Capability Instruction Tuning: A New Paradigm for Dynamic LLM Routing

Yi-Kai Zhang, De-Chuan Zhan, Han-Jia Ye[*]

School of Artificial Intelligence, Nanjing University
National Key Laboratory for Novel Software Technology, Nanjing University
{zhangyk, zhandc, yehj}@lamda.nju.edu.cn

Abstract

Large Language Models (LLMs) have demonstrated human-like instruction-following abilities, particularly those exceeding 100 billion parameters. The combined capability of some smaller, resource-friendly LLMs can address most of the instructions that larger LLMs excel at. In this work, we explore how to route the best-performing LLM for each instruction to achieve better overall performance. We develop a new paradigm, constructing *capability instructions* with model capability representation, user instruction, and performance inquiry prompts to assess the performance. To learn from capability instructions, we introduce a new end-to-end framework called Model Selection with Aptitude Test (MODEL-SAT), which generates positive and negative samples based on what different models perform well or struggle with. MODEL-SAT uses a model capability encoder that extends its model representation to a lightweight LLM. Our experiments show that MODEL-SAT understands the performance dimensions of candidate models and provides the probabilities of their capability to handle various instructions. Additionally, during deployment, a new model can quickly infer its aptitude test results across 50 tasks, each with 20 shots. MODEL-SAT performs state-of-the-art model routing without candidate inference and in real-world new model-released scenarios. The code is available at https://github.com/Now-Join-Us/Model-SAT

Introduction

Large Language Models (LLMs) (OpenAI 2022; Du et al. 2022; Touvron et al. 2023a; Chiang et al. 2023; Jiang et al. 2023) rapidly evolve, demonstrating near-human general capabilities, especially in understanding, reasoning, and creative tasks related to instruction-response scenarios. Recent advancements have even enabled these LLMs to be trained in multilingual (Yang et al. 2024; Dubey et al. 2024), multidomain (Zhou et al. 2024; Yang et al. 2024), and multimodal (Chen et al. 2015, 2023; Reid et al. 2024) environments, allowing them to tackle complex instructions such as "What is the relationship between Fourier series and Hilbert space?" or to interpret images by identifying, "What are the basis vectors of the Hilbert space?"

The rise of LLMs and their extensions has incredibly energized community applications. However, achieving more comprehensive capabilities often requires LLMs of a larger scale. According to the Open LLM Leaderboard (Aidar Myrzakhan 2024), 60% of the top 50 LLMs have around 70 billion (B) parameters or more, with only three LLMs under 10B. Additionally, some closed-source LLMs consistently dominate performance rankings over extended periods. Consequently, optimizing LLM applications often hinges on substantial computational resources or costly token purchases. A natural idea arises: Can we utilize multiple smaller LLMs, which are more resource-friendly and have below one-tenth of the parameters of their larger counterparts, to achieve performance comparable to gigantic LLMs while maintaining low inference costs?

In the experiments, we find that the combined capability of

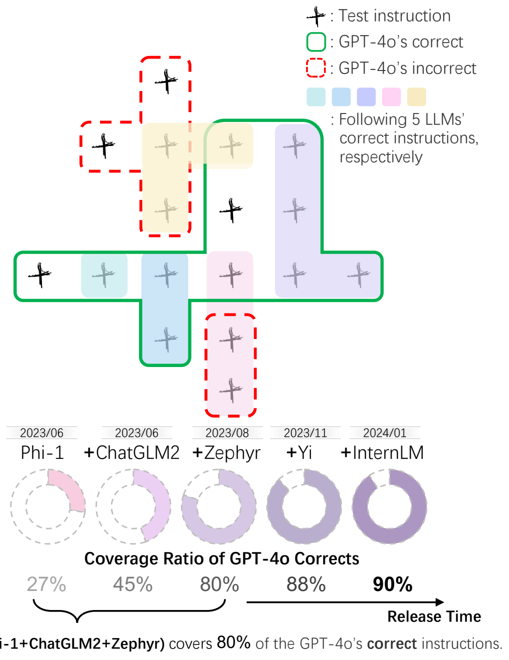

: Test instruction
: GPT-4o's correct
: GPT-4o's incorrect
: Following 5 LLMs' correct instructions, respectively
2023/06
Phi-1
2023/06
+ChatGLM2
2023/08
+Zephyr
2023/11
+Yi
2024/01
+InternLM
Coverage Ratio of GPT-4o Corrects
27%
45%
80%
88%
90%
Release Time
(Phi-1+ChatGLM2+Zephyr) covers 80% of the GPT-4o's correct instructions.

Figure 1: **Illustration of Coverage Observation**: The combined capabilities of the earlier-released model zoo effectively address most of the instructions that GPT-4o excels at. The union of samples managed accurately by Phi-1, ChatGLM2, and Zephyr covers 80% of GPT-4o's correct instructions. The smaller-scale model zoo can enhance overall performance by selecting a suitable model for each instruction.

[*]Corresponding author.

some smaller-scale LLMs, despite their lower overall performance, can address most of the instructions that larger LLMs excel at. As shown in Figure 1, on the Massive Multitask Language Understanding (MMLU) (Hendrycks et al. 2020) benchmark, the Phi-1 LLM with 1.3B performs nearly 50% worse than GPT-4o. However, it exhibits similar effectiveness to GPT-4o in the `high school mathematics` category. Moreover, we create an early-access LLM zoo that includes Phi-1 (Gunasekar et al. 2023) and four 7B LLMs, which were released a year earlier than GPT-4o and exhibit an approximately 30% performance gap compared to GPT-4o. However, the combined accurate responses from this zoo cover 90% of which GPT-4o handles correctly and address nearly 80% with which GPT-4o struggles. By strategically assigning instructions to the suitable LLM in the zoo, there is potential to exceed GPT-4o's performance by 15%. From this phenomenon, the model routing for each instruction enhances performance with seamless LLM transitions and minimal inference costs, all without user awareness.

