stripelis, Zijian Hu, Jipeng Zhang, Zhaozhuo Xu, Alay Dilipbhai Shah, Han Jin, Yuhang Yao, Salman Avestimehr, and Chaoyang He. 2024. TensorOpera router: A multi-model router for efficient LLM inference. *Preprint*, arXiv:2408.12320.

Sainbayar Sukhbaatar, Olga Golovneva, Vasu Sharma, Hu Xu, Xi Victoria Lin, Baptiste Rozière, Jacob Kahn, Daniel Li, Wen tau Yih, Jason Weston, and Xian Li. 2024. Branch-train-mix: Mixing expert llms into a mixture-of-experts llm.

Fei Sun, Jun Liu, Jian Wu, Changhua Pei, Xiao Lin, Wenwu Ou, and Peng Jiang. 2019. Bert4rec: Sequential recommendation with bidirectional encoder representations from transformer. In *Proceedings of the 28th ACM international conference on information and knowledge management*, pages 1441–1450.

Mirac Suzgun, Nathan Scales, Nathanael Schärli, Sebastian Gehrmann, Yi Tay, Hyung Won Chung, Aakanksha Chowdhery, Quoc V Le, Ed H Chi, Denny Zhou, et al. 2022. Challenging big-bench tasks and whether chain-of-thought can solve them. *arXiv preprint arXiv:2210.09261*.

Yi Tay, Mostafa Dehghani, Samira Abnar, Hyung Won Chung, William Fedus, Jinfeng Rao, Sharan Narang, Vinh Q Tran, Dani Yogatama, and Donald Metzler. 2022. Scaling laws vs model architectures: How does inductive bias influence scaling? *arXiv preprint arXiv:2207.10551*.

Flavian Vasile, Elena Smirnova, and Alexis Conneau. 2016. Meta-prod2vec: Product embeddings using side-information for recommendation. In *Proceedings of the 10th ACM conference on recommender systems*, pages 225–232.

A Vaswani. 2017. Attention is all you need. *Advances in Neural Information Processing Systems*.

Neha Verma and Maha Elbayad. 2024. Merging text transformer models from different initializations. *arXiv preprint arXiv:2403.00986*.

Oriol Vinyals, Charles Blundell, Tim Lillicrap, Koray kavukcuoglu, and Daan Wierstra. 2016. Matching networks for one shot learning. *Advances in neural information processing systems*, pages 3630–3638.

Marija Šakota, Maxime Peyrard, and Robert West. 2024. 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*, WSDM '24, page 606–615, New York, NY, USA. Association for Computing Machinery.

Fanqi Wan, Xinting Huang, Deng Cai, Xiaojun Quan, Wei Bi, and Shuming Shi. 2024. Knowledge fusion of large language models. In *The Twelfth International Conference on Learning Representations*.

Hongyi Wang, Felipe Maia Polo, Yuekai Sun, Souvik Kundu, Eric Xing, and Mikhail Yurochkin. 2023a. Fusing models with complementary expertise.

Xinyuan Wang, Yanchi Liu, Wei Cheng, Xujiang Zhao, Zhengzhang Chen, Wenchao Yu, Yanjie Fu, and Haifeng Chen. 2025. Mixllm: Dynamic routing in mixed large language models. *arXiv preprint arXiv:2502.18482*.

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

Yixin Wang, Dawen Liang, Laurent Charlin, and David M Blei. 2020. Causal inference for recommender systems. In *Proceedings of the 14th ACM Conference on Recommender Systems*, pages 426–431.

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

Jason Wei, Yi Tay, Rishi Bommasani, Colin Raffel, Barret Zoph, Sebastian Borgeaud, Dani Yogatama, Maarten Bosma, Denny Zhou, Donald Metzler, et al. 2022. Emergent abilities of large language models. *arXiv preprint arXiv:2206.07682*.

Yinwei Wei, Xiang Wang, Qi Li, Liqiang Nie, Yan Li, Xuanping Li, and Tat-Seng Chua. 2021. Contrastive learning for cold-start recommendation. In *Proceedings of the 29th ACM international conference on multimedia*, pages 5382–5390.

Mitchell Wortsman, Gabriel Ilharco, Samir Ya Gadre, Rebecca Roelofs, Raphael Gontijo-Lopes, Ari S Morcos, Hongseok Namkoong, Ali Farhadi, Yair Carmon, Simon Kornblith, et al. 2022. Model soups: averaging weights of multiple fine-tuned models improves accuracy without increasing inference time. In *International conference on machine learning*, pages 23965–23998. PMLR.

Chuhan Wu, Fangzhao Wu, Yongfeng Huang, and Xing Xie. 2023. Personalized news recommendation: Methods and challenges. *ACM Transactions on Information Systems*, 41(1):1–50.

Fangzhao Wu, Ying Qiao, Jiun-Hung Chen, Chuhan Wu, Tao Qi, Jianxun Lian, Danyang Liu, Xing Xie, Jianfeng Gao, Winnie Wu, et al. 2020. Mind: A large-scale dataset for news recommendation. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 3597–3606.

Jiahao Wu, Wenqi Fan, Jingfan Chen, Shengcai Liu, Qing Li, and Ke Tang. 2022. Disentangled contrastive learning for social recommendation. In *Proceedings of the 31st ACM International Conference on Information & Knowledge Management*, pages 4570–4574.

Prateek Yadav, Derek Tam, Leshem Choshen, Colin Raffel, and Mohit Bansal. 2023a. Ties-merging: Resolving interference when merging models.

Prateek Yadav, Derek Tam, Leshem Choshen, Colin A. Raffel, and Mohit Bansal. 2023b. Ties-merging: Resolving interference when merging models. In *Advances in Neural Information Processing Systems 36: Annual Conference on Neural Information Processing Systems 2023, NeurIPS 2023, New Orleans, LA, USA, December 10 - 16, 2023*.

Enneng Yang, Li Shen, Guibing Guo, Xingwei Wang, Xiaochun Cao, Jie Zhang, and Dacheng Tao. 2024. Model merging in llms, mllms, and beyond: Methods, theories, applications and opportunities. *arXiv preprint arXiv:2408.07666*.

Enneng Yang, Zhenyi Wang, Li Shen, Shiwei Liu, Guibing Guo, Xingwei Wang, and Dacheng Tao. 2023. Adamerging: Adaptive model merging for multi-task learning. *CoRR*, abs/2310.02575.

Rex Ying, Ruining He, Kaifeng Chen, Pong Eksombatchai, William L Hamilton, and Jure Leskovec. 2018. Graph convolutional neural networks for web-scale recommender systems. In *Proceedings of the 24th ACM SIGKDD international conference on knowledge discovery & data mining*, pages 974–983.

Le Yu, Bowen Yu, Haiyang Yu, Fei Huang, and Yongbin Li. 2024. Language models are super mario: Absorbing abilities from homologous models as a free lunch.

Weizhe Yuan, Graham Neubig, and Pengfei Liu. 2021. Bartscore: Evaluating generated text as text generation. *Advances in Neural Information Processing Systems*, 34:27263–27277.

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

Murong Yue, Jie Zhao, Min Zhang, Liang Du, and Ziyu Yao. 2024. Large language model cascades with mixture of thought representations for cost-efficient reasoning. In *The Twelfth International Conference on Learning Representations*.

Rowan Zellers, Ari Holtzman, Yonatan Bisk, Ali Farhadi, and Yejin Choi. 2019. Hellaswag: Can a machine really finish your sentence? *arXiv preprint arXiv:1905.07830*.

Biao Zhang, Behrooz Ghorbani, Ankur Bapna, Yong Cheng, Xavier Garcia, Jonathan Shen, and Orhan Firat. 2022. Examining scaling and transfer of language model architectures for machine translation. In *International Conference on Machine Learning*, pages 26176–26192. PMLR.

Jieyu Zhang, Ranjay Krishna, Ahmed H Awadallah, and Chi Wang. 2023a. Ecoassistant: Using llm assistant more affordably and accurately. *arXiv preprint arXiv:2310.03046*.

Longteng Zhang, Lin Zhang, Shaohuai Shi, Xiaowen Chu, and Bo Li. 2023b. Lora-fa: Memory-efficient low-rank adaptation for large language models fine-tuning. *arXiv preprint arXiv:2308.03303*.

Tianyi Zhang*, Varsha Kishore*, Felix Wu*, Kilian Q. Weinberger, and Yoav Artzi. 2020. Bertscore: Evaluating text generation with bert. In *International Conference on Learning Representations*.

Yi-Kai Zhang, Ting-Ji Huang, Yao-Xiang Ding, De-Chuan Zhan, and Han-Jia Ye. 2023c. Model spider: Learning to rank pre-trained models efficiently. *Advances in Neural Information Processing Systems*, 36:13692–13719.

