Harnessing the Power of Multiple Minds: Lessons Learned from LLM Routing

KV Aditya Srivatsa* **Kaushal Kumar Maurya*** **Ekaterina Kochmar**
Mohamed bin Zayed University of Artificial Intelligence, Abu Dhabi, UAE
`{vaibhav.kuchibhotla, kaushal.maurya, ekaterina.kochmar}@mbzuai.ac.ae`

Abstract

With the rapid development of LLMs, it is natural to ask how to harness their capabilities efficiently. In this paper, we explore whether it is feasible to direct each input query to a single most suitable LLM. To this end, we propose *LLM routing* for challenging reasoning tasks. Our extensive experiments suggest that such routing shows promise but is not feasible in all scenarios, so more robust approaches should be investigated to fill this gap.[1]

1 Introduction

Large language models (LLMs) demonstrate remarkable capabilities in many natural language generation and understanding tasks (Bommasani et al., 2021; Chang et al., 2023; Minaee et al., 2024). At the same time, Jiang et al. (2023) show that no single open-source LLM outperforms all others across different benchmarks and datasets, as various LLMs may exhibit different domain expertise (Beeching et al., 2023). Experiments towards predicting model behavior (Rabinovich et al., 2023; Srivatsa and Kochmar, 2024) also suggest that particular aspects of input prompts can affect different LLMs in different ways.

It is, therefore, reasonable to investigate whether the capabilities of different LLMs can be harnessed to achieve better results more efficiently. Recent findings suggest that performance can be improved with ensembling (Wang et al., 2022, 2023; Li et al., 2024) and collaborative frameworks (Wu et al., 2023b; Li et al., 2023). However, the research in this area is still in the early stages, with a number of open research questions. In this work, we propose *LLM routing*, which investigates whether *directing an input prompt to the most suitable single LLM can lead to better performance than what can be achieved with individual LLMs while maintaining a reasonable (e.g., single LLM) latency*.

With the rise of larger and more capable models in NLP and the wider field of ML, the research on sparse expert models has also extended. This class of models includes mixture-of-experts (Jacobs et al., 1991; Collobert et al., 2002; Eigen et al., 2013), switch-transformers (Fedus et al., 2022), and routing networks (Rosenbaum et al., 2017), among other models.[2] Approaches to building these sparse models vary along several dimensions: (i) how the optimal parameter subset(s) or model-pool candidates are identified for each input (e.g., feature-based or deep-encoder-based classification), (ii) whether the subset selection involves pre-training the candidate models or model layers (e.g., Mixtral (Jiang et al., 2024)), which can incur significant training compute and data costs, (iii) how many experts are selected for each input (e.g., HybridLLM (Ding et al., 2024) selects only the single best, whereas Shazeer et al. (2017) selects the top-k), and (iv) whether the approach also aims to improve the response quality or overall performance beyond that of any single candidate model. In this context, our paper aims to build and analyze a sparse LLM routing model that selects the single best LLM (from a pool of at least two LLMs) for each input query. The proposed router only requires fine-tuning of a relatively small pre-trained Transformer encoder model on the data without the need for pre-training or fine-tuning the LLMs.

Given that LLMs frequently face challenges with reasoning and planning tasks (Wei et al., 2022; Kojima et al., 2022), we focus on two well-established reasoning task benchmarks. We empirically investigate the feasibility of building *LLM routing* model capable of selecting the most suitable LLM for each input from a pool of diverse LLMs. The routing is grounded on responses generated by LLMs.

---

[*] Equal contribution

[1] Our code and data are available at `https://github.com/kvadityasrivatsa/llm-routing`.

[2] For more details on the related work, see Appendix F.

Proceedings of the Fifth Workshop on Insights from Negative Results in NLP, pages 124–134
June 20, 2024 ©2024 Association for Computational Linguistics

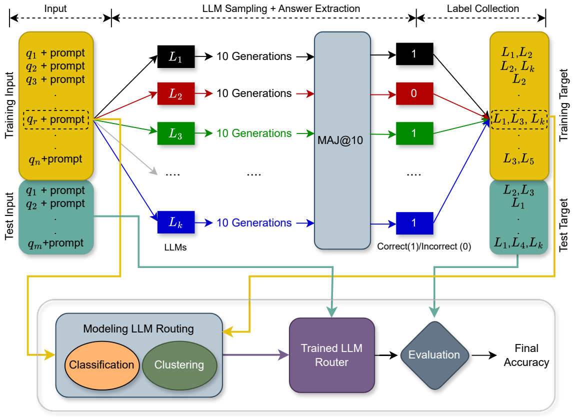


Figure 1: Overview of the proposed workflow.

We explore binary and multi-label classification modeling at the input query level, as well as a clustering approach based on the similarity between the queries. Finally, leveraging prediction confidence scores, we design multiple optimal policies to select a single suitable LLM from the pool.

The contributions and key findings of this work are as follows: (1) We propose an LLM routing model, which directs input queries to the most suitable *single* LLM. (2) We explore two different types of approaches for LLM routing, treating it as a classification and a clustering task. (3) We conduct experiments with 7 open-source LLMs and on two challenging reasoning benchmarks (`GSM8K` and `MMLU`). (4) We introduce theoretical upper bounds for two scenarios: (i) highest possible performance achieved jointly with all LLMs (i.e., oracle), and (ii) highest performance achieved with the proposed routing model. (5) Our findings indicate that theoretical upper bounds of the routing model are higher than individual model performance, however, the proposed model developed in practice is unable to achieve those scores. Specifically, the performance of the routing model is better than that of the weak LLMs but is similar to or slightly lower than that of the top-performing LLMs, which may be due to the small size of the training data.

Despite the somewhat negative results, we believe this study demonstrates the feasibility of modeling LLM routing and contributes to new research directions on efficient usage of LLMs, which can benefit researchers and practitioners.

2 Methodology

We present an overview of the proposed workflow in Figure 1. Below, we describe our approaches to *LLM sampling* and *LLM routing*.