The key to the proposed instruction-level model routing is to efficiently identify the optimal model from a vast pool of options, without prior access to the potential candidates' inference outputs (Tan et al. 2023; Xiao et al. 2023) or the target task's ground truth (You et al. 2022; Pándy et al. 2022). In this paper, we introduce MODEL-SAT: **Model S**election with **A**ptitude **T**est. Our approach leverages 50 core 20-shot tasks, where the model test result represents model capability. By learning the generalization relationships between the capability representations of the candidate models and the instructions to be assigned, we can select the most suitable model across various repositories and target instructions.

Driven by the model capability representation, the MODEL-SAT framework establishes a novel paradigm, denoted as capability instruction tuning. Capability instructions consist of a capability representation, a user instruction, and a prompt to probe whether the model can perform that instruction. Using extensive historical performance data, capability instruction tuning learns an implicit relationship between core capability representations and unseen instructions. Moreover, it delves deeper into understanding the mapping between the capabilities' performance and the instructions' semantic distribution. This intuition comes from the observation that individuals who perform well in the mathematical sections of the college admission SAT in the United States often pursue careers that involve logical reasoning. Capability instruction tuning aims to equip the model with a lightweight standardized guide to assess its effectiveness in handling future instructions.

Specifically, we combine model capability representation with positive and negative training instructions regarding current model performance, yielding statements like, "The model achieves accuracy 85% on the task of 'Mathematics, Geometry, ...'. Instruction: ..., Predict whether the model can handle ...". To align the performance distribution inherent in model representation to the instruction semantic, we are the first to incorporate a capability encoder and extend the input of a lightweight LLM to include capability representation. The end-to-end MODEL-SAT functions as a model router that outputs the probabilities indicating which models will likely excel at specific instructions.

Additionally, we establish several comprehensive benchmarks for model routing of LLMs and their extensions. Our benchmarks cover a range of model zoos, such as (**1**) smaller-scale, weaker ones, (**2**) mixed-scale options, and (**3**) high-performance larger-scale LLMs. Furthermore, we expand the model routing to include multimodal LLM-instruction settings. MODEL-SAT achieve significantly improved overall performance across model zoos without incurring any inference overhead, comparable to the performance levels of larger-scale LLMs. Notably, the capability instruction tuning maintains the model representation generalization to unseen data. The new LLM can quickly develop effective model representations after just a few inferences (only on 50 x 20-shot tasks). In light of practical routing scenarios with the emergence of new-version LLMs, we establish 60 incremental routing scenarios that impose higher routing speed and overhead requirements. Throughout these settings, MODEL-SAT consistently demonstrates superior performance.

In summary, our contributions are:

- **A novel paradigm: capability instruction tuning**, where model representation with efficient aptitude tests and instructions create capability instructions for high-performance-driven instruction-level model routing.
- **MODEL-SAT framework**, features a model capability encoder and a lightweight LLM to end-to-end learn the router via various model capability representations.
- **Comprehensive model routing benchmarks for LLMs and their extensions**, covering five LLM zoo setups with multimodal scenarios, as well as simulating 60 incremental-released model routers to ensure quick adaptation to unseen data and new LLMs.
- **An open-source, deployable model routing toolkit** that applies model routing techniques to any model zoo, enhancing performance while remaining unaware of users with acceptable routing delays.

Preliminary

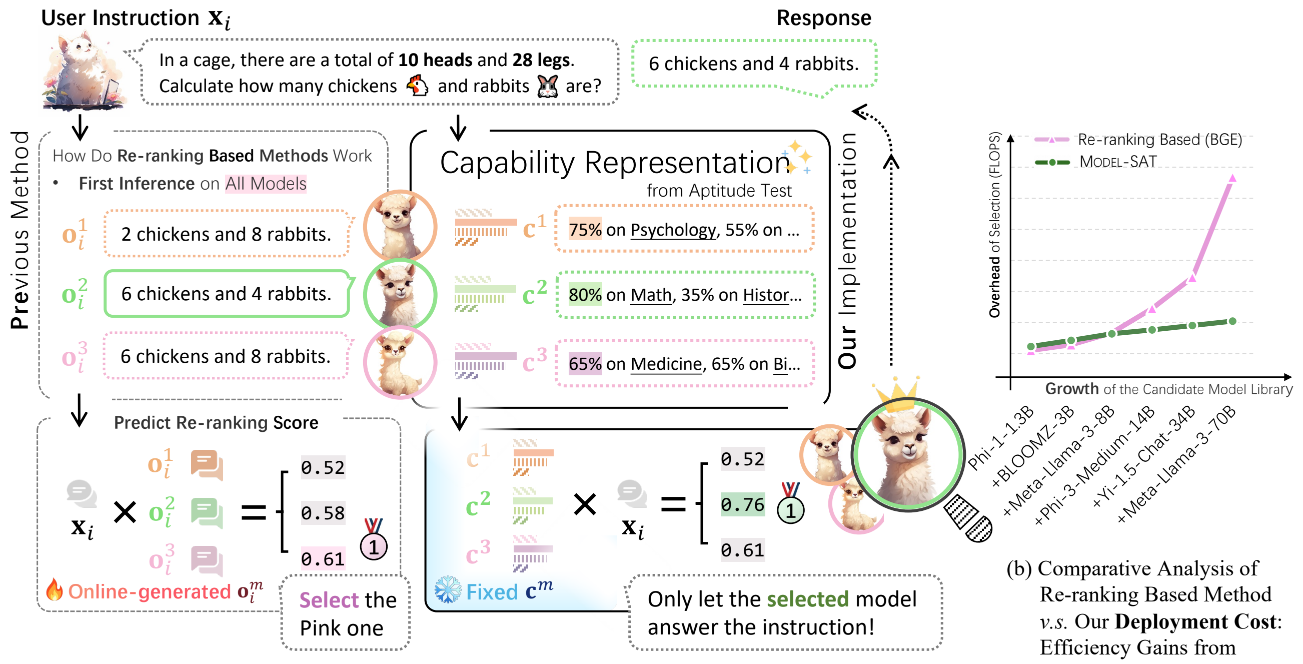
We begin by discussing the key elements and the pipeline of model selection, followed by the evolution of related works.