Yi-Kai Zhang, De-Chuan Zhan, and Han-Jia Ye. 2025. Capability instruction tuning: A new paradigm for dynamic llm routing. *arXiv preprint arXiv:2502.17282*.

Wenkuan Zhao, Shanshan Zhong, Yifan Liu, Wushao Wen, Jinghui Qin, Mingfu Liang, and Zhongzhan Huang. 2025. Dvib: Towards robust multimodal recommender systems via variational information bottleneck distillation. In *Proceedings of the ACM Web Conference 2025*.

Xiangyu Zhao, Haochen Liu, Wenqi Fan, Hui Liu, Jiliang Tang, and Chong Wang. 2021a. Autoloss: Automated loss function search in recommendations. In *Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining*, pages 3959–3967.

Xiangyu Zhao, Haochen Liu, Wenqi Fan, Hui Liu, Jiliang Tang, Chong Wang, Ming Chen, Xudong Zheng, Xiaobing Liu, and Xiwang Yang. 2021b. Autoemb: Automated embedding dimensionality search in streaming recommendations. In *2021 IEEE International Conference on Data Mining (ICDM)*, pages 896–905. IEEE.

Xinyu Zhao, Guoheng Sun, Ruisi Cai, Yukun Zhou, Pingzhi Li, Peihao Wang, Bowen Tan, Yexiao He, Li Chen, Yi Liang, et al. 2024a. Model-glue: Democratized llm scaling for a large model zoo in the wild. *arXiv preprint arXiv:2410.05357*.

Zihuai Zhao, Wenqi Fan, Jiatong Li, Yunqing Liu, Xiaowei Mei, Yiqi Wang, Zhen Wen, Fei Wang, Xiangyu Zhao, Jiliang Tang, et al. 2024b. Recommender systems in the era of large language models (llms). *IEEE Transactions on Knowledge and Data Engineering*.

Ziyu Zhao, Leilei Gan, Guoyin Wang, Wangchunshu Zhou, Hongxia Yang, Kun Kuang, and Fei Wu. 2024c. LoraRetriever: Input-aware LoRA retrieval and composition for mixed tasks in the wild. In *Findings of the Association for Computational Linguistics: ACL 2024*, pages 4447–4462, Bangkok, Thailand. Association for Computational Linguistics.

Lei Zheng, Vahid Noroozi, and Philip S Yu. 2017. Joint deep modeling of users and items using reviews for recommendation. In *Proceedings of the tenth ACM international conference on web search and data mining*, pages 425–434.

Shanshan Zhong, Shanghua Gao, Zhongzhan Huang, Wushao Wen, Marinka Zitnik, and Pan Zhou. 2024a. Moextend: Tuning new experts for modality and task extension. *arXiv preprint arXiv:2408.03511*.

Shanshan Zhong, Zhongzhan Huang, Shanghua Gao, Wushao Wen, Liang Lin, Marinka Zitnik, and Pan Zhou. 2024b. Let's think outside the box: Exploring leap-of-thought in large language models with creative humor generation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 13246–13257.

Shanshan Zhong, Zhongzhan Huang, Daifeng Li, Wushao Wen, Jinghui Qin, and Liang Lin. 2024c. Mirror gradient: Towards robust multimodal recommender systems via exploring flat local minima. In *Proceedings of the ACM Web Conference 2024*, pages 3700–3711.

Shanshan Zhong, Zhongzhan Huang, Wushao Wen, Zhijing Yang, and Jinghui Qin. 2023a. Esa: excitation-switchable attention for convolutional neural networks. *Neurocomputing*, 557:126706.

Shanshan Zhong, Wushao Wen, Jinghui Qin, Qiangpu Chen, and Zhongzhan Huang. 2023b. Lsas: Lightweight sub-attention strategy for alleviating attention bias problem. In *2023 IEEE International Conference on Multimedia and Expo (ICME)*, pages 2051–2056. IEEE.

Jeffrey Zhou, Tianjian Lu, Swaroop Mishra, Siddhartha Brahma, Sujoy Basu, Yi Luan, Denny Zhou, and Le Hou. 2023. Instruction-following evaluation for large language models. *arXiv preprint arXiv:2311.07911*.

Yanqi Zhou, Tao Lei, Hanxiao Liu, Nan Du, Yanping Huang, Vincent Y Zhao, Andrew M. Dai, Zhifeng Chen, Quoc V Le, and James Laudon. 2022. Mixture-of-experts with expert choice routing. In *Advances in Neural Information Processing Systems*.

Yuyan Zhou, Liang Song, Bingning Wang, and Weipeng Chen. 2024. Metagpt: Merging large language models using model exclusive task arithmetic. *Preprint*, arXiv:2406.11385.

Michael Zhu and Suyog Gupta. 2017. To prune, or not to prune: exploring the efficacy of pruning for model compression. *arXiv preprint arXiv:1710.01878*.

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

Richard Zhuang, Tianhao Wu, Zhaojin Wen, Andrew Li, Jiantao Jiao, and Kannan Ramchandran. 2024b. EmbedLLM: Learning compact representations of large language models. *Preprint*, arXiv:2410.02223.

Barret Zoph, Golnaz Ghiasi, Tsung-Yi Lin, Yin Cui, Hanxiao Liu, Ekin Dogus Cubuk, and Quoc Le. 2020. Rethinking pre-training and self-training. *Advances in neural information processing systems*, 33:3833–3845.

A The More Details of RouterEval

In this paper, we have considered 12 benchmarks across areas such as knowledge-based Q&A, commonsense reasoning, semantic understanding, mathematical reasoning, and instruction following, etc., including ARC, HellaSwag, MMLU, TruthfulQA, Winogrande, GSM8k, IFEval, BBH, GPQA, MUSR, MATH Lvl 5, and MMLU-PRO (Zellers et al., 2019; Clark et al., 2018; Wang et al., 2024; Hendrycks et al., 2020; Lin et al., 2021; Sakaguchi et al., 2021; Cobbe et al., 2021; Zhou et al., 2023; Suzgun et al., 2022; Rein et al., 2024; Sprague et al., 2023). Next, the test samples for each benchmark are split into training, validation, and test sets in a ratio of 8:1:1, as detailed in Table 4.

Furthermore, as discussed in Section 5.1, we need to designate a representative and high-performing LLM for each benchmark to construct a reference value. For most benchmarks, GPT-4, given its widespread application and popularity, is highly suitable for constructing reference values. However, some benchmarks lack evaluation results for GPT-4. Therefore, we individually identify representative LLMs from these benchmarks to construct reference values. When the final performance of the model constructed under the Routing LLMs paradigm exceeds a reference value of 1, it indicates that the mechanism can enable relatively weak candidates to collaborate and surpass the performance of some well-known commercial LLMs.

B The Details of LLMs Considered

Due to the involvement of different LLMs in constructing performance records for various benchmarks, the LLMs associated with each benchmark vary. Given that each benchmark involves thousands of LLMs and that table space is limited, we are unable to directly display the complete list of all LLMs. Please refer to our code for detailed information. In this section, we provide statistical results of these LLMs for reference.

On the one hand, we specifically show the number of LLMs involved in each benchmark in Table 4. It should be noted that some performance records contain duplicates or errors, and the performance of some open-source models collected is very low (below 0.1), which has almost no reference value. We have carefully filtered these data. It can be seen that the number of LLMs involved in different benchmarks ranges from 1800 to 5000, with a total of more than 8,500 LLMs involved.

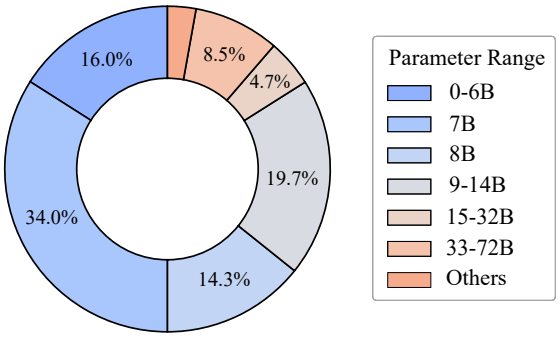

Figure 4: **The Distribution of Model Parameters**. We conduct a statistical analysis of the parameter counts for all LLMs considered in this paper and find that models with 7B parameters are predominant.

Furthermore, we have conducted a statistical analysis of the parameters of all LLMs involved in this study, as shown in Fig. 4. It can be observed that the majority of these LLMs are 7B models, most of which are relatively weak open-source models. Specifically, in Fig. 5, we present the performance distribution of these models across 12 different benchmarks. Nevertheless, the experiments in the main text have demonstrated that even using relatively weak models as candidates, the Routing LLMs paradigm can still fully leverage the strengths of each model, achieve complementary advantages, and ultimately achieve outstanding overall performance.