Split/Criteria	GSM8K	MMLU
Training	6,816	13,757
Validation	359	285
Test	1,319	1,530
#examples for few-shot CoT	5	5

Table 1: Dataset statistics for the `GSM8K` and `MMLU` datasets. `MMLU` data splits are remapped to have a distribution similar to `GSM8K`. `CoT`: Chain-of-Thought

LLMs	Chat?	Specialized?	#Parameters
`llama2-7b`	×	×	7B
`llama2-13b-chat`		×	13B
`mistral-7b`	×	×	7B
`mistral-7b-it`		×	7B
`gemma-7b`	×	×	7B
`gemma-7b-it`		×	7B
`metamath-7b`	×		7B

Table 2: List of diverse LLMs selected in this study.

2.1 LLM Sampling

Selection of Benchmarks and LLMs As it has been observed that most of the existing LLMs struggle with reasoning tasks (Patel et al., 2021; Wu et al., 2023a), we focus on two challenging datasets associated with distinct domains – mathematical (`GSM8K` by Cobbe et al. (2021)) and natural language reasoning (`MMLU` by Hendrycks et al. (2021)). `GSM8K` consists of 8,792 diverse grade-school level math word problems (MWPs), while `MMLU` contains 15k multiple-choice questions spanning 57 subjects across STEM, humanities, and social sciences, among others (see Table 1). We have selected diverse LLMs based on criteria such as performance on benchmarks, training methodologies, model specialization, and more. The final set of LLMs is presented in Table 2.

Routing Data In this study, we assess each LLM's performance by generating 10 responses for each input query to ensure more reliable and replicable behavior in our modeling. For LLM prompting and answer extraction from responses, we have followed the standard guidelines (see Appendices B and C for details). Figure 2 presents the sample prompting templates. We use majority voting scores as labels for each input query to train routing classifiers. *Majority Voting* (MAJ@K $\in \{0, 1\}$) determines whether the most frequent answer matches the gold answer or not. The mean MAJ@10 scores across all input queries are reported in Table 3. Furthermore, to ensure a reliable response from an LLM, we consider only those LLMs for which the extracted answer viability scores are above 90% (please refer to Appendix B for more details), resulting in 6 viable LLMs for the GSM8K dataset and 7 for the MMLU dataset, respectively. We prepare the routing dataset by associating each input query with those viable LLM(s) that have a MAJ@10 score of 1. Formally, the target label for an input query $q \in Q$ is given by $label\,(q) = \{l \,|\, l \in L, maj@10\,(q, l) = 1\}$, where L is the set of candidate LLMs and Q is the set of query prompts from GSM8K or MMLU.

2.2 LLM Routing

Next, we build an LLM router, *determining which model to select from a pool of LLMs for a given input query based on performance and inference latency.* The ideal routing algorithm should select an optimal single LLM with high accuracy and low latency. To this end, we explore modeling at the individual query level using classification and utilize similarities among queries using clustering.

Classifier-Based Routing The classification-based routing consists of (1) the development of a classifier that can predict a set of LLMs capable of solving the input query along with prediction confidence scores, and (2) the identification of the policy to select optimal LLMs (with high accuracy and low latency) from the predicted LLMs based on confidence scores in the range [0-1].

Multi-label and Separate Classifiers: We have considered two types of classifiers: a multi-label classifier (MLC) and separate classifiers (SC). MLC aims to predict all LLMs apt for a given input query together in a single prediction step. The SC model, on the other hand, employs a separate binary classifier for each LLM and accumulates the results post hoc. Both types of classifiers are built on top of existing popular pre-trained language models (PLMs). Specifically, we experimented with BERT, DistilBERT, RoBERTa, and T5 models. Additionally, due to the small size of the training data, we explored smaller models, utilizing only a few layers of PLMs, as well as simpler models such as Random Forests. RoBERTa emerged as the best-performing model, and all results in this paper are reported with classifiers built by fine-tuning the RoBERTa PLM exclusively.

Proposed Policies: We utilize the classifiers' predicted confidence scores to design the following policies:

1. **ArgMax:** Select an LLM with the highest confidence score.
2. **Random:** Select a pool of LLMs with confidence above a certain threshold (i.e., 0.80) and randomly pick one LLM from the pool.
3. **Prediction:** Train a `RandomForest` regressor using training data confidence scores, where each input represents the confidence score for each predicted label, and the target is the confidence score of the first gold reference LLM. At test time, we select the LLM with a confidence score closest to the predicted score.
4. **Sorted Prediction (Sorted Pred):** Similarly to the 'Prediction' policy, the input confidence scores are arranged in ascending order based on LLMs' performance. This ensures that weaker LLMs have a fair opportunity.

Clustering-Based Routing Additionally, to incorporate the query-level similarities, we explore clustering for LLM routing as detailed below.

Learning Clusters: We fit a KMeans[3] clustering model on query-specific features extracted from the training data to learn discrete clusters. The features are extracted using: (1) TF-IDF vectorizer,[4] and (2) pooled hidden embedding of the RoBERTa[5] model's last layer.

Routing: For each cluster in the training set, the best performing LLM is determined. At inference, input queries in the test set are routed to the best-performing LLM for their corresponding cluster.

3 Experimental Setup

LLM Routing Baseline Models The following baseline models are included for comparison:

1. **Oracle:** The maximum possible performance is assumed under the premise that an oracle always selects a single LLM capable of solving each query if it is solvable.
2. **Random:** This represents the mean performance of randomly selecting an LLM uniformly for each input query across 1000 independent runs.
3. **Individual Models:** This is the mean performance of individual models with MAJ@10 across all queries.