Instruction, Output, and Answer

Consider a test instruction dataset $\mathcal{D}_{\text{test}} = \{(\mathbf{x}_i, \mathbf{a}_i)\}_{i=1}^{N}$ with N labeled samples. The $\mathbf{x}_i$ and $\mathbf{a}_i$ represent the instruction and its corresponding answer, respectively. Given an LLM or its extension, represented as f, the output generated for instruction $\mathbf{x}_i$ is denoted as $\mathbf{o}_i$, *i.e.*, $f(\mathbf{x}_i) = \mathbf{o}_i$. There are no restrictions on the language, domain, or modality of $\mathbf{x}_i$; In this paper, we focus on decoder-only text generation models, which means that $\mathbf{a}_i$ is typically presented in text form. For the model f to excel at instruction $\mathbf{x}_i$, it is equivalent to obtaining a high score on the evaluation $\text{eval}(\mathbf{o}_i, \mathbf{a}_i)$.

Pipeline of Model Routing

Consider a candidate model zoo composed of many trained LLMs, $\mathcal{M} = \{f^m\}_{m=1}^{M}$. Model routing involves selecting a model from the zoo for each instruction $\mathbf{x}_i$ in the test dataset $\mathcal{D}_{\text{test}}$. Specifically, the sequence of selected models is formalized as $\boldsymbol{f} = (f_1, f_2, \ldots, f_N)$, where $f_i \in \mathcal{M}$. We define


User Instruction $\mathbf{x}_i$
In a cage, there are a total of 10 heads and 28 legs. Calculate how many chickens and rabbits are?
Response
6 chickens and 4 rabbits.
Previous Method
How Do Re-ranking Based Methods Work
First Inference on All Models
$\mathbf{o}_i^1$ 2 chickens and 8 rabbits.
$\mathbf{o}_i^2$ 6 chickens and 4 rabbits.
$\mathbf{o}_i^3$ 6 chickens and 8 rabbits.
Predict Re-ranking Score
$\mathbf{x}_i$ × $\mathbf{o}_i^1$ $\mathbf{o}_i^2$ $\mathbf{o}_i^3$ = 0.52 0.58 0.61
Online-generated $\mathbf{o}_i^m$
Select the Pink one
Capability Representation
from Aptitude Test
$\mathbf{c}^1$ 75% on Psychology, 55% on ...
$\mathbf{c}^2$ 80% on Math, 35% on Histor...
$\mathbf{c}^3$ 65% on Medicine, 65% on Bi...
Our Implementation
$\mathbf{c}^1$ $\mathbf{c}^2$ $\mathbf{c}^3$ × $\mathbf{x}_i$ = 0.52 0.76 0.61
Fixed $\mathbf{c}^m$
Only let the selected model answer the instruction!
Overhead of Selection (FLOPS)
Re-ranking Based (BGE)
MODEL-SAT
Growth of the Candidate Model Library
Phi-1-1.3B
+BLOOMZ-3B
+Meta-Llama-3-8B
+Phi-3-Medium-14B
+Yi-1.5-Chat-34B
+Meta-Llama-3-70B

(a) Evaluation Comparison: Re-ranking Based Method *v.s.* **Model S**election with **A**ptitude **T**est.

(b) Comparative Analysis of Re-ranking Based Method *v.s.* Our **Deployment Cost**: Efficiency Gains from **Capability Representation**.

Figure 2: **Illustration of Model Routing with Capability Instructions: A Comparison with Re-ranking Based Methods.** The goal of model router is to select the optimal model for a given user instruction without access to ground truth and enhance overall performance. Previous re-ranking methods require inference for each candidate. MODEL-SAT employs a lightweight aptitude test to create capability representations. It learns the intrinsic relationship between model representations and the instructions to be assigned, significantly speeding up model routing and streamlining deployment.

the optimal model $\hat{f}$ for instruction $\mathbf{x}_i$ as the model that maximizes the score: $\text{eval}\left(\hat{f}(\mathbf{x}_i), \mathbf{a}_i\right)$. The objective of the instruction-level model routing is:

$$\hat{\boldsymbol{f}} = \left(\underset{f^m \in \mathcal{M}}{\arg\min}\, \ell\left(f^m\left(\mathbf{x}_i\right),\, \mathbf{a}_i\right)\right)_{i=1}^{N}, \tag{1}$$

where $\ell\left(\cdot\right)$ represents the loss function associated with the metrics between $\mathbf{o}_i^m = f^m(\mathbf{x}_i)$ and the ground truth $\mathbf{a}_i$. The model routing bottleneck arises from the number of instructions on which no model in the zoo performs well.

Revisit from Requirement, Target, and Key Inputs

Routing target of *parameter initialization* or *models with zero-shot capabilities*: Early model router (Tran, Nguyen, and Hassner 2019; Nguyen et al. 2020; Tan, Li, and Huang 2021; Ding et al. 2022) efforts primarily focus on identifying a good training initialization that facilitates fine-tuning downstream tasks to achieve optimal performance. In this context, candidate models likely required additional training to adapt to the target task. Recently, guided by scaling laws, foundational models like LLMs have experienced remarkable advancements in their zero-shot capabilities (Touvron et al. 2023b; Wei et al. 2022; Team et al. 2023). Extended models have demonstrated considerable potential in multilingual, multi-domain, and multimodal applications. For instance, Llama 3.1 (Dubey et al. 2024) serves as a multilingual agent, Qwen2-Math (Yang et al. 2024) tackles several Olympiad-level problems, and GPT-4o (OpenAI 2023) processes information from multiple sources.