C Related Work

C.1 Routing LLMs

LLMs trained on massive datasets with trillions of tokens (Radford et al., 2019; Brown et al., 2020) have transformed natural language processing and other research areas. However, deploying these models for specific tasks—such as classification or question-answering—presents distinct challenges. Traditional model selection techniques in statistics and machine learning, such as k-fold cross-validation, estimate population errors for in-distribution data (Bishop and Nasrabadi, 2006; Hastie et al., 2009; Raschka, 2018; Hendy et al., 2023; Hari and Thomson, 2023a). Yet, these methods are impractical for LLMs due to their vast, often inaccessible training data and the computational infeasibility of retraining them repeatedly.

To address this, routing (Frick et al., 2025; Zhang et al., 2025, 2023c; Ramírez et al., 2023; Mohammadshahi et al., 2024; Nguyen et al., 2024;

Benchmark	#LLMs	#Train	#Val	#Test	Ref. Per.	Ref. Name
ARC (Clark et al., 2018)	5000	937	117	118	0.852	GPT-3.5
HellaSwag (Zellers et al., 2019)	5000	8033	1004	1005	0.953	GPT-4
MMLU (Wang et al., 2024)	5000	11215	1397	1430	0.864	GPT-4
TruthfulQA (Lin et al., 2021)	5000	653	82	82	0.669	Qwen1.5-32B
Winogrande (Sakaguchi et al., 2021)	5000	1013	127	127	0.875	GPT-4
GSM8k (Cobbe et al., 2021)	5000	1055	132	132	0.920	GPT-4
IFEval (Zhou et al., 2023)	3811	432	54	55	0.769	GPT-4
BBH (Suzgun et al., 2022)	3811	4607	577	577	0.830	GPT-4
GPQA (Rein et al., 2024)	3811	952	120	120	0.397	GPT-4
MUSR (Sprague et al., 2023)	3811	604	76	76	0.699	GPT-4
MATH Lvl 5 (Hendrycks et al., 2021)	3811	1057	131	136	0.400	GPT-4
MMMU-PRO (Wang et al., 2024)	1823	9625	1203	1204	0.637	GPT-4 Turbo

Table 4: **The Details of Each Benchmark in RouterEval.** The notation "#LLMs" represents the number of LLMs included in the performance records of each benchmark. The terms "#Train/#Val/#Test" denote the respective counts of training, validation, and test datasets constructed for each benchmark within RouterEval. "Ref.Name" and "Ref.Per." indicate the name and performance of the LLM used to establish the reference value in Section 5.1. Avatar All benchmarks involve a total of 8,576 LLMs. In RouterEval, we additionally provide over 200,000,000 performance records of these LLMs across different benchmarks.

Liu et al., 2024; Ramírez et al., 2024; Zhang et al., 2023a) offers an efficient solution by dynamically selecting the most suitable LLM for a given input, eliminating the need to evaluate all possible models. Routing strategies fall into two main categories: non-predictive and predictive. Non-predictive approaches, like FrugalGPT (Chen et al., 2023), sequentially invoke LLMs until an output meets a predefined quality threshold determined by a judger. Other non-predictive methods use layered frameworks to escalate complex queries to more advanced models (Wang et al., 2023b) or combine smaller models with LLMs (Madaan et al., 2023; Yue et al., 2023; Lee et al., 2023a; Lu et al., 2023b; Hu et al., 2024b).

In contrast, predictive routing leverages machine learning techniques—such as supervised learning (Shnitzer et al., 2023b), reward models (Hari and Thomson, 2023b; Lu et al., 2023b), or meta-models predicting performance scores (Šakota et al., 2023)—to preemptively identify the optimal LLM, thereby reducing latency and computational overhead. Recent advancements have enhanced routing's effectiveness through innovative techniques, including neural network-based meta-models (Ding et al., 2024a; Šakota et al., 2024; Chen et al., 2024b; Aggarwal et al., 2024), k-nearest neighbors (Hu et al., 2024b; Shnitzer et al., 2023c; Stripelis et al., 2024; Lee et al., 2024), matrix factorization (Ong et al., 2024; Zhuang et al., 2024b; Li, 2025), and graph neural networks (Feng et al., 2024). These methods enable coordination among multiple LLMs or even sub-models within a single framework, such as MatFormer (Devvrit et al., 2024; Cai et al., 2024).

Unlike earlier approaches that required scoring outputs from every candidate LLM (Liu and Liu, 2021; Ravaut et al., 2022a; Jiang et al., 2023a), modern routing uses benchmark data to map each LLM's strengths across tasks and domains, requiring inference only from the chosen model. Further developments include supervised router training (Lu et al., 2024b; Zhao et al., 2024c) and efforts to improve robustness (Dann et al., 2024; Montreuil et al., 2025; Shafran et al., 2025), highlighting routing's ability to balance cost and performance.

Despite these innovations, the growing variety of routing strategies lacks a unified evaluation standard. The proposed RouterEval aims to bridge that gap by proposing a systematic framework to assess router effectiveness, providing a foundation for future advancements.

C.2 Model Ensemble

LLM ensemble methods offer a compelling strategy for leveraging the collective strengths of multiple LLMs to boost performance, efficiency, and robustness in NLP tasks. Departing from the conventional reliance on a single model for inference, LLM ensembles integrate outputs from diverse models, either by aggregating predictions after inference or by orchestrating their contri-

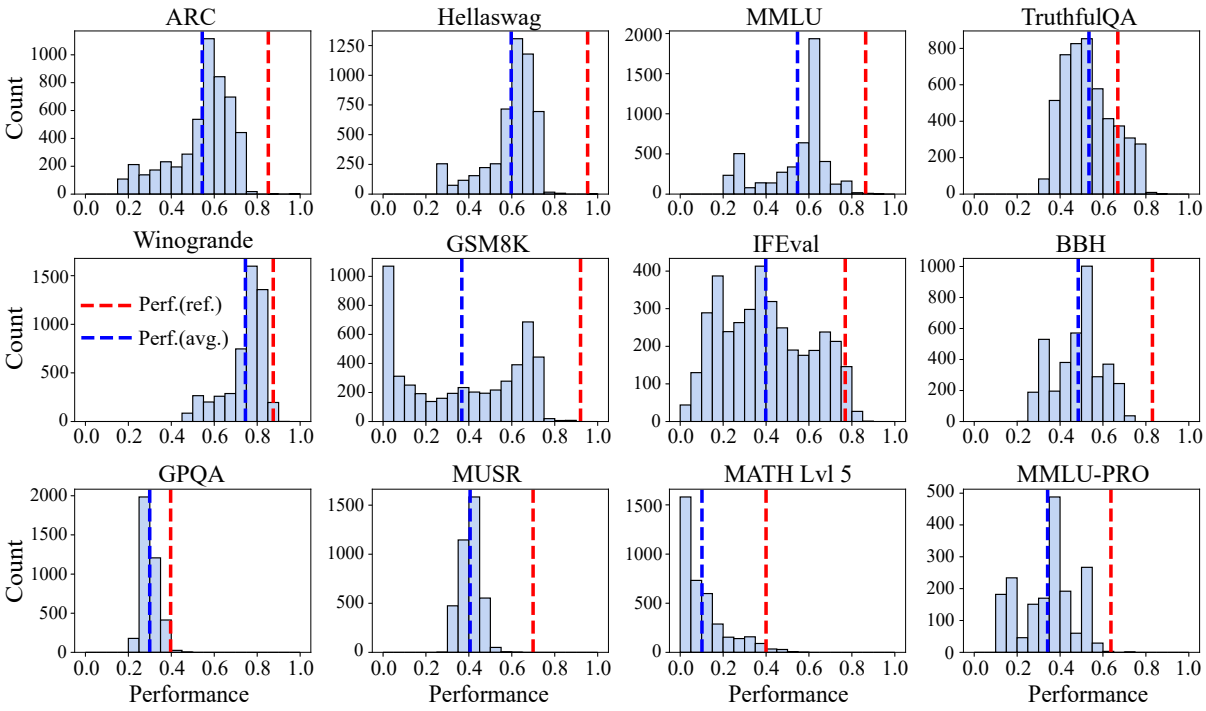

Figure 5: **The Distribution of Model Performance Under 12 Benchmarks**. We present the performance distribution of the LLMs involved in each of the 12 benchmarks. As shown in Fig. 4, the majority of LLMs are 7B models, which tend to exhibit relatively weaker performance across the benchmarks from a statistical standpoint.

butions throughout the process. Drawing inspiration from classical machine learning ensemble techniques—such as bagging and boosting—these methods are tailored to tackle the distinct challenges of modern LLMs, including their immense scale, computational demands, and architectural complexity.