---

[3] https://scikit-learn.org/stable/modules/generated/sklearn.cluster.KMeans.html

[4] https://scikit-learn.org/stable/modules/generated/sklearn.feature_extraction.text.TfidfVectorizer.html

[5] https://huggingface.co/FacebookAI/roberta-base

Models		GSM8K		MMLU	
		Acc	Lat (sec)	Acc	Lat (sec)
Oracle		87.18	3.46	89.15	1.89
Random		55.37	3.52	52.50	2.35
`gemma-7b`		71.11	7.10	63.85	3.00
`metamath-7b`		67.55	4.70	42.28	2.40
`mistral-7b`		59.74	3.70	62.09	1.80
`*mistral-7b-it`		50.41	1.00	51.63	1.10
`llama2-13b-chat`		46.70	1.80	50.52	4.80
`*gemma-7b-it`		36.84	0.70	49.28	1.00
`llama2-7b`		–	–	48.36	2.30
All LLMs		74.37	19.00	60.39	16.40
MLC	Upper bound	79.68	5.16	77.18	1.94
	ArgMax policy	67.62	4.76	62.28	2.95
	Random policy	67.47	4.76	58.16	2.86
	Prediction policy	**67.70**	4.77	**63.85**	2.95
	Sorted Pred policy	59.90	4.77	48.36	2.92
SC	ArgMax policy	67.55	4.70	62.87	2.94
Clustering	TF-IDF	67.55	4.70	61.76	2.83
	RoBERTa	67.55	4.70	61.76	2.83

Table 3: Performance of different routing models on `GSM8K` and `MMLU` test sets. For all queries, we have considered 10 generations with each LLM. Acc: mean accuracy with MAJ@10 (%), Lat: LLM inference latency in seconds per query (10 generations for each query), MLC: multi-label classifier, and SC: separate classifiers. * The term '`it`' indicates instruction-tuned LLMs. The highest individual-LLM accuracy is underlined, and the highest classifier accuracy is in **bold** for each dataset.

4. **All LLMs:** This baseline reports the mean accuracy of MAJ@($10\times|L|$) based on the combined pool of 10 generations from each LLM, where $|L|$ is the total number of LLMs.

Classifier Upper Bound This is similar to the oracle model, where the upper bound is calculated with predicted labels instead of gold labels.

4 Results and Discussion

In Table 3, we present the performance of each individual LLM across both datasets, alongside the performance of baselines and routing models. We observe that, even though `gemma-7b` outperforms other LLMs, there are diverse performance trends for other LLMs across datasets, with some performing better on `GSM8K`, and others on `MMLU`. To investigate the results further, we pose and address a number of research questions.

Does including multiple LLMs solve all questions in a given dataset? The Oracle model's Acc scores for both datasets are lower than 90%, indicating that more than 10% of questions cannot be solved by all LLMs combined. For details, see Figure 3 in the Appendix, where we project the distribution of questions solved by each of the LLMs.

How effective is a routing model when randomly picking LLMs? As expected, the random baseline model achieves the lowest Acc score for both datasets. This highlights the necessity for an effective routing model to navigate through LLMs.

Is the joint performance of multiple LLMs better than that of individual LLMs? Considering extreme cases like top-k and bottom-k LLMs as shown in Appendix Tables 5 and 6, we find that multiple LLMs collectively outperform single LLMs in terms of Acc. Even the joint performance with the bottom-2 model is better than that of individual models, underscoring LLMs' diverse problem-solving capabilities. However, we note two limitations in joint modeling: (i) the joint performance with all LLMs may not always be the best (see *All LLMs* baseline Acc scores), as reported for the `MMLU` dataset, and (ii) joint modeling drastically increases inference latency costs (i.e., Lat), aligning with recent research (Li et al., 2024). In contrast, the proposed LLM routing aims to leverage joint LLM capabilities while minimizing latency by selecting the single best-suited LLM.

Can the upper bound performance of the classifier/clustering be equal to the Oracle model performance? This is possible in an ideal scenario where classifier/clustering routing algorithms are perfect and bias-free. However, in our case, the training data for the algorithms is small (∼9k in `GSM8K` and 15k in `MMLU`), which leads to suboptimal performance. Still, the multi-label classifier's upper bound (Acc) has achieved a higher score than any individual LLMs, which is also close to the Oracle model performance. We hypothesize that more training data for classification/clustering may bridge this gap.

Does router modeling with multi-label classifiers exhibit better performance than individual LLMs? Unfortunately, the proposed multi-label classifier with different confidence-based policies does not lead to better performance (i.e., Acc) than some individual LLMs. This may be due to the small training data for the classifier. However, it can be observed that the classifier's performance is better than most of the weak-performing LLMs and close to the top-performing LLM. This suggests that LLM routing is a promising direction that requires better classifier modeling.

What is the impact of different policies on LLM router modeling? We have proposed four policies based on the label confidence scores of the multi-label classifier. The best policy can push the model performance closer to the upper bound performance of the multi-label classifier. However, we observe that due to the imperfect classifier (which yields weighted F1 scores of 0.71 for `GSM8K` and 0.67 for `MMLU`), the predictions (and confidence scores) are skewed towards only a few labels (see Figure 4 in the Appendix) which leads to sub-optimal ACC score. The predictions-based policy is better than other policies; however, the classifier performance presents a serious bottleneck. We conclude that larger training data and the development of a better classifier are essential for improving the ACC scores. Small sizes of both `GSM8K` and `MMLU` datasets prevent further investigation of this question.

How does a separate classifier compare to a multi-label classifier for LLM routing? With relatively small and imbalanced training sets, separate classifiers for each LLM are more prone to over-fitting. Despite attempts to address this with measures like early stopping and weighted class-based loss, most individual models usually converge to the overall best performers such as `gemma-7b` on test split. Ultimately, with the argmax policy in place, the separate classifier-based routing model's performance is similar to that of the argmax policy of the multi-label classifier.

How does clustering-based LLM routing compare to other models? The cluster-level routing approach aims to select the best LLM for a group of similar query prompts. It assumes that the relative performance of LLMs for each cluster remains consistent between the training and test sets. We find that this assumption does not hold for many clusters (39 out of 50 for `GSM8K` and 28 out of 50 for `MMLU`). In general, the best-performing LLM for most clusters in the training set is the same as the best LLM overall. The impact of different feature extraction methods (TF-IDF vs. `RoBERTa`) was minimal, resulting in a similar performance to the MLC+ArgMax model.