Routing requirements with *target instruction annotation*, *backpropagation delay*, or *candidate output*: Some works (Bao et al. 2019; Li et al. 2021; You et al. 2021; Deshpande et al. 2021; Pándy et al. 2022) design the proxy metric of transferability, which approximates the lower bound of fine-tuned performance. These works often rely on certain source clues, labeled instructions, or backpropagation steps to assess the transferability from the source pre-trained model to the target dataset. Additionally, some re-ranking-based works (Tan et al. 2023; Xiao et al. 2023; Zhang et al. 2024a) train an extra model to learn the contrastive relationships between the instruction and the candidate inference outputs $\{\mathbf{o}_i^m\}_{m=1}^M$, routing the optimal one linked to model f^m. However, obtaining all inferences may introduce significant delays when the number of models M in the repository becomes excessively large (Shnitzer et al. 2023; Lu et al. 2023; Hu et al. 2024). Our MODEL-SAT aims to route models without annotation or inference requirements, considering candidates as black boxes. A central feature is constructing model representations for each model and learning the adjusted relationship between it and the target instructions.

Key input – *model representation* for model routing: When routing a model for instruction, the router requires the key representation that captures the model's characteristics. We followed the concept of learnware (Zhou 2016), leveraging a small amount of model-proficient data to con-

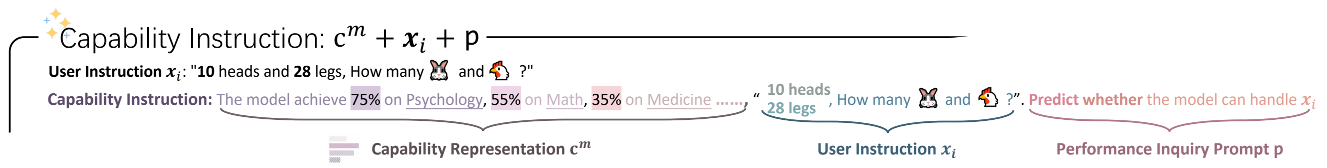

Capability Instruction: $c^m + x_i + p$
User Instruction x_i: "10 heads and 28 legs, How many and ?"
Capability Instruction: The model achieve 75% on Psychology, 55% on Math, 35% on Medicine ......, "10 heads 28 legs, How many and ?". Predict whether the model can handle x_i
Capability Representation c^m
User Instruction x_i
Performance Inquiry Prompt p

Figure 3: **One example of a Capability Instruction.** It is an instruction for model routing that inquires whether a model can handle a specific user instruction. It comprises three components: the capability representation c^m based on the streamlined aptitude test, the user instruction $\mathbf{x}_i$ to be assigned, and a performance inquiry prompt p. This instruction is inputted into the MODEL-SAT Capability LLM, which outputs the probability that the model can perform the user instruction well.

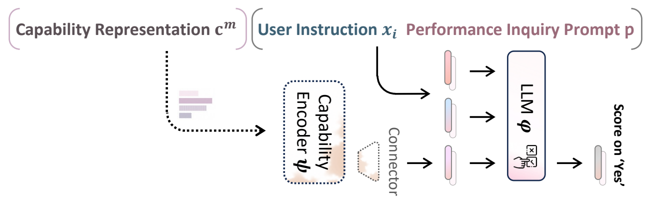

Capability Representation c^m
User Instruction x_i
Performance Inquiry Prompt p
Capability Encoder ψ
Connector
LLM φ
Score on 'Yes'

Figure 4: The Architecture of MODEL-SAT.

struct shared specifications (Zhou and Tan 2024; Tan et al. 2024). Other relevant methods leverage forward behavior or results on target as model representation, which inevitably introduces inference delays. Recently, some approaches (Lu et al. 2024; Srivatsa, Maurya, and Kochmar 2024; Ding et al. 2024; Feng, Shen, and You 2024) have started to utilize learnable parameters as model representations. For instance, some introduce a surrogate scorer as the corresponding model representation, learning the mapping from the task to the accuracy of candidate model outputs. Model Spider (Zhang et al. 2023b) takes this concept by encoding the model representation into a learnable vector, which acts as the input token for a Transformer-based router. However, learnable representation face challenges when new models are introduced, as they require extensive historical performance for costly pre-training of the router. Our solution uses text-only descriptions of capabilities. New models can create representations by inferring 50 quick tasks, each with 20 shots.

MODEL-SAT: Model Routing with Aptitude Test

In this section, we start by building the model representation and progress to the details of MODEL-SAT, training data, and optimization process. Finally, we outline an efficient deployment framework for model routing.

Capability Instructions

The *capability instruction* mainly comprises the capability representation of the candidate model f^m, user instruction $\mathbf{x}_i$, and performance inquiry prompt. Specifically, the model's capability representation is formed from 50 distinct tasks across various categories from the MMLU dataset, with each task being 20-shot. We provide a concise description of five keywords for each task. Next, we evaluate the candidate models across these 50 tasks and describe the results in natural language, *i.e.*, model representation. Furthermore, the advantage of representing in natural language is that it helps to include extra expert knowledge, such as mentioning *which languages a model supports*. The easy-to-obtain representations serve as an aptitude test for the models, indicating their potential capabilities across various dimensions.

To assess how well the candidate model can follow a single or a set of instructions $\mathbf{x}_i$, we introduce the training instructions that were executed correctly versus those incorrectly. These will be paired with the performance inquiry prompt p to form the *capability instruction*, denoted as $\mathbf{z}_i$, which drives the router to predict adaptation scores. As illustrated in Figure 3, it combines the capability representation c^m for candidate m, the instruction $\mathbf{x}_i$, and a inquiry prompt p.
Core task sampling: We sample instructions of core tasks with the highest distinguishability, avoiding those where most models perform correctly or incorrectly. In the model zoo of training, samples for which half of the models make mistakes while the other half are correct carry greater weight.