A notable approach within LLM ensembling is the development of LLM cascades, designed to optimize inference efficiency without compromising output quality. For example, Chen et al. (2023) propose a sequential framework in which LLMs are arranged in order of increasing parameter size. The process starts with a smaller, computationally lightweight model producing an initial output. If this output satisfies a predefined quality threshold, the cascade terminates, delivering the result; if not, the task escalates to progressively larger and more capable models. Similarly, Yue et al. (2024) introduce a verification-based cascade, where a smaller LLM generates a preliminary answer that is then evaluated for accuracy. Should it fall short, a more powerful LLM steps in to refine or correct the response. These cascading strategies significantly alleviate the computational load of depending exclusively on large-scale LLMs, making them especially valuable for resource-limited scenarios.

Another promising avenue in LLM ensemble research centers on generating a range of candidate outputs from distinct LLMs and then selecting or synthesizing the optimal result. This method capitalizes on the diversity of model architectures, training datasets, and reasoning abilities to elevate overall performance. For instance, Lee et al. (2023b) employ this technique to enhance instruction-tuning data construction, choosing the most effective instruction from a pool of candidates produced by various LLMs. In a similar vein, Jiang et al. (2023a) investigate unsupervised evaluation metrics—such as BERTScore (Zhang* et al., 2020), BLEURT (Sellam et al., 2020), BARTScore (Yuan et al., 2021), and scores derived from ChatGPT—to rank and select the strongest output from a set of candidates. However, their work reveals a key challenge: the effectiveness of this selection hinges on the quality and variety within the candidate pool. To address this limitation, Jiang et al. (2023a) propose an advanced fusion model that takes the highest-ranked candidates as inputs and generates a polished final output, seamlessly blending the strengths of individual predictions.

Overall, the LLMs ensemble paradigm, for a given input, emphasizes requiring all candidate LLMs to perform inference, followed by aggregat-

ing and organizing their outputs. The final result is typically determined through strategies like majority voting. In contrast, the Routing LLMs paradigm prioritizes efficiency by assigning tasks to specific candidates before inference, thereby reducing computational overhead and enhancing efficiency. Naturally, as these paradigms are not entirely distinct and can draw inspiration from one another, there remains a degree of technical overlap, particularly in various optimization techniques and improvement strategies.

C.3 Scaling Law

Scaling laws (Li et al., 2025) have become a cornerstone for deciphering the behavior of deep learning models across a wide array of domains and tasks. They provide critical insights into how performance correlates with key factors such as dataset size, model capacity, and computational resources. Early work by Banko and Brill (2001) laid the groundwork by identifying a power-law relationship between validation error and training dataset size in tasks like confusion set disambiguation. Their findings revealed that as the dataset grows, the average error decreases predictably, while the model size needed to effectively fit the data scales log-linearly. This seminal observation was later expanded by Amodei et al. (2016), who demonstrated power-law improvements in word error rate with increased training data for the 38M-parameter Deep Speech 2 model, and by Hestness et al. (2017), who extended these exponential trends to diverse fields such as machine translation, language modeling, image processing, and speech recognition. These studies collectively highlighted the robustness of scaling laws, showing that performance gains remain consistent even as models and architectures evolve.

Building on this foundation, subsequent research has pushed the boundaries of scale while refining the implications of these laws. For instance, Kaplan et al. (2020) investigated models with up to 1.5B parameters trained on 23B tokens, deriving power-law relationships to optimize computational budget allocation. However, later critiques from Hoffmann et al. (2022) and Hu et al. (2024c) pointed out an underestimation of required training data, underscoring subtle methodological challenges in scaling studies. Beyond sheer scale, researchers have delved into more nuanced phenomena: Wei et al. (2022) identified emergent abilities in large language models that are absent in smaller ones, while Hernandez et al. (2021) explored scaling laws in transfer learning and finetuning contexts. Architectural diversity has also come under scrutiny, with Tay et al. (2022) showing that not all model designs scale equally, advocating for scaling studies to inform architecture development. The adaptability of scaling laws to emerging paradigms is evident in recent innovations. Hybrid models like Mamba (Gu and Dao, 2023), analyzed by Poli et al. (2024), alongside specialized scaling laws for mixture-of-experts models (Clark et al., 2022; Fedus et al., 2022a; Shazeer et al., 2017) and sparse architectures (Frantar et al., 2023; Zhu and Gupta, 2017), demonstrate their versatility. The scope of scaling laws extends far beyond language tasks, encompassing vision-language models (Cherti et al., 2023; Henighan et al., 2020), reinforcement learning (Hilton et al., 2023; Gao et al., 2023), and recommendation systems (Ardalani et al., 2022), underscoring their wide-ranging applicability.

While Transformer-based models (Vaswani, 2017)—exemplified by behemoths like Llama 3 with 405B parameters—dominate scaling law research due to their exceptional scalability, alternative architectures have not been overlooked. For comparison, ResNet101 boasts a modest 44M parameters (Sorscher et al., 2022), where Sorscher et al. (2022) investigated data pruning laws. Smaller-scale studies have also employed MLPs or SVMs (Hashimoto, 2021) to probe scaling behavior. Additional dimensions, such as the impact of data quality and language-specific effects on scaling coefficients (Bansal et al., 2022; Zhang et al., 2022), as well as multimodal scaling in foundation models (Aghajanyan et al., 2023), further enrich this research landscape.

Together, these efforts illustrate that scaling laws do more than predict performance—they serve as a guiding framework for resource allocation, architecture design, and generalization across domains. As such, they have become an indispensable tool in advancing artificial intelligence research and its real-world deployment, offering a lens through which we can better understand and harness the potential of ever-growing models and datasets.

C.4 Recommender System

Recommender systems (Zhao et al., 2024b) have emerged as essential tools to tackle the challenge of information overload, offering tailored content and services to users across diverse online platforms (Wu et al., 2022; Fan et al., 2022a). These systems

primarily rely on two core methodologies: Collaborative Filtering and Content-based recommendation. Collaborative Filtering, the most prevalent approach, harnesses historical user-item interactions—such as purchase histories or ratings—to uncover behavioral similarities among users and forecast their future preferences (Fan et al., 2019b). A key technique within Collaborative Filtering, Matrix Factorization, transforms discrete user and item identities into continuous embedding vectors, facilitating efficient computation of recommendation scores (Fan et al., 2018, 2019a; Zhao et al., 2021a,b). In contrast, content-based methods enhance precision by incorporating supplementary data, such as user demographics or item descriptions, with a special focus on widely available textual information (Vasile et al., 2016).

The rise of deep learning has revolutionized recommender systems, unlocking advanced representation learning capabilities (Fan et al., 2022b). For example, Neural Matrix Factorization leverages deep neural networks to model non-linear interactions between users and items, outperforming traditional inner product techniques (He et al., 2017). Meanwhile, Graph Neural Networks have gained traction by representing user-item relationships as graph-structured data, employing message propagation to generate insightful node embeddings (Ying et al., 2018; Fan et al., 2020; Ma and Tang, 2021; Derr et al., 2020). To incorporate textual insights, models like DeepCoNN utilize Convolutional Neural Networks to process user reviews, thereby improving rating predictions (Zheng et al., 2017). Similarly, NARRE introduces a neural attention mechanism that not only predicts ratings but also delivers review-level explanations (Chen et al., 2018). These breakthroughs highlight the profound influence of deep learning on enhancing recommendation quality.

In recent developments, the incorporation of language models has propelled recommender systems to new heights by tapping into their ability to comprehend and generate human-like natural language (Wu et al., 2020, 2023; Dongre and Agrawal, 2023). For instance, BERT4Rec employs Bidirectional Encoder Representations from Transformers to capture the sequential patterns in user behavior, significantly improving sequential recommendation tasks (Sun et al., 2019). Likewise, transformer-based approaches, such as the framework by (Liu et al., 2023), leverage the generative power of Transformers to recommend items while simultaneously crafting explanatory narratives. These innovations enable highly contextualized and personalized recommendations, with applications ranging from news curation (Wu et al., 2020) to drug suggestions (Dongre and Agrawal, 2023). This convergence of natural language processing and recommender system design exemplifies their dynamic evolution, promising ever more sophisticated and user-centric solutions.

In essence, Routing LLMs can be regarded as a specialized subset of RS. In this framework, the input corresponds to the user in a traditional RS, the pool of LLM candidates represents the items to be recommended, and the performance record serves as the interaction history between users and items. The router's task is to select an appropriate LLM from this pool based on the given input, aiming to optimize for various goals such as maximizing accuracy, minimizing computational cost, or reducing hallucination. Unlike conventional RS, however, Routing LLMs face a significant constraint: the "user information" available for collection is extremely limited, and annotating or labeling this data poses a considerable challenge. This scarcity of detailed input data complicates the routing process, requiring innovative approaches to achieve effective recommendations.