What is the impact of LLM routing on inference latency? Table 3 provides the inference latencies for all LLMs, baselines, and LLM routing models in seconds per query, recorded using a single Nvidia A100 GPU. Ideally, the best routing policies should maximize model accuracy (while maintaining at least same-level latency) or minimize overall latency (with the best LLM accuracy maintained). For instance, the MLC+ArgMax latency is lower than the corresponding highest individual model latency (of `gemma-7b`) for `GSM8K`. However, as the routing classifiers overfit to the best LLMs on the training sets (`metamath-7b` for `GSM8K` and `gemma-7b` for `MMLU`), the overall latency, much like mean accuracy, differs very slightly from that of the best LLMs. These findings validate our claim that the proposed *LLM routing* model consistently maintains a latency score equal to or lower than any individual LLM.

Ablations with multi-label routing: In appendix Figure 5, we overview ablation tests for LLM routing using a multi-label classifier trained with best- and worst-performing LLMs across both datasets. Key insights include: (1) Increasing the number of top-performing LLMs improves oracle scores but has marginal effects on the classifier's upper bound or argmax policy. (2) Increasing the number of worse-performing LLMs results in higher scores across oracle, MLC's upper bound, and MLC+ArgMax policy model, highlighting the effectiveness of LLM routing.

5 Conclusions and Future Directions

This study investigates the feasibility of *LLM routing*, i.e., navigating input queries by efficiently selecting the most suitable single LLM from a pool of LLMs. Through extensive experimentation with multi-label and separate classifiers, as well as clustering across two challenging benchmarks, we conclude that (i) there are theoretical bounds that can be achieved with LLM routing that are much higher than individual models' performance, and (ii) routing LLMs is a feasible direction that works best with equally capable LLMs. However, if a few LLMs dominate, the router's performance degrades, even though it still outperforms weak LLMs. At the same time, the inference latency of the routing model is at least at the same level as that of single LLMs.

With these findings in mind, we envision future research to investigate the following directions: (1) collecting larger datasets for LLM routing design; (2) developing novel models for LLM routing to accommodate LLMs with diverse capabilities; (3) designing better routing policies with confidence scores; (4) incorporating LLM-specific features for improved modeling; and (5) scaling up using more diverse LLMs and benchmarks.

Limitations

One of the key limitations of the proposed routing model is the limited training data available for training different algorithms with varying policies, which can result in biased learning despite taking a number of precautionary measures. Another limitation is the extraction of answers from generated responses: despite utilizing our best answer extraction algorithm, we could only extract viable answers for 83% to 95% of queries (with different LLMs). For the remaining queries, the answers extracted with our algorithm may be invalid or incorrect. Next, the proposed model works well with equally capable LLMs but is not yet effective enough for LLMs that have very different capabilities.

Finally, although the inference latency of the proposed model is comparable to that of the most suitable single LLM, frequent switches between the LLMs (based on the input queries) necessitate loading most of the LLMs into memory, posing a limited memory issue. This issue is also observed with different emerging LLMs (Jiang et al., 2023, 2024) similarly to our case. At the same time, the problem of limited memory in the context of LLMs has been well studied (Alizadeh et al., 2023; Eliseev and Mazur, 2023), and the solutions developed are directly (or with minor adjustments) applicable to our modeling, thereby ensuring the practical usability of the proposed model. We leave investigation of such approaches to future work.

Ethics Statement

This paper introduces router modeling to effectively harness the power of LLMs with different capabilities. As the proposed routing models use LLMs, we must acknowledge that, independently of this research, there are certain risks that pertain to all LLMs, as such models may generate outputs that, although plausible, are factually incorrect or nonsensical. Such *hallucinations* can misguide decision-making and propagate biases, especially in critical scenarios where accuracy is vital. Without proper safeguards, widespread LLM adoption could worsen these concerns. Thus, it is essential to develop mechanisms to mitigate hallucination risks, ensuring responsible and beneficial deployment of these powerful models before adopting the proposed routing model.

Acknowledgments

We thank Jad Doughman and Ted Briscoe for insightful discussions about this research. We are grateful to the Campus Super Computing Center at MBZUAI for supporting this work. We also thank the anonymous reviewers for their valuable feedback.

References

Keivan Alizadeh, Iman Mirzadeh, Dmitry Belenko, Karen Khatamifard, Minsik Cho, Carlo C Del Mundo, Mohammad Rastegari, and Mehrdad Farajtabar. 2023. LLM in a flash: Efficient Large Language Model Inference with Limited Memory. *arXiv preprint arXiv:2312.11514*.

Edward Beeching, Clémentine Fourrier, Nathan Habib, Sheon Han, Nathan Lambert, Nazneen Rajani, Omar Sanseviero, Lewis Tunstall, and Thomas Wolf. 2023. Open LLM Leaderboard.

Christopher M Bishop. 2006. *Pattern Recognition and Machine Learning*. Springer.

Rishi Bommasani, Drew A Hudson, Ehsan Adeli, Russ Altman, Simran Arora, Sydney von Arx, Michael S Bernstein, Jeannette Bohg, Antoine Bosselut, Emma Brunskill, et al. 2021. On the opportunities and risks of foundation models. *arXiv preprint arXiv:2108.07258*.

Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. 2020. Language Models are Few-Shot Learners. *Advances in neural information processing systems*, 33:1877–1901.

Yupeng Chang, Xu Wang, Jindong Wang, Yuan Wu, Linyi Yang, Kaijie Zhu, Hao Chen, Xiaoyuan Yi, Cunxiang Wang, Yidong Wang, et al. 2023. A survey on evaluation of large language models. *ACM Transactions on Intelligent Systems and Technology*.

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*.

Ronan Collobert, Yoshua Bengio, and Samy Bengio. 2002. Scaling Large Learning Problems with Hard Parallel Mixtures. In *Pattern Recognition with Support Vector Machines*, pages 8–23, Berlin, Heidelberg. Springer Berlin Heidelberg.

Dujian Ding, Ankur Mallick, Chi Wang, Robert Sim, Subhabrata Mukherjee, Victor Rühle, Laks V. S. Lakshmanan, and Ahmed Hassan Awadallah. 2024. Hybrid LLM: Cost-Efficient and Quality-Aware Query Routing. In *The Twelfth International Conference on Learning Representations*.