Architecture

Motivation: Although LLMs demonstrate strong instruction-following abilities, a gap exists between performance and the semantic distribution in *capability instructions*, particularly in understanding combinations of performance dimensions. For example, if a candidate model achieves 80% in mathematics and 95% in legal principles, the model may possess legal reasoning skills. To address this, we propose extending a capability encoder E5-Large ~0.5B ($\boldsymbol{\psi}$) before a Phi-3-Mini 3.8B LLM ($\boldsymbol{\varphi}$) to align the candidate performance with the instructions. The architecture is illustrated in Figure 4.

Structure: The capability instruction comprises the capability representation c^m for model m, the instruction $\mathbf{x}_i$, and the query prompt p. We first align the model representation, mapped by the capability encoder, into an embedded feature of LLM inputs. This is achieved using a single-layer MLP, which acts as a connector to adapt the dimensions. Consequently, we derive the aligned model capability vector:

$$\mathbf{e}_{\mathrm{c}^m} = \mathbf{W} \cdot \boldsymbol{\psi}\left(\mathrm{c}^m\right) , \tag{2}$$

where $\mathbf{e}_{\mathrm{c}^m}$ is combined with the input embeddings of $\mathbf{x}_i$ and p to form the capability instruction $\mathbf{z}_i$, *i.e.*,

$$\mathbf{z}_i = [\mathbf{e}_{\mathrm{c}^m}, \mathbf{e}_{\mathbf{x}_i}, \mathbf{e}_{\mathrm{p}}] = [\mathbf{e}_1, \mathbf{e}_2, \ldots, \mathbf{e}_s] , \tag{3}$$

where s denotes the length of the concatenated capability instruction sequence. Our alignment module operates at a natural language level, allowing for a streamlined design. In the following Section , we also explore alternative approaches, including training without the alignment module.

Tuning Recipe

Forward Process of the Prediction Score: As shown in Figure 3, the query prompt p in the capability instruction includes keywords related to positive terms that the model excels at. For example, "Yes" serves as the key response in the prompt "predict whether the model can handle test instruction by indicating 'Yes' or 'No'." In this context, the model routing prediction score is:

$$\Pr\left(\text{'Yes'} \mid \mathbf{z}_i\right) = \prod_{t=1}^{s} \mathbf{1}_{(|\mathrm{c}^m|, s]} \cdot \boldsymbol{\varphi}\left(\mathbf{e}_t \mid [\mathbf{e}_1, \cdots, \mathbf{e}_{t-1}]\right) , \tag{4}$$

where we omit the input embedding layer for the LLM $\boldsymbol{\varphi}$.

Positive and Negative Instructions for Training: We apply Homogeneous In-Batch Negative Sampling (Karpukhin et al. 2020; Zhang et al. 2023a) for each capability representation c^m with its well-performed and poorly-performed instructions to enhance the discriminative during training. Typically, a k-shot training batch $\mathbf{Z} = \{\mathbf{z}_i\}_{i=0}^{k}$ contains 1 positive instruction and $k-1$ negative ones.

Loss Design: We denote the position of the positive instruction in the training batch $\mathbf{Z}$ as y_{pos}, and the remaining ones are $k-1$ negative instructions. Our objective is to enhance the prediction score for the positive ones as the candidate performs better on this instruction. We employ the cross-entropy loss to optimize this in one batch $\mathbf{Z}$:

$$\mathcal{L}_{\text{CE}} = \mathbb{E}_{\mathbf{Z}\in\mathcal{D}_{\text{test}}}\left[-\log \Pr\left(\boldsymbol{h}_{\boldsymbol{\varphi},\text{'Yes'}}\left(\mathbf{Z}\right) = y_{\text{pos}} \mid \mathbf{Z}\right)\right] , \tag{5}$$

where $\boldsymbol{h}_{\boldsymbol{\varphi},\text{'Yes'}}\left(\mathbf{Z}\right) = \arg\max_{\mathbf{z}_i\in\mathbf{Z}} \Pr\left(\text{'Yes'} \mid \mathbf{z}_i\right)$ is the LLM $\boldsymbol{\varphi}$ to identify which instruction $\mathbf{z}_i \in \mathbf{Z}$ can be done well (positive) and which cannot (negative).

Learning Strategy: The model representation is derived from the capability distribution on MMLU. Similarly, we develop both in-domain and out-of-domain learning environments for MODEL-SAT. In the first stage, we collect in-domain positive and negative training instructions, primarily sourced from the same category as the MMLU dataset. We only fine-tune the connector between the capability encoder $\boldsymbol{\psi}$ and the LLM $\boldsymbol{\varphi}$, establishing an initial capability-to-instruction mapping. In the second stage, we fine-tune all model parameters. We apply a larger learning rate on the encoder and connector to enhance capability alignment with instruction semantics.

Data Refinement: We further address noise in whether the candidate model can accurately perform the instructions, influencing whether a capability instruction is a positive or negative training sample. For those difficult instructions that only a few models handle correctly, we implement a circle test by rotating the sequence of options to prevent lucky guesses. Furthermore, we prioritize higher-ranked candidates in the training data by sampling with increased weight.

Efficient Deployment

MODEL-SAT provides the routing prediction for the candidate model applied to the target instruction. These scores are generated by the same model, rendering them comparable. In this paper, we propose an open-source and comprehensive model routing toolkit, MODEL-SAT. This toolkit offers a viable solution for dynamic model routing within communities such as HuggingFace, harnessing the repository to boost performance on target tasks.