C.5 LLM Model Fusion

Model merging (Lu et al., 2024a) has become a pivotal technique in the realm of LLMs, enabling the integration of strengths from multiple pre-trained or fine-tuned models to boost performance, adaptability, and efficiency. This approach can be broadly divided into two paradigms: zero-shot merging, which fuses models without further training, and merge-then-train, which involves refining the combined model after integration.

Early zero-shot techniques, such as weight averaging (Nagarajan and Kolter, 2021; Wortsman et al., 2022) and Linear Mode Connectivity, laid the groundwork, evolving into more advanced methods like Task Arithmetic(Ilharco et al., 2023a)—where task vectors steer parameter adjustments—and TIES(Yadav et al., 2023a), which reduces interference through trimming and conflict resolution. Recent innovations, including DARE(Yu et al., 2024) and Evolutionary Model Merge(Akiba et al., 2024a), further refine this by optimizing selective parameters or inference pathways, all without additional training. On the other hand, merge-then-train strategies, such as Fisher Merging(Matena

and Raffel, 2022), utilize the Fisher information matrix to assign parameter weights, while RegMean(Jin et al., 2023) fine-tunes linear merging on a per-layer basis, carefully balancing embeddings and biases. However, both approaches encounter difficulties when merging models with divergent initializations, prompting research into permutation symmetries (Ainsworth et al., 2022; Verma and Elbayad, 2024) to better align parameters. A notable distinction exists between model merging and *model fusion*. Merging typically aims for efficiency by creating a single, cohesive model (Singh and Jaggi, 2020), whereas fusion often combines multiple models to enhance quality, potentially at the expense of speed (Ravaut et al., 2022b; Jiang et al., 2023b).

Within the merging domain, weighted-average techniques—refined by methods like Hessian-based estimates (Daheim et al., 2023) or pruning-enhanced Fisher weights (Nathan et al., 2024)—adjust parameter significance but may fail to capture task-specific subtleties, resulting in performance degradation (e.g., a reported 10% drop with basic averaging (Ilharco et al., 2023b)). To counter this, the notion of *task vectors*, defined as $\boldsymbol{\tau}_t = \boldsymbol{\theta}_t^{\text{ft}} - \boldsymbol{\theta}^{\text{pre}}$ (Ilharco et al., 2023b), has gained prominence. These vectors encapsulate task-specific shifts in parameter space, facilitating precise conflict resolution during merging. Building on this, methods like Task Arithmetic (Ilharco et al., 2023b), AdaMerging (Yang et al., 2023), and TIES-Merging (Yadav et al., 2023b) address redundancy and sign discrepancies, improving cross-model compatibility.

Resolving parameter conflicts remains a core challenge, inspiring a variety of innovative strategies. Task Arithmetic (Ilharco et al., 2023b) introduced arithmetic-based vector merging, while TIES-Merging (Yadav et al., 2023b) and AdaMerging enhance this by targeting interference sources. Evolutionary methods (Akiba et al., 2024b) blend TIES with optimized inference routes, and practical applications like MetaGPT (Zhou et al., 2024) and LLM evaluators (Kim et al., 2024) showcase real-world efficacy. Alternatively, ZipIt (Stoica et al., 2024) retains correlated parameters while preserving distinct layers, offering adaptability. Additional advancements, such as geometric weight analysis (Shoemake, 1985; Jang et al., 2024) and safety alignment (Hammoud et al., 2024), further enrich the field. For models with shared architectures and initializations, zero-shot merging stands out as a key focus, striking a balance between efficiency and performance without the computational burden of retraining. This makes it a foundational element in the ongoing evolution of LLMs.

C.6 Mixture-of-Experts (MoE)

The MoE framework, first proposed as a technique for training independent models with distinct parameters and a routing mechanism (Jacobs et al., 1991; Jordan and Jacobs, 1993), has matured into a robust strategy for scaling neural networks. Originally, MoE models were designed to delegate tasks to specialized sub-models of uniform size, providing an appealing alternative to the limitations of single-model specialization (Jang et al., 2023; Douillard et al., 2024). With sub-models of equal capacity, the routing rule—responsible for directing inputs to the appropriate expert—did not need to account for varying computational costs. Over time, however, the paradigm has shifted toward integrating MoE into larger architectures, such as Transformers, where sub-models function as interconnected components within a cohesive system (Fedus et al., 2022b; Zhou et al., 2022). This evolution has unlocked the power of sparse activation, where only a subset of parameters is engaged for each input, significantly boosting efficiency without sacrificing performance. A prime example is Mixtral (Jiang et al., 2024), which competes with dense LLMs while activating far fewer parameters.

Modern MoE implementations have further refined this approach, with innovations like those from Shazeer et al. (2017), who introduced router networks to dynamically activate specific experts for individual input tokens. This technique has become a cornerstone of LLMs, celebrated for its generative capabilities and computational efficiency. Building on this, model mixture techniques have expanded the MoE framework by incorporating diverse dense LLM models—regardless of their size—into a unified system. For instance, Branch-Train-MiX (Sukhbaatar et al., 2024) starts with a seed dense LLM, branches into parallel expert models during training, and later merges them into MoE layers by averaging non-expert parameters; yet, this method is limited to models sharing identical architectures. In contrast, model fusion strategies (Wan et al., 2024; Wang et al., 2023a) blend expert outputs to harness insights from distinct data distributions. Most recently, UltraFuser (Ding et al., 2024b) has pushed the boundaries further with a token-level soft gating mechanism and a two-stage training process, offering enhanced flex-

m	**Group**	**Oracle** r_o	$r_o(0.5)$	**Details**
3	all-strong	0.972	0.924	[0.869, 0.878, 0.879]
3	all_weak	0.765	0.583	[0.406, 0.392, 0.408]
3	strong-to-weak	0.938	0.755	[0.241, 0.599, 0.878]
5	all-strong	0.983	0.930	[0.863, 0.869, 0.878, 0.879, 0.892]
5	all_weak	0.843	0.566	[0.279, 0.300, 0.313, 0.282, 0.276]
5	strong-to-weak	0.943	0.761	[0.330, 0.462, 0.615, 0.622, 0.869]
10	all-strong	0.984	0.881	[0.766, 0.776, 0.781, 0.772, 0.771, 0.783, 0.787, 0.780, 0.778, 0.782]
10	all-weak	0.955	0.620	[0.256, 0.299, 0.276, 0.296, 0.284, 0.287, 0.274, 0.280, 0.301, 0.278]
10	strong-to-weak	0.981	0.765	[0.245, 0.259, 0.487, 0.534, 0.577, 0.614, 0.606, 0.628, 0.658, 0.869]
100	all-strong	0.996	0.890	See Fig. 6
100	all-weak	1.000	0.630	See Fig. 6
100	strong-to-weak	1.000	0.769	See Fig. 6
1000	all-strong	1.000	0.839	See Fig. 6
1000	all-weak	1.000	0.642	See Fig. 6
1000	strong-to-weak	1.000	0.769	See Fig. 6

Table 5: **The Details of Candidate Groups on MMLU.** m denotes the number of LLMs per candidate group. "Details" shows the performance of each LLM in the corresponding group. When $m = 100$ or $m = 1000$, the number of candidates is too large to present individually. Therefore, we only display their performance distributions in Fig. 6. For specific performance metrics and the details of other evaluations, please refer directly to our code.

ibility in combining expert contributions. These developments highlight MoE's remarkable ability to balance scalability, specialization, and resource efficiency, cementing its role as a pivotal advancement in contemporary machine learning research. The Routing LLMs paradigm is a special type of MoE. Experts can be seen as LLMs in a candidate pool, and are selected by a router to process a given input. While MoE generally can choose multiple "experts", the current Routing LLMs paradigm only selects one LLM.

D The Details of Candidate Groups

In Section 4.2, we set three candidate types for the given benchmark and m, namely "all-strong," "all-weak," and "strong-to-weak." The final model performance is the average of the results from these three candidates. Specifically, we sort all N LLMs with performance between 0.1 and 0.9 based on their individual performance on the given benchmark, obtaining $\{\ell'_i\}_{i=1}^{N}$. We then consider the following optimization problem regarding the performance of G and its corresponding oracle r_o, as shown in Eq. (8),

$$\hat{G} = \max_G \text{Perf.}(r_o, G). \tag{8}$$

When the m LLMs in G are all selected from $\{\ell'_i\}_{i=1}^{\lfloor 0.2N \rfloor}$ and $\{\ell'_i\}_{i=\lfloor 0.8N \rfloor}^{N}$ respectively, $\hat{G}$ forms the "all-strong" group and the "all-weak" group. Meanwhile, $\hat{G}$ forms the "strong-to-weak" group when the j-th LLM in G is selected from $\{\ell'_i\}_{i=(j-1)m}^{\min(jm,N)}$.

By solving Eq. (8), we obtained the candidate selections for the three group types under the given benchmark and m. For example, in the case of MMLU, we detail these selections in Table 5. Additionally, for the settings of $m = 100$ and $m = 1000$, due to space limitations, we only show their performance distributions in Fig. 6. For specific candidates of these settings and other evaluation candidates, please refer directly to our code.