David Eigen, Marc'Aurelio Ranzato, and Ilya Sutskever. 2013. Learning Factored Representations in a Deep Mixture of Experts. *arXiv preprint arXiv:1312.4314*.

Artyom Eliseev and Denis Mazur. 2023. Fast inference of mixture-of-experts language models with offloading. *arXiv preprint arXiv:2312.17238*.

William Fedus, Barret Zoph, and Noam Shazeer. 2022. Switch Transformers: Scaling to Trillion Parameter Models with Simple and Efficient Sparsity. *Journal of Machine Learning Research*, 23(120):1–39.

Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. 2021. Measuring Massive Multitask Language Understanding. *Proceedings of the International Conference on Learning Representations (ICLR)*.

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.

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, Gianna Lengyel, Guillaume Bour, Guillaume Lample, Lélio Renard Lavaud, Lucile Saulnier, Marie-Anne Lachaux, Pierre Stock, Sandeep Subramanian, Sophia Yang, Szymon Antoniak, Teven Le Scao, Théophile Gervet, Thibaut Lavril, Thomas Wang, Timothée Lacroix, and William El Sayed. 2024. Mixtral of Experts. *arXiv preprint arXiv:2401.04088*.

Dongfu Jiang, Xiang Ren, and Bill Yuchen Lin. 2023. 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.

Mandar Joshi, Eunsol Choi, Daniel Weld, and Luke Zettlemoyer. 2017. TriviaQA: A Large Scale Distantly Supervised Challenge Dataset for Reading Comprehension. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1601–1611, Vancouver, Canada. Association for Computational Linguistics.

Takeshi Kojima, Shixiang Shane Gu, Machel Reid, Yutaka Matsuo, and Yusuke Iwasawa. 2022. Large language models are zero-shot reasoners. *Advances in neural information processing systems*, 35:22199–22213.

Guohao Li, Hasan Abed Al Kader Hammoud, Hani Itani, Dmitrii Khizbullin, and Bernard Ghanem. 2023. Camel: Communicative agents for "mind" exploration of large scale language model society. *arXiv preprint arXiv:2303.17760*.

Junyou Li, Qin Zhang, Yangbin Yu, Qiang Fu, and Deheng Ye. 2024. More Agents Is All You Need. *arXiv preprint arXiv:2402.05120*.

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.

Shervin Minaee, Tomas Mikolov, Narjes Nikzad, Meysam Chenaghlu, Richard Socher, Xavier Amatriain, and Jianfeng Gao. 2024. Large language models: A survey. *arXiv preprint arXiv:2402.06196*.

Arkil Patel, Satwik Bhattamishra, and Navin Goyal. 2021. Are NLP Models really able to Solve Simple Math Word Problems? In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 2080–2094, Online. Association for Computational Linguistics.

Ella Rabinovich, Samuel Ackerman, Orna Raz, Eitan Farchi, and Ateret Anaby-Tavor. 2023. Predicting Question-Answering Performance of Large Language Models through Semantic Consistency. *arXiv preprint arXiv:2311.01152*.

Sebastian Raschka. 2020. Model Evaluation, Model Selection, and Algorithm Selection in Machine Learning. *arXiv preprint arXiv:1811.12808*.

Mathieu Ravaut, Shafiq Joty, and Nancy Chen. 2022. SummaReranker: A Multi-Task Mixture-of-Experts Re-ranking Framework for Abstractive Summarization. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 4504–4524, Dublin, Ireland.

Clemens Rosenbaum, Tim Klinger, and Matthew Riemer. 2017. Routing Networks: Adaptive Selection of Non-linear Functions for Multi-Task Learning. *arXiv preprint arXiv:1711.01239*.

Pranab Sahoo, Ayush Kumar Singh, Sriparna Saha, Vinija Jain, Samrat Mondal, and Aman Chadha. 2024. A Systematic Survey of Prompt Engineering in Large Language Models: Techniques and Applications. *arXiv preprint arXiv:2402.07927*.

Melanie Sclar, Yejin Choi, Yulia Tsvetkov, and Alane Suhr. 2023. Quantifying Language Models' Sensitivity to Spurious Features in Prompt Design or: How I learned to start worrying about prompt formatting. *arXiv preprint arXiv:2310.11324*.

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. *arXiv preprint arXiv:1701.06538*.

Tal Shnitzer, Anthony Ou, Mírian Silva, Kate Soule, Yuekai Sun, Justin Solomon, Neil Thompson, and Mikhail Yurochkin. 2023. Large Language Model Routing with Benchmark Datasets. *arXiv preprint arXiv:2309.15789*.

KV Aditya Srivatsa and Ekaterina Kochmar. 2024. What makes math word problems challenging for llms? *arXiv preprint arXiv:2403.11369*.

Derek Tam, Anisha Mascarenhas, Shiyue Zhang, Sarah Kwan, Mohit Bansal, and Colin Raffel. 2023. Evaluating the Factual Consistency of Large Language Models Through News Summarization. In *Findings of the Association for Computational Linguistics: ACL 2023*, pages 5220–5255, Toronto, Canada. Association for Computational Linguistics.

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

Xuezhi Wang, Jason Wei, Dale Schuurmans, Quoc Le, Ed Chi, Sharan Narang, Aakanksha Chowdhery, and Denny Zhou. 2022. Self-consistency improves chain of thought reasoning in language models. *Advances in neural information processing systems*, 35:22199–22213.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Fei Xia, Ed Chi, Quoc V Le, Denny Zhou, et al. 2022. Chain-of-thought prompting elicits reasoning in large language models. *Advances in neural information processing systems*, 35:24824–24837.

Dingjun Wu, Jing Zhang, and Xinmei Huang. 2023a. Chain of Thought Prompting Elicits Knowledge Augmentation. In *Findings of the Association for Computational Linguistics: ACL 2023*, pages 6519–6534, Toronto, Canada. Association for Computational Linguistics.

Qingyun Wu, Gagan Bansal, Jieyu Zhang, Yiran Wu, Shaokun Zhang, Erkang Zhu, Beibin Li, Li Jiang, Xiaoyun Zhang, and Chi Wang. 2023b. AutoGen: Enabling Next-Gen LLM Applications via Multi-Agent Conversation. *arXiv preprint arXiv:2308.08155*.