MODEL-SAT exhibits remarkable generalization capabilities for unseen data, which can be directly concatenated into the capability instruction. Similarly, the incremental extension to new models proves highly efficient, requiring only inference on 50 core tasks for the model representation. As later addressed in the experiments, MODEL-SAT exhibits zero-shot model routing abilities, facilitating the streamlined development of capability instructions in broader contexts.

Experiments

This section begins by detailing the construction of training and test instructions in *capability instructions tuning*. It then presents different zoo setups for testing and concludes with an analysis of results and ablation studies.

Training and Test Instructions

As mentioned earlier, the *capability instruction* consists of model representation c^m, instructions $\mathbf{x}_i$ to assign, and performance inquiry prompts p.

Candidate Model Representations c^m **for candidate** m: We introduced 66 open-source LLMs of varying scales. This includes 60 models under 10B, 15 ones between 10B and 20B, and 5 ones around 60B. We sample 50 categories from the MMLU dataset, with 20 distinguishing instructions from each. Different candidate models share core tasks to ensure stability in capability demonstration.

Instructions x_i **Pending to Assign**: We consider more than 20 datasets that include areas such as language, analysis, understanding, and reasoning in general evaluations, as well as specialized fields like mathematics and medicine. For each dataset, we sample sets of positive and negative instructions where the model performed well or poorly, with sampling on stronger models assigned greater weight. Each dataset contains about 100 instructions on average.

Performance Inquiry Prompts p: We explore different approaches for the probability of model routing. For *capability instructions*, we design the performance inquiry prompt, such as "predict whether the model can handle test instruction by indicating 'Yes' or 'No'." In this context, a response of 'Yes' signifies that the model is well-performed to the instruction. We also experiment with integrating a regression linear layer onto the next token embedding.

The capability instruction for the test $\mathbf{z}_i$ similarly consists of the model representation c^m, the target instruction $\mathbf{x}_i$ to be assigned, and the performance inquiry prompt p. To ensure test stability, we conduct a perturbation evaluation on model representation. Specifically, we randomly alter the ranking of the aptitude test results in capability representation twice and then calculate the average routing scores $\Pr\left(\text{'Yes'} \mid \mathbf{z}_i\right)$. The response on this instruction $\mathbf{x}_i$ is provided by the candidate model with the highest routing score.

Benchmarks of LLM Routing

In this section, we outline benchmarks with various LLMs and their extension zoos, featuring detailed settings.

Smaller-Scale LLM do Better: As demonstrated in the Table 1, the **smaller-scale zoo** contains InternLM2.5 (7.7B),

Method	#Params	In-Domain MMLU	In-Domain WinoG.	Out-of-Domain ARC-C	Out-of-Domain BoolQ	Out-of-Domain TruthfulQA	Out-of-Domain MRPC	Out-of-Domain MNLI	Mean
					Smaller-scale LLMs (<10B)				
InternLM2.5 (Cai et al. 2024)	7.7B	69.88	81.22	60.75	70.43	54.56	68.38	60.68	66.89
Meta-Llama-3 $_{\text{Instruct}}$ (Touvron et al. 2023b)	8.0B	65.59	75.45	62.12	76.76	51.63	68.38	55.82	65.62
Qwen2 $_{\text{Instruct}}$ (Yang et al. 2024)	7.6B	69.13	74.11	61.43	82.57	55.49	78.92	54.96	68.19
GLM-4 (Zeng et al. 2024)	9.4B	69.28	80.82	66.13	84.77	59.32	78.92	40.73	68.65
Phi-3 $_{\text{Small-128K}}$ (Abdin et al. 2024)	7.4B	75.90	77.11	71.08	86.70	64.62	75.98	46.82	71.38
Best-Performing among the five above	xB	75.90	81.22	71.08	86.70	64.62	78.92	60.68	74.16
					LLM Selection on *Smaller-scale* LLMs (5 models)				
Random Selection	xB	70.11	77.66	64.59	79.94	57.58	72.79	51.54	67.74
Cappy (Tan et al. 2023)	M·xB	69.53	78.06	63.99	81.10	57.77	74.75	53.17	68.34
BGE $_{\text{Large}}$ (Xiao et al. 2023)	(0.3+M)·xB	71.68	78.53	66.38	82.48	61.44	73.28	55.39	69.88
GTE $_{\text{Large}}$ (Zhang et al. 2024a)	(1.8+M)·xB	72.02	79.32	68.09	83.73	59.61	75.74	56.16	70.67
MODEL-SAT **(Ours)**	M·4.3B+xB	79.86	82.24	72.53	86.73	65.12	79.66	60.83	75.28
					Larger-Scale LLMs (10B~50B)				
Phi-3 $_{\text{Medium-128K}}$ (Abdin et al. 2024)	14B	76.63	74.35	66.49	86.30	54.54	78.92	59.42	70.95
Yi-1.5 $_{\text{Chat}}$ (Young et al. 2024)	34B	77.15	81.47	70.62	87.84	62.02	80.88	61.56	74.50
Meta-Llama-3 $_{\text{Instruct}}$ (Touvron et al. 2023b)	70B	79.89	82.62	71.67	93.61	61.83	83.58	65.07	76.90
Qwen2 $_{\text{Instruct}}$ (Yang et al. 2024)	72B	83.79	84.41	68.62	94.90	54.85	84.31	66.95	76.83
Mixtral-8x22B $_{\text{Instruct-v0.1}}$ (Jiang et al. 2024)	140B	77.63	85.25	72.68	92.71	68.19	81.13	67.70	77.90
					Capability Instruction Tuning w/o Inference Overhead				
Smaller-Scale LLM Zoo		79.86	82.24	72.53	86.73	65.12	79.66	60.83	75.28
Smaller-Mixed LLM Zoo		78.60	82.08	72.01	86.48	64.50	78.19	60.72	74.65
Ours Middle-Mixed LLM Zoo	M·4.3B+xB	79.97	83.03	72.69	87.80	64.87	83.82	61.80	76.28
Larger-Mixed LLM Zoo		84.16	86.27	73.21	93.94	69.16	85.54	67.92	80.03
High-Performance LLM Zoo		**85.64**	**87.85**	**73.63**	**95.02**	**69.40**	**88.24**	**68.39**	**81.17**

Table 1: **A Comprehensive Performance Evaluation**: Covering smaller-scale, high-performance giant LLMs, and a mixed LLM zoo of small, medium, and large levels. Model-SAT performs instruction-level model selection, consistently maintaining efficient and precise results that outperform the optimal one in the LLM zoo. Bold is the best, and underlined is the second-best.