E The Reproducing Details of Baselines

In this section, we provide the specific implementation methods for all the existing routers mentioned in Section 5.2. Since some settings may not have been specifically discussed in their original papers, we attempt to supplement them. Specifically,

- **LinearR**: A simple linear layer is used as the classifier. The input to the linear layer is the query representation, and the output dimension corresponds to the number of LLMs in the candidate pool, representing the selection scores for each LLM. As mentioned in Section 4.1, we employ RoBERTa as the example encoder. The output dimension of RoBERTa is 768, hence the input dimension of the linear layer is also 768. During training, BCEWithLogitsLoss is used as the loss function with a batch size of 1, learning rate of $1e-2$, and trained for 10 epochs. During testing, the

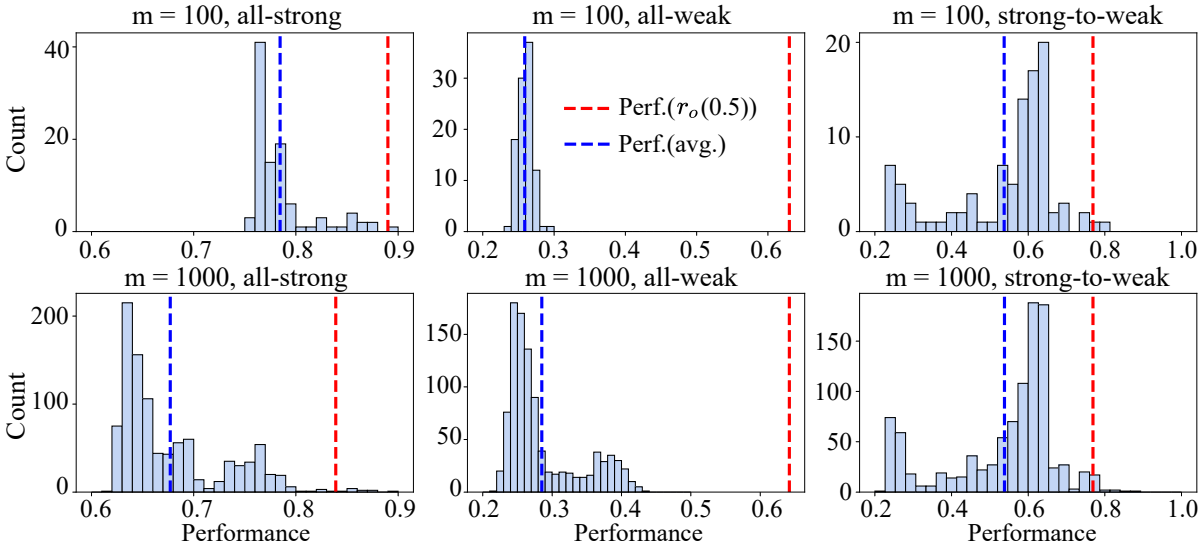

Figure 6: **The Distribution of Model Performance in Candidate Group on MMLU**. Due to the space limitations of Table 5, it is not feasible to present the detailed performance of all candidates in the candidate groups where $m = 100$ or $m = 1000$. Therefore, we illustrate their performance distributions in this figure. For specific performance metrics and the details of other evaluations, please refer directly to our code.

query representation is input to obtain an m-dimensional score vector, and the LLM with the highest score is selected for response generation.

- **MLPR**: A Multi-Layer Perceptron (MLP) serves as the classifier, with the hidden layer size set to 256. The input is the 768-dimensional query representation, and the output is an m-dimensional score vector, consistent with LinearR. Training employs BCEWithLogitsLoss with a batch size of 1, learning rate of $1e-4$, and runs for 100 epochs.

- **MLC**: To handle the multi-label property of text embeddings, a multi-class classification(MLC) approach is used. Specifically, during training, the model treats multiple labels as positive classes for supervised learning. During inference, the input embedding passes through the classifier to generate a probability distribution, and select the highest one as the final router result. Training employs BCEWithLogitsLoss with a batch size of 10, and runs for 10 epochs.

- **C-RoBERTa**: We used K-Means to cluster the text embeddings of the training set into k clusters. Specifically, for each cluster, the performance of each model in the candidate model pool is evaluated using the samples within that cluster. Then, the best-performing model is selected as the dedicated predictor for that cluster. During inference, the Euclidean distance between the test sample and each cluster center is calculated. The sample is matched to the nearest cluster using a nearest neighbor strategy and the corresponding dedicated predictor is used as the router result. In this paper, k is set to 3.

- **PRknn**: A k-Nearest Neighbors (KNN) classifier is adopted. During testing, given a test query representation, we retrieve the K train queries with the smallest Euclidean distances. We then compute the average scores of the m models on these K train queries and select the model with the highest average score to process the test query. In our experiments, $K = 5$.

F The Results on Hard Level Settings

In the main text, we primarily present the experimental results of RouterEval at the easy level, that is, the cases where $m \in \{3, 5\}$. In this section, we provide the experimental results of RouterEval at the hard level, where $m \in \{10, 100, 1000\}$. Given the scarcity of data, the difficulty of the classification problem at the hard level is significantly higher than that at the easy level. However, as demonstrated by the model-level scaling up phenomenon shown in the main text, the Rout-

	Router	GPQA				MUSR				MATH Lvl 5				MMLU-PRO			
		$\mu_o\uparrow$	$V_R\uparrow$	$V_B\uparrow$	E_p	$\mu_o\uparrow$	$V_R\uparrow$	$V_B\uparrow$	E_p	$\mu_o\uparrow$	$V_R\uparrow$	$V_B\uparrow$	E_p	$\mu_o\uparrow$	$V_R\uparrow$	$V_B\uparrow$	E_p
$m=10$	Oracle r_o	0.93	2.34	2.58	1.58	0.88	1.26	1.99	1.97	0.71	1.77	1.96	2.17	0.81	1.27	2.73	1.96
	$r_o(0.5)$	0.61	1.54	1.70	2.92	0.63	0.90	1.41	3.01	0.51	1.28	1.36	3.04	0.59	0.92	1.75	2.98
	LinearR	**0.43**	**1.09**	**1.19**	3.15	**0.48**	**0.68**	**1.07**	3.24	0.39	0.99	1.00	3.20	**0.52**	**0.82**	**1.00**	3.08
	MLPR	0.40	1.01	1.09	3.10	0.44	0.63	0.99	3.27	0.40	0.99	0.97	3.17	0.51	0.81	0.97	3.07
	C-RoBERTa	0.34	0.85	0.90	1.03	0.41	0.59	0.91	1.54	**0.40**	**0.99**	**0.98**	0.99	0.52	0.81	1.00	0.50
	MLC	0.36	0.90	0.97	0.49	0.39	0.55	0.86	1.13	0.31	0.77	0.76	0.92	0.52	0.81	0.99	1.10
	PRknn	0.37	0.93	1.01	3.29	0.39	0.56	0.88	3.30	0.37	0.93	0.95	3.30	0.45	0.70	0.90	3.30
	Random	0.30	0.75	0.82	3.32	0.38	0.54	0.84	3.32	0.32	0.79	0.77	3.32	0.36	0.57	0.76	3.32
$m=100$	Oracle r_o	0.99	2.50	2.54	4.62	0.96	1.37	2.06	4.67	0.89	2.22	2.40	4.13	0.97	1.53	3.60	4.23
	$r_o(0.5)$	0.64	1.62	1.63	6.15	0.66	0.94	1.41	6.12	0.58	1.45	1.53	5.90	0.64	1.01	2.18	5.97
	LinearR	**0.43**	**1.09**	**1.08**	6.49	0.48	0.68	0.99	6.55	0.40	1.01	0.94	6.52	0.46	0.72	0.95	6.50
	MLPR	0.38	0.94	0.93	6.58	0.45	0.65	0.94	6.59	**0.41**	**1.04**	**1.00**	6.49	**0.46**	**0.72**	**0.97**	6.53
	C-RoBERTa	0.35	0.88	0.88	1.55	**0.49**	**0.70**	**1.02**	1.54	0.41	1.03	0.98	1.15	0.45	0.71	0.93	1.00
	MLC	0.29	0.72	0.70	1.06	0.40	0.57	0.83	3.33	0.38	0.94	0.88	0.00	0.45	0.70	0.94	2.81
	PRknn	0.43	1.07	1.04	6.61	0.38	0.54	0.78	6.62	0.36	0.89	0.88	6.62	0.36	0.56	0.83	6.62
	Random	0.29	0.74	0.73	6.64	0.36	0.51	0.75	6.64	0.27	0.67	0.67	6.64	0.32	0.50	0.75	6.64
$m=1000$	Oracle r_o	1.00	2.52	2.34	7.95	0.99	1.41	1.92	7.66	0.97	2.41	2.16	6.53	0.99	1.55	1.77	7.66
	$r_o(0.5)$	0.65	1.64	1.53	9.47	0.67	0.96	1.31	9.30	0.60	1.49	1.32	8.87	0.67	1.05	1.18	9.32
	LinearR	**0.47**	**1.18**	**1.08**	9.81	**0.54**	**0.77**	**1.03**	9.87	0.46	1.15	0.91	9.77	**0.62**	**0.98**	**1.04**	9.76
	MLPR	0.38	0.94	0.87	9.88	0.54	0.77	1.01	9.92	0.46	1.15	0.90	9.72	0.62	0.97	1.02	9.77
	C-RoBERTa	0.40	1.00	0.91	0.91	0.53	0.75	0.98	1.54	**0.46**	**1.16**	**0.92**	1.43	0.61	0.96	1.00	0.50
	MLC	0.28	0.70	0.64	0.86	0.49	0.70	0.92	5.13	0.41	1.04	0.77	1.77	0.52	0.81	0.76	5.24
	PRknn	0.40	1.01	0.93	9.94	0.29	0.41	0.53	9.94	0.28	0.70	0.59	9.94	0.40	0.63	0.67	9.94
	Random	0.31	0.77	0.71	9.97	0.36	0.51	0.69	9.97	0.23	0.57	0.48	9.97	0.35	0.55	0.58	9.97