Wenxuan Zhang, Yue Deng, Bing Liu, Sinno Jialin Pan, and Lidong Bing. 2023. Sentiment Analysis in the Era of Large Language Models: A Reality Check. *arXiv preprint arXiv:2305.15005*.

A LLM Inference Latency

Prompt Type	**LLM**	**GSM8K**	**MMLU**
FCoT	`llama2-7b`	4.21	2.30
	`gemma-7b`	7.10	3.00
	`mistral-7b`	3.70	1.80
	`metamath-7b`	4.70	2.40
ZCoT	`gemma-7b-it`	0.70	1.00
	`llama2-13b-chat`	1.80	4.80
	`mistral-7b-it`	1.00	1.10

Table 4: Statistics on the inference latency (i.e., runtime in seconds) for various LLMs over 10 generations for each input query. The timings were recorded using a single Nvidia A100 GPU. `FCoT` denotes few-shot Chain-of-Thought, and `ZCoT` denotes zero-shot CoT. We have considered 5 examples for `FCoT` prompting.

B Prompting for LLM Sampling

The consideration of diverse LLMs and datasets contributed to the challenges in prompting, as there is no single uniform prompting approach across LLMs and datasets (Sclar et al., 2023). Considering recent findings about the appropriate usage of prompts (Sahoo et al., 2024) and those from our own experimentation, we have converged on the following prompting decisions:

- For non-chat LLMs, few-shot Chain-of-Thought (CoT; Wei et al. (2022)) prompting works better than zero-shot (Kojima et al., 2022) for both datasets. We used 5 few-shot examples. The few-shot prompting leads to over 95% *viable* answers (except for `llama2-7b` LLM, which has the viability score of 83%) in generated solutions. A *viable* answer is a single numeric/alphabetic answer that can be extracted from the generated solution using extraction algorithms (see Appendix C) to compare with the reference answer. The viable answer can be correct or incorrect.

- For chat LLMs, few-shot CoT distracts the generation, which leads to unexpected outputs. The zero-shot CoT works best. We utilize different models' chat-templates from Hugging Face[6] to ensure correctness. The viability of answer extraction for chat models is 92%.

The sample zero-shot and few-shot CoT prompt templates are presented in Figure 2.

---

[6] https://huggingface.co/docs/transformers/en/chat_templating

C Answer Extraction from LLM Responses

The adapted prompting approaches used in our LLM queries are designed to instruct LLMs to specify their final answers at the very end of each of their responses. We thus use a simple answer extraction policy of selecting the last mentioned numerical value (for `GSM8K`) and multiple-choice option (for `MMLU`) from the generated responses. Responses failing to report any final answer are regarded as invalid and counted as incorrect answers. For `MMLU`, we evaluate the extracted options directly against the annotated correct options (among 'A', 'B', 'C', and 'D') in the dataset. For `GSM8K`, questions where the absolute difference between the ground truth and predicted numerical answers are less than $\epsilon = 0.1$ are considered to be solved correctly. This threshold was set to accommodate instances where model-generated real-valued answers differ slightly from the expected answer.

Lessons Learned: It is observed that sometimes the expected answer is present in one of the last sentences of the response instead of at the very end. We extracted all such answers as well. Allowing a 0.1 absolute error difference leads to more accurate answers.

D Implementation, Hyperparameters, and Hardware Details

Querying LLMs We use the vLLM package[7] to query LLMs. All models were queried with a temperature of 0.8 and a max token length of 2000. Each question prompt was queried 10 times with different initialization seeds. We used a single Nvidia A100 GPU for all runs. Querying each dataset once took approximately 1-2 hours.

Training Routing Classifiers We use the HuggingFace library[8] for loading and tuning all pretrained Transformer encoders in our experiments. Each model was trained for 10 epochs, with an initial learning rate of 2e-5, warmup ratio of 0.1, and class-balanced CrossEntropy loss. The training checkpoint with the lowest validation loss was selected for inference.

E Detailed Results for Routing Models

See Figures 3-5 and Tables 5-6.

Models	**ACC (%)**	**LAT (sec)**
Oracle	87.18	3.46
Random	55.37	3.52
`gemma-7b`	71.11	7.10
`metamath-7b`	67.55	4.70
`mistral-7b`	59.74	3.70
`mistral-7b-it`	50.41	1.00
`llama2-13b-chat`	46.70	1.80
`gemma-7b-it`	36.84	0.70
top-2 LLMs	81.80	11.80
top-3 LLMs	84.00	15.5
top-4 LLMs	85.82	16.5
top-5 LLMs	86.03	18.3
bottom-2 LLMs	55.64	2.50
bottom-3 LLMs	67.02	3.50
bottom-4 LLMs	75.51	7.20
bottom-5 LLMs	79.91	11.90
All LLMs	74.37	19.00
Upper Bound of MLC	79.68	5.16
MLC + Argmax policy	67.62	4.76
MLC + Random policy	67.47	4.76
MLC + Prediction policy	67.70	4.77
MLC + Sorted Pred policy	59.90	4.77
SC + Argmax policy	67.55	4.70
Clustering + TF-IDF	67.55	4.70
Clustering + RoBERTa	67.55	4.70

Table 5: Performance of different routing models on `GSM8K` data. ACC: mean accuracy with MAJ@10 (%), LAT: LLM inference latency in seconds per query (10 generations for each query), MLC: multi-label classifier, SC: separate classifiers, and top-k: best k performing models. All other notation is the same as for Table 3.