Method	MMLU	ARC-C	TruthfulQA	Mean
Random Selection	70.11	64.59	57.58	64.09
Best-Perfoming	75.90	71.08	64.62	70.53
RoBERTa + MLP	71.25	64.33	59.12	64.90
+ k Nearest Neighbors	70.02	65.02	58.51	64.52
+ Random Forest	73.75	68.05	61.44	67.75
Phi-3 $_{\text{Mini-128K}}$	74.71	70.58	61.70	68.83
MODEL-SAT	**79.86**	**72.53**	**65.12**	**72.50**

Table 2: **Performance Comparisons of Other Learning Strategies for Capability Instructions**. The capability encoder of Model-SAT learns the mapping of performance to semantics, demonstrating strong model selection abilities.

Meta-Llama-3-Instruct (8.0B), Qwen2-Instruct (7.6B), GLM-4 (9.4B), and Phi-3-Small-128K (7.4B). The **smaller-mixed zoo** includes the smaller-scale zoo and Phi-1 (1.3B), BLOOMZ (3B), and Zephyr-Alpha (7.2B). These LLMs have fewer than 10B parameters and low deployment costs. In Figure 1, we show that the union of correct responses can cover a set of instructions that only larger-scale ones can manage.

General LLM Zoo Settings. 1) Middle-Mixed and Larger-Mixed LLM Zoo: The **middle-mixed zoo** includes the smaller-scale zoo and Phi-3-Medium-128K (14B), and Yi-1.5-Chat (34B). The **larger-mixed zoo** consists of the middle-mixed ones, Meta-Llama-3-Instruct (70B), Qwen2-Instruct (72B), and Mixtral-8x22B-Instruct-v0.1 (140B). The mixed zoo can validate the routing method across different capabilities. **2) High-Performance LLM Zoo**: We select from larger-scale LLMs to boost performance further. The model zoo contains only three models above with over 70B parameters. **3) Multimodal LLM Zoo**: To verify the generality of capability instruction tuning, we construct a multimodal LLM zoo that includes MiniCPM-Llama3-V 2.5, Phi-3-Vision-128k-Instruct, and InternLM-XComposer2-VL-7B.

Dataset	Phi-3 Vision	InternLM XC2-VL	MiniCPM Llama3-V	Random Selection	MODEL-SAT (Ours)
MMMU VAL	41.86	41.06	43.10	42.19	**43.21**
AI2D TEST	78.40	79.44	77.33	78.24	**80.38**
CCBench	37.60	56.62	**57.16**	50.49	56.96

Table 3: **Performance Comparisons in Selection of Multimodal LLM**. Model-SAT maintains excellent average performance on the above three popular evaluation benchmarks.

Instructions for Model Routing Evaluation: The test capability instructions differ from the training ones of model routers and consist of seven evaluation datasets. Datasets including MMLU (Hendrycks et al. 2021) (5-shot) and WinoGrande (Sakaguchi et al. 2020) (5-shot) cover a broad range and are involved in the training part as the in-domain

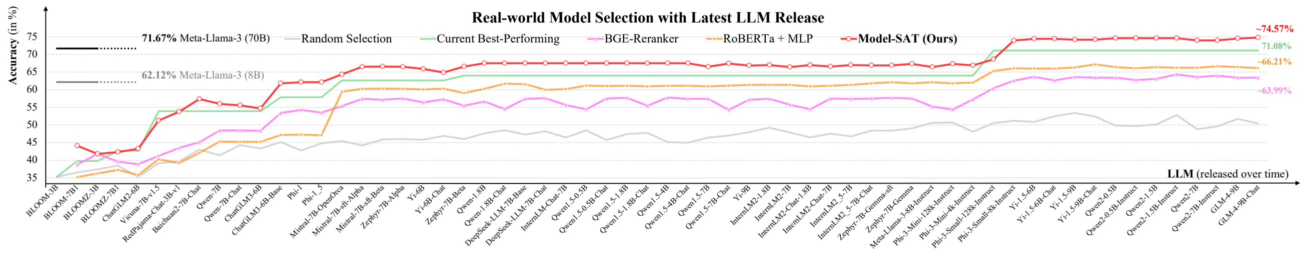


Figure 5: **Real-world Model Routing with Latest LLM Release** on ARC-Challenge. MODEL-SAT (in red) quickly generalizes to unseen LLMs without extra training, maintaining robust performance despite dynamically adding diverse LLMs.

evaluation. On the other hand, datasets such as ARC-Challenge (Bhakthavatsalam et al. 2021) (25-shot), TruthfulQA (6-shot), and BoolQ (Clark et al. 2019) (1-shot) with MRPC (1-shot) and MNLI (1-shot) in GLUE (Wang et al. 2019) benchmark focus on specific capabilities, serving as the unseen out-of-domain evaluation of the model routers. We consider the evaluation datasets MMMU-VAL (Hendrycks et al. 2020), AI2D-TEST (Kembhavi et al. 2016), and CCBench (Liu et al. 2024) in the multimodal scenario.

Real-world Model Routing with Unseen Datasets & Latest LLMs: **1**) In Table 1, *In-Domain* and *Out-of-Domain* indicate whether the dataset is included in the training set for LLM routing. **2**) In Figure 5, We design a novel model routing setting that, with the **release of 60 LLMs**, we update the existing zoo after each new model with the top 5 historically best and the latest one. With the continual increment of unseen LLMs, this dynamic environment tests whether methods can maintain a compelling performance.