Table 6: **The Results on Hard Level RouterEval (part1)**. See Section 5.1 for detials of various metrics. Red area and blue area highlights indicate the "Strong router" and "Existing router" mentioned in Section 5.2, respectively. The best results in existing router methods are highlighted with underlines and on bold. The values in the table are rounded to two decimal places.

ing LLMs paradigm can only exhibit its surprisingly strong performance when there are a sufficient number of LLM candidates, typically ranging from 100 to 1000 candidates. Therefore, exploring RouterEval at the hard level is highly necessary. Tables 7 and 6 show the classification performance of different routers, and the results are consistent with our analysis. Thus, the Routing LLMs paradigm still has considerable room for improvement.

G More Examples Visualization of Model-level Scaling Up

In Section 3, we discussed the model-level scaling up phenomenon on four well-known LLM benchmarks. In this section, we supplement our findings with observations from eight additional LLM benchmarks to illustrate the prevalence of this phenomenon. Specifically, similar to Section 3, we construct the oracle router r_o based on the performance record, and then define $r_o(p)$ to create other routers with different capabilities as follows:

$$r_o(p) = \begin{cases} r_o, & \text{with probability } p, \\ \omega_m, & \text{with probability } 1-p, \end{cases} \quad (9)$$

where ω_m is a router that samples uniformly from the m candidate LLMs with probability $1/m$. As $p \to 1$, $r_o(p)$ approaches the oracle router r_o, leading to the strongest classification performance among the m LLM candidates. Conversely, as $p \to 0$, $r_o(p)$ degenerates into a random sampler.

From the experimental results in Fig. 7, we can draw similar observations. On the one hand, across different benchmarks, we still observe that with the support of a capable router, the overall capability of the model increases as the number of LLM candidates grows. On the other hand, even weak candidates can achieve satisfactory performance under the routing LLMs paradigm, working together to deliver good results, even when their number is relatively small, such as in the case of 3 to 10 LLMs.

		ARC				HellaSwag				MMLU				TruthfulQA			
	Router	$\mu_o\uparrow$	$V_R\uparrow$	$V_B\uparrow$	E_p	$\mu_o\uparrow$	$V_R\uparrow$	$V_B\uparrow$	E_p	$\mu_o\uparrow$	$V_R\uparrow$	$V_B\uparrow$	E_p	$\mu_o\uparrow$	$V_R\uparrow$	$V_B\uparrow$	E_p
$m=10$	Oracle r_o	0.90	1.06	1.48	2.30	0.84	0.88	1.15	2.83	0.97	1.13	1.85	2.17	0.92	1.38	1.28	2.61
	$r_o(0.5)$	0.72	0.85	1.16	3.08	0.75	0.79	1.02	3.21	0.75	0.87	1.35	3.06	0.75	1.13	1.03	3.15
	LinearR	0.62	0.73	0.94	3.21	**0.75**	**0.79**	**1.01**	3.18	0.67	0.77	1.01	3.06	**0.77**	**1.15**	**1.00**	2.99
	MLPR	0.62	0.73	0.94	3.24	0.72	0.75	1.01	5.38	**0.67**	**0.77**	**1.02**	3.03	0.74	1.10	0.94	3.01
	C-RoBERTa	**0.62**	**0.73**	**0.96**	1.03	0.74	0.78	0.99	0.80	0.65	0.75	0.98	1.02	0.75	1.12	0.94	0.53
	MLC	0.61	0.72	0.93	1.96	0.73	0.77	0.98	2.29	0.65	0.75	0.98	1.61	0.76	1.14	0.98	1.70
	PRknn	0.61	0.72	0.96	3.30	0.70	0.73	0.94	3.31	0.60	0.69	0.92	3.29	0.69	1.03	0.92	3.29
	Random	0.54	0.63	0.84	3.32	0.66	0.70	0.90	3.32	0.54	0.62	0.85	3.32	0.59	0.88	0.78	3.32
$m=100$	Oracle r_o	0.96	1.12	1.53	4.96	0.87	0.92	1.17	5.75	1.00	1.16	1.94	5.31	0.97	1.45	1.36	5.76
	$r_o(0.5)$	0.74	0.87	1.16	6.15	0.76	0.80	1.02	6.38	0.76	0.88	1.38	6.32	0.77	1.16	1.06	6.40
	LinearR	0.64	0.75	0.93	6.55	0.75	0.79	1.00	6.57	0.66	0.76	1.01	6.51	0.74	1.10	0.94	6.48
	MLPR	0.63	0.74	0.91	6.57	0.75	0.79	0.99	6.58	**0.66**	**0.77**	**1.01**	6.52	0.74	1.11	0.93	6.47
	C-RoBERTa	**0.64**	**0.76**	**0.94**	0.52	**0.76**	**0.80**	**1.00**	0.52	0.66	0.76	1.01	1.02	0.75	1.12	0.94	0.53
	MLC	0.64	0.75	0.92	4.20	0.75	0.79	0.99	6.21	0.64	0.74	0.95	3.69	**0.77**	**1.16**	**0.99**	4.05
	PRknn	0.56	0.66	0.84	6.63	0.67	0.70	0.89	6.63	0.55	0.63	0.86	6.62	0.68	1.01	0.91	6.62
	Random	0.52	0.61	0.78	6.64	0.65	0.68	0.87	6.64	0.53	0.61	0.82	6.64	0.58	0.86	0.76	6.64
$m=1000$	Oracle r_o	0.99	1.16	1.45	7.76	0.92	0.97	1.24	8.28	1.00	1.16	1.53	8.56	0.99	1.47	1.35	8.90
	$r_o(0.5)$	0.74	0.86	1.06	9.27	0.76	0.80	1.01	9.40	0.75	0.87	1.10	9.63	0.76	1.14	1.02	9.65
	LinearR	0.69	0.82	0.93	9.85	0.75	0.79	0.99	9.88	**0.75**	**0.87**	**1.04**	9.79	0.75	1.11	0.95	9.78
	MLPR	0.68	0.80	0.91	9.86	0.76	0.79	0.99	9.88	0.75	0.87	1.03	9.80	0.72	1.07	0.90	9.76
	C-RoBERTa	**0.70**	**0.82**	**0.94**	0.52	**0.76**	**0.80**	**1.00**	0.52	0.73	0.84	1.00	1.02	**0.77**	**1.15**	**0.95**	0.53
	MLC	0.63	0.74	0.80	6.41	0.75	0.79	0.98	8.81	0.71	0.82	0.95	6.31	0.62	0.93	0.77	6.26
	PRknn	0.49	0.57	0.66	9.95	0.61	0.64	0.80	9.95	0.50	0.58	0.69	9.94	0.67	1.00	0.89	9.94
	Random	0.49	0.57	0.67	9.97	0.60	0.63	0.78	9.97	0.50	0.58	0.68	9.97	0.54	0.81	0.69	9.97
		Winogrande				**GSM8k**				**IFEval**				**BBH**			
$m=10$	Oracle r_o	0.99	1.14	1.36	2.69	0.93	1.02	1.46	2.45	0.85	1.10	1.43	2.38	0.94	1.13	1.79	2.18
	$r_o(0.5)$	0.85	0.97	1.14	3.19	0.76	0.82	1.15	3.13	0.67	0.88	1.07	3.10	0.72	0.87	1.33	3.06
	LinearR	0.72	0.82	0.91	3.18	0.70	0.77	0.99	3.06	**0.70**	**0.91**	**1.02**	2.77	0.61	0.73	1.02	3.12
	MLPR	0.74	0.85	0.95	3.15	0.70	0.76	0.98	3.09	0.68	0.88	0.96	2.69	**0.61**	**0.74**	**1.03**	3.08
	C-RoBERTa	**0.74**	**0.85**	**0.96**	1.14	**0.71**	**0.77**	**1.01**	0.82	0.65	0.85	0.89	1.02	0.59	0.71	0.99	1.03
	MLC	0.72	0.82	0.92	2.13	0.69	0.76	0.97	1.91	0.62	0.81	0.81	1.59	0.60	0.72	1.01	1.65
	PRknn	0.71	0.81	0.91	3.30	0.67	0.73	0.95	3.29	0.62	0.80	0.91	3.29	0.59	0.71	0.99	3.29
	Random	0.71	0.81	0.91	3.32	0.58	0.63	0.84	3.32	0.50	0.65	0.72	3.32	0.51	0.61	0.86	3.32
$m=100$	Oracle r_o	1.00	1.14	1.31	6.02	0.99	1.08	1.60	5.27	0.96	1.25	1.69	5.09	1.00	1.20	1.84	5.18
	$r_o(0.5)$	0.86	0.98	1.10	6.51	0.77	0.84	1.19	6.28	0.72	0.93	1.20	6.23	0.74	0.89	1.34	6.28
	LinearR	0.71	0.82	0.87	6.50	**0.69**	**0.75**	**0.95**	6.45	**0.68**	**0.88**	**1.05**	6.20	0.65	0.79	1.14	6.43
	MLPR	**0.76**	**0.87**	**0.94**	6.53	0.66	0.71	0.88	6.49	0.63	0.82	0.89	6.28	**0.66**	**0.80**	**1.14**	6.39
	C-RoBERTa	0.75	0.86	0.94	0.83	0.65	0.71	0.87	0.82	0.62	0.80	0.91	1.30	0.59	0.71	0.99	1.31
	MLC	0.74	0.85	0.92	5.97	0.66	0.72	0.89	4.09	0.61	0.80	0.89	3.82	0.63	0.75	1.08	3.88
	PRknn	0.68	0.78	0.85	6.62	0.56	0.61	0.76	6.62	0.59	0.77	0.95	6.62	0.58	0.70	1.01	6.61
	Random	0.71	0.82	0.89	6.64	0.55	0.59	0.77	6.64	0.47	0.61	0.71	6.64	0.49	0.59	0.84	6.64
$m=1000$	Oracle r_o	1.00	1.14	1.18	9.37	1.00	1.09	1.38	8.45	0.99	1.29	1.56	8.03	1.00	1.20	1.61	8.54
	$r_o(0.5)$	0.87	0.99	1.02	9.84	0.76	0.83	1.02	9.56	0.72	0.93	1.07	9.42	0.75	0.90	1.17	9.62
	LinearR	0.79	0.90	0.92	9.82	0.70	0.76	0.86	9.73	0.70	0.91	0.97	9.46	0.72	0.86	1.10	9.70
	MLPR	**0.83**	**0.94**	**0.97**	9.82	**0.74**	**0.80**	**0.92**	9.76	**0.71**	**0.92**	**0.93**	9.51	**0.72**	**0.87**	**1.13**	9.67
	C-RoBERTa	0.78	0.89	0.91	1.36	0.73	0.80	0.90	1.13	0.70	0.91	0.91	1.06	0.66	0.79	0.99	1.54
	MLC	0.80	0.91	0.92	9.75	0.64	0.70	0.73	6.34	0.61	0.79	0.73	7.77	0.61	0.74	0.92	6.20
	PRknn	0.62	0.71	0.72	9.95	0.53	0.58	0.65	9.94	0.56	0.73	0.79	9.94	0.54	0.65	0.83	9.93
	Random	0.73	0.84	0.85	9.97	0.53	0.57	0.65	9.97	0.44	0.58	0.57	9.97	0.49	0.59	0.74	9.97