F Related Work

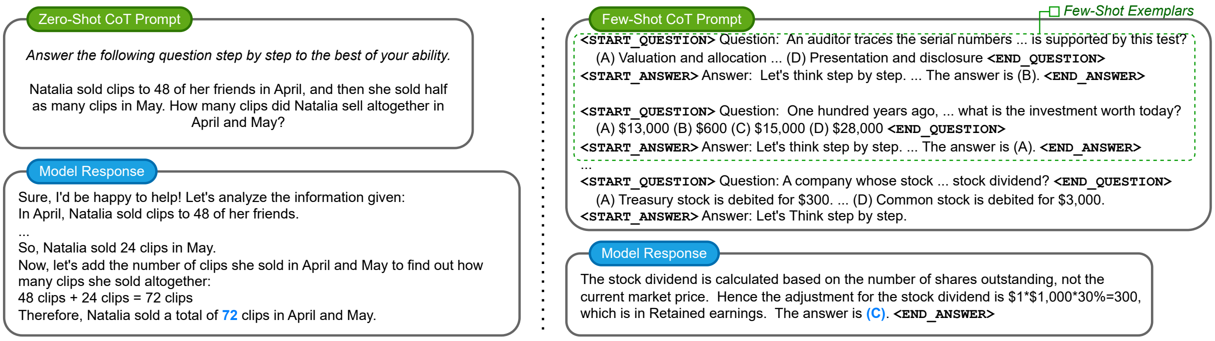
Model Diversity Several surveys (Bommasani et al., 2021; Minaee et al., 2024, inter alia) suggest that LLMs can develop emergent capabilities. Specifically, this suggests that models can show behavior and demonstrate skills beyond explicitly constructed ones. By virtue of differing training data, models may exhibit a wide variety of domain expertise. Jiang et al. (2023) demonstrates that no single open-source LLM outperforms other models across popular benchmarks. This further motivates the need to develop ensembling or routing methods aimed at improving the combined performance of a pool of LLMs with a diverse range of abilities.

Model Selection A fundamental step in routing queries within an ensemble of models is to estimate the extent of overlap between the capabilities of the LLMs in the candidate pool with those deemed necessary to resolve an input query. Model selection in the context of LLM routing greatly differs from its traditional form in ML (Bishop, 2006; Raschka, 2020), wherein the training and test datasets are similar in distribution. Training data

---

[7] https://github.com/vllm-project/vllm

[8] https://huggingface.co/



Answer the following question step by step to the best of your ability.

Natalia sold clips to 48 of her friends in April, and then she sold half as many clips in May. How many clips did Natalia sell altogether in April and May?

Sure, I'd be happy to help! Let's analyze the information given:
In April, Natalia sold clips to 48 of her friends.
...
So, Natalia sold 24 clips in May.
Now, let's add the number of clips she sold in April and May to find out how many clips she sold altogether:
48 clips + 24 clips = 72 clips
Therefore, Natalia sold a total of 72 clips in April and May.

<START_QUESTION> Question: An auditor traces the serial numbers ... is supported by this test?
(A) Valuation and allocation ... (D) Presentation and disclosure **<END_QUESTION>**
<START_ANSWER> Answer: Let's think step by step. ... The answer is (B). **<END_ANSWER>**

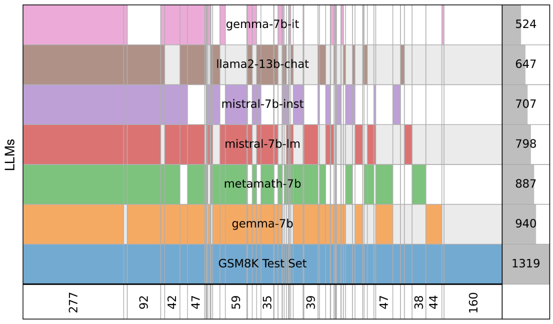
<START_QUESTION> Question: One hundred years ago, ... what is the investment worth today?
(A) $13,000 (B) $600 (C) $15,000 (D) $28,000 **<END_QUESTION>**
<START_ANSWER> Answer: Let's think step by step. ... The answer is (A). **<END_ANSWER>**
...
<START_QUESTION> Question: A company whose stock ... stock dividend? **<END_QUESTION>**
(A) Treasury stock is debited for $300. ... (D) Common stock is debited for $3,000.
<START_ANSWER> Answer: Let's Think step by step.

The stock dividend is calculated based on the number of shares outstanding, not the current market price. Hence the adjustment for the stock dividend is $1*$1,000*30%=300, which is in Retained earnings. The answer is (C). **<END_ANSWER>**

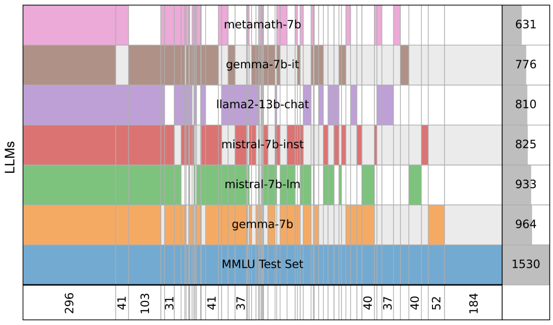
Zero-Shot CoT prompt for GSM8K

Few-Shot CoT prompt for MMLU

Figure 2: Sample zero-shot Chain-of-Thought (CoT) prompt template for a chat (or instruction-tuned) LLM and few-shot Chain-of-Thought (CoT) prompt template for a standard LLM.



(a) GSM8K



(b) MMLU

Figure 3: Distribution of queries from the `GSM8K` and `MMLU` test sets solved (score 1 with MAJ@10) by each LLM. The counts at the bottom of each figure denote the number of questions in each chunk, and those on the right denote the total number of questions solved by each LLM.

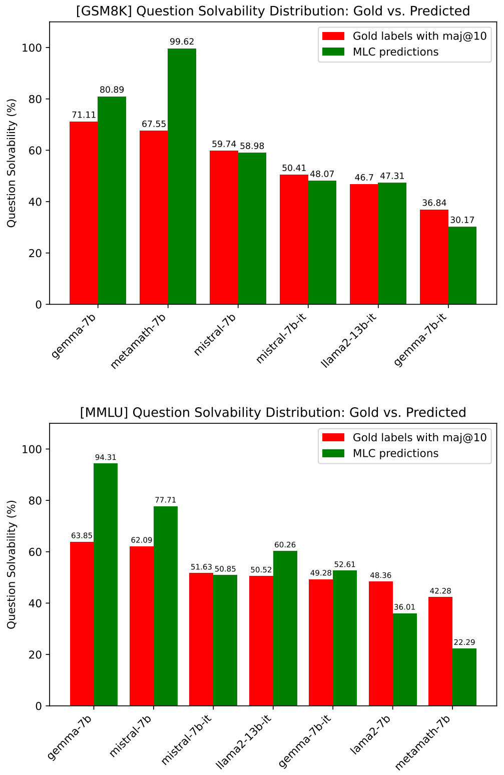


Figure 4: LLMs "solvability" distribution. The gold label scores are obtained with MAJ@10, and prediction label scores are obtained with a multi-label classifier.

for LLMs include massive corpora spanning trillions of tokens with relatively straightforward learning objectives like next-token prediction (Brown et al., 2020). Test data, on the other hand, often involves highly structured tasks like reasoning and question answering (Hendrycks et al., 2021; Cobbe et al., 2021; Joshi et al., 2017), summarization (Tam et al., 2023), and classification (Zhang et al., 2023), which may not be very prevalent in corresponding training data. This makes gauging the pain points of resolving a complex query non-trivial. Furthermore, studies like Rabinovich et al. (2023) and Srivatsa and Kochmar (2024) suggest that certain aspects of the prompt phrasing, i.e., its length and readability, significantly impact LLMs' ability to tackle the underlying tasks.

LLM Ensembling Previous attempts at ensembling and routing of LLMs aim to tackle one of two tasks: **(1)** Opting between LLM generations to select the best response. Liu and Liu (2021); Ravaut

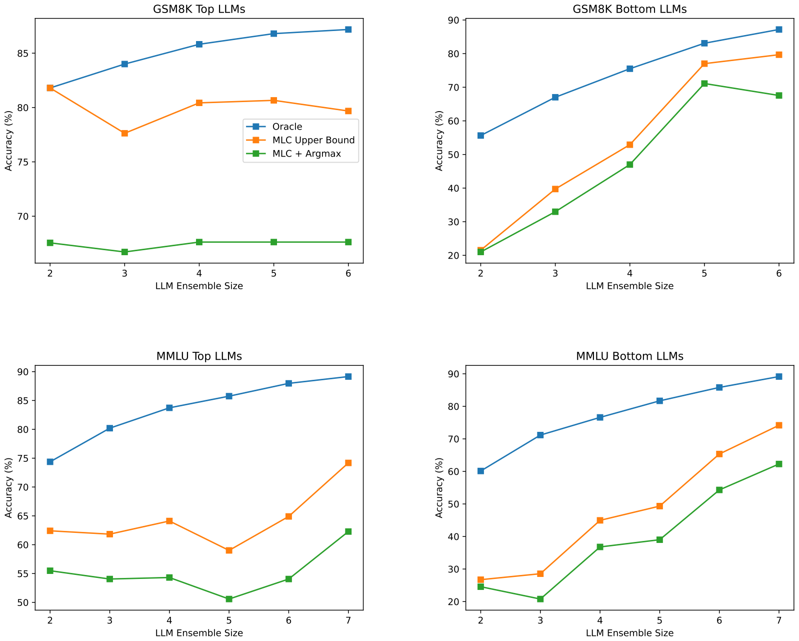


Figure 5: Different ablation configurations for LLMs for `GSM8K` and `MMLU` datasets.

Models	**Acc** (%)	**Lat** (sec)
Oracle	89.15	1.89
Random	52.50	2.35
`gemma-7b`	63.85	3.00
`mistral-7b`	62.09	1.80
`mistral-7b-it`	51.63	1.10
`llama2-13b-chat`	50.52	4.80
`gemma-7b-it`	49.28	1.00
`llama2-7b`	48.36	2.30
`metamath-7b`	42.28	2.40
top-2 LLMs	73.47	4.80
top-3 LLMs	79.54	5.90
top-4 LLMs	83.72	10.70
top-5 LLMs	85.75	11.70
top-6 LLMs	87.88	14.0
bottom-2 LLMs	60.13	4.70
bottom-3 LLMs	71.17	5.70
bottom-4 LLMs	78.10	10.50
bottom-5 LLMs	81.69	11.60
bottom-6 LLMs	83.11	13.40
All LLMs	60.39	16.40
Upper Bound of MLC	77.18	1.94
MLC + Argmax policy	62.28	2.95
MLC + Random policy	58.16	2.86
MLC + Prediction policy	63.85	2.95
MLC + Sorted Pred policy	48.36	2.92
SC + Argmax policy	62.87	2.94
Clustering + TF-IDF	61.76	2.83
Clustering + RoBERTa	61.76	2.83

Table 6: Performance of different routing models on the `MMLU` data. Acc: mean accuracy with MAJ@10 (%), Lat: LLM inference latency in seconds per query (10 generations for each query), MLC: multi-label classifier, SC: separate classifiers, and top-k: best k performing models. All other notation is the same as for Table 3.

et al. (2022); Jiang et al. (2023) train models to rank or classify the most suitable response for a given query. However, this requires querying all LLMs in the model pool for each query during inference time. This can become computationally expensive with a large number of LLMs in the candidate pool. **(2)** Building routing networks (Rosenbaum et al., 2017) that utilize only a subset of parameters of a model or a subset of experts from a pool of candidate models. For example, Jiang et al. (2024) employ a Mixture-of-Experts (MoE) (Jacobs et al., 1991; Collobert et al., 2002; Eigen et al., 2013) model with 8 experts, wherein only 2 experts are accessed at each model layer to produce the next token. This, however, requires pre-training the model weights, which incurs large computing and data costs. Alternatively, HybridLLM (Ding et al., 2024), Shazeer et al. (2017), and Shnitzer et al. (2023) train separate classifiers which select the best LLM(s) for each input query.

This paper aims to create and study a sparse routing network for selecting the best LLM from a pool of more than two LLMs for each example. The routing network only needs to tune an extra Transformer-based classifier without needing to pre-train or fine-tune the LLMs. Furthermore, we also incorporate the former task by measuring the response quality (through accuracy) and determining if it can outperform the individual experts (LLMs) in the pool.