Table 7: **The Results on Hard Level RouterEval (part2)**. See Section 5.1 for detials of various metrics. Red area and blue area highlights indicate the "Strong router" and "Existing router" mentioned in Section 5.2, respectively. The best results in existing router methods are highlighted with underlines and on bold. The values in the table are rounded to two decimal places.

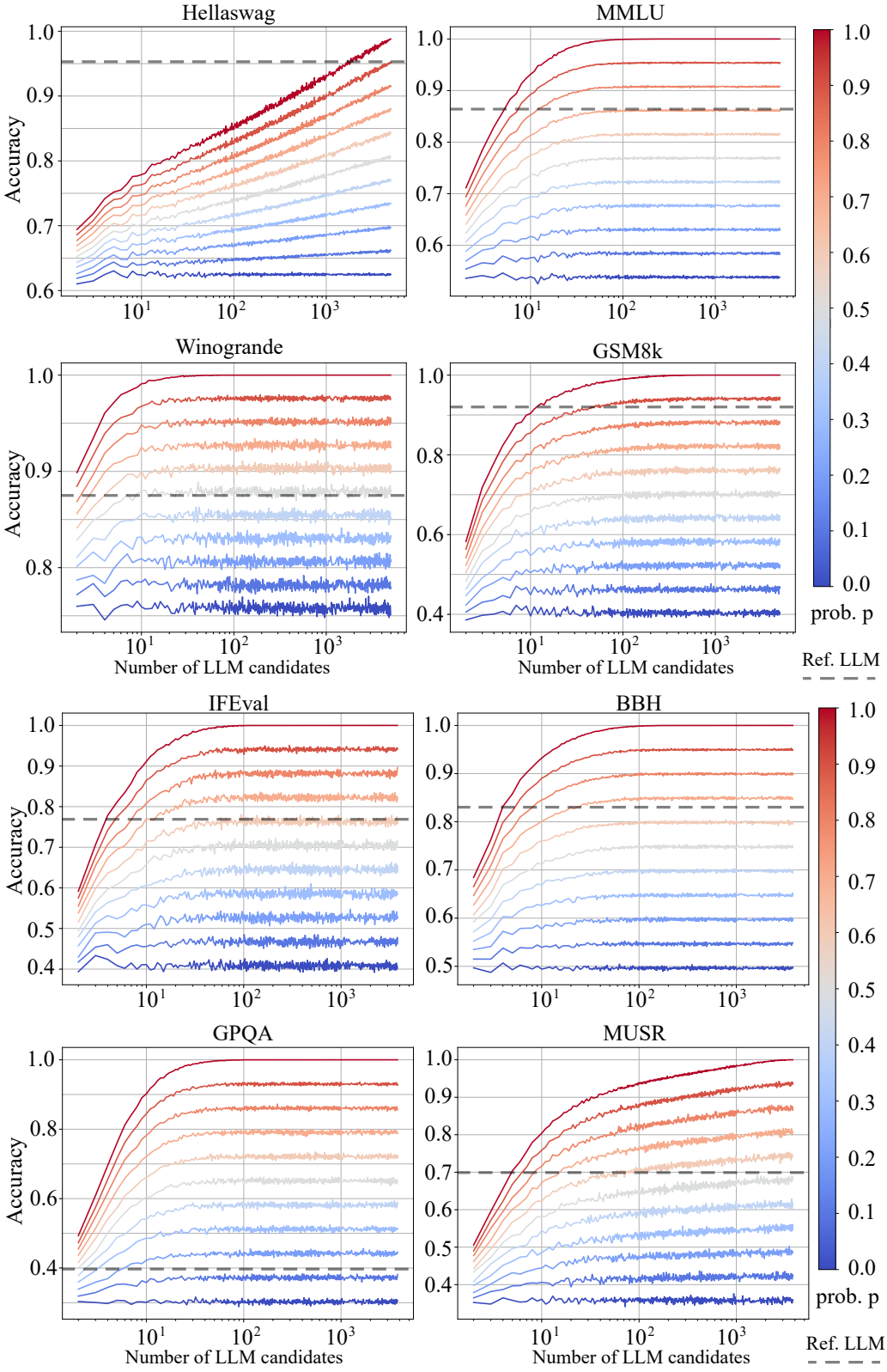

Figure 7: **The Model-level Scaling Up Phenomenon in Routing LLMs (part2)**. As shown in Section 3, the Prob. p indicates the performance of the router, with values closer to 1 representing greater similarity to the oracle router's capability. If $p \to 0$, then $r_o(p)$ degenerates into a random sampler. When the router $r_o(p)$ reaches a certain level of capability, it induces a scaling up phenomenon in the Routing LLMs paradigm. Specifically, as the number of LLM candidates increases, performance rapidly improves. "Ref. LLM" denotes a representative LLM with strong performance on given benchmark, such as GPT-4.