Toward Comprehensive and Effective Routing

Performance Analysis in Various Model Zoos. Table 1 demonstrates that MODEL-SAT performs impressively across five comprehensive LLM Zoos. (a) Smaller-Scale: routing of LLMs under 10B achieve performance comparable to the ~70B LLMs. MODEL-SAT's average score of 75.28% closely matches Meta-Llama-3-Instruct-70B's 76.90%, and outperforms it on the ARC-C and TruthfulQA benchmarks. Furthermore, MODEL-SAT selects the optimal model for each instruction, surpassing the best-performing models in the Smaller-Scale Zoo. (b) Smaller-Mixed: We add three earlier released, weaker, and smaller LLMs. MODEL-SAT maintained stable performance, with a slight decrease of about 1% compared to row (a), while performance on ARC-C, BoolQ, and MNLI benchmarks remained nearly identical. (c) Middle-Mixed: Row (c) includes two medium-scale models (10B to 70B), resulting in improved performance for MODEL-SAT compared to row (a). Its average performance now closely matches that of the 70B model. (d) Larger-Mixed: Incorporating three 70B models in row (d) showed that MODEL-SAT remains robust despite significant performance variances in the LLM zoo, with improvements of nearly 5% on MMLU, BoolQ, and so on. (e) High-Performance: Row (e) features a routing of only three 70B models, revealing that the capabilities of gigantic LLMs are further unleashed, achieving a state-of-the-art score of 85.64% on MMLU and 73.63% on the ARC-Challenge.

Comparative Analysis of Routing Delay. In Table 1, the selected model parameter-scale is denoted as xB. The overhead associated with the re-ranking method is related to the M candidates of xB. Figure 2 illustrates that the re-ranking requires obtaining all inference results from the zoo first, while MODEL-SAT processes model representations all at once and utilizes them throughout its lifetime. As the scale of models in the zoo grows, MODEL-SAT's routing cost remains unaffected by inference ones, ensuring efficient model routing.

Comparison Ranking-based Methods: We evaluate re-ranking methods such as Cappy, BGE-Large, and GTE-Large. Although they have access to the outputs of each candidate, re-ranking often struggles to find optimal results, potentially because it primarily focuses on retrieving different semantics rather than optimizing performance across similar outputs.

Detailed Comparative Analysis: Furthermore, we explore various learning strategies within the capability instruction tuning framework. Using features extracted from RoBERTa, we train MLP (classification-based), k Nearest Neighbors (clustering-based), and Random Forest (tree-based). We observed that, except for tree models that are suitable for handling capability representation as similar tabular data, other learning strategies fail to capture performance distribution mappings. Additionally, we analyze MODEL-SAT without the capability encoder. Since the model representation is expressed at the natural language level, Phi-3 in Table 2 can also learn some LLM-Instruction mappings, but its performance remains inferior to that of capability-encoder-based ones.

Ablation Studies: Explore Generalization in Multimodal Scenarios. Most multimodal LLMs (MLLMs) are derived from input-extending LLMs. Multimodal MODEL-SAT is built on the Wings (Zhang et al. 2024b) training architecture, integrating model representation embeddings with visual ones. It maintains strong performance in multimodal scenarios, achieving optimal average performance across MMMU-VAL, AI2D-TEST, and CCBench datasets.

Conclusion

This paper proposes a novel model routing paradigm called *capability instruction tuning* with instruction-level model routing. It constructs a capability instruction comprising capabilities, instructions, and inquiry prompts to select the most suitable one. We present MODEL-SAT, featuring a capability encoder and lightweight LLM. It selects models without inference overhead of candidates and quickly adapts to new models. Model-SAT performs optimally across five proposed LLM routing settings and its multimodal extension.

Acknowledgments

This work is partially supported by NSFC (62376118, 62250069), Key Program of Jiangsu Science Foundation (BK20243012), Collaborative Innovation Center of Novel Software Technology and Industrialization.

References

Abdin, M. I.; Jacobs, S. A.; Awan, A. A.; et al. 2024. Phi-3 Technical Report: A Highly Capable Language Model Locally on Your Phone. *CoRR*, abs/2404.14219.

Aidar Myrzakhan, Z. S., Sondos Mahmoud Bsharat. 2024. Open-LLM-Leaderboard: From Multi-choice to Open-style Questions for LLMs Evaluation, Benchmark, and Arena. *arXiv preprint arXiv:2406.07545*.

Bao, Y.; Li, Y.; Huang, S.-L.; Zhang, L.; Zheng, L.; Zamir, A.; and Guibas, L. 2019. An Information-Theoretic Approach to Transferability in Task Transfer Learning. In *ICIP*.

Bhakthavatsalam, S.; Khashabi, D.; Khot, T.; et al. 2021. Think you have Solved Direct-Answer Question Answering? Try ARC-DA, the Direct-Answer AI2 Reasoning Challenge. *CoRR*, abs/2102.03315.

Cai, Z.; Cao, M.; Chen, H.; et al. 2024. Internlm2 technical report. *arXiv preprint arXiv:2403.17297*.

Chen, L.; Li, J.; Dong, X.; Zhang, P.; He, C.; Wang, J.; Zhao, F.; and Lin, D. 2023. Sharegpt4v: Improving large multi-modal models with better captions. *arXiv preprint arXiv:2311.12793*.

Chen, X.; Fang, H.; Lin, T.-Y.; Vedantam, R.; Gupta, S.; Dollar, P.; and Zitnick, C. L. 2015. Microsoft COCO Captions: Data Collection and Evaluation Server. arXiv:1504.00325.

Chiang, W.-L.; Li, Z.; Lin, Z.; Sheng, Y.; Wu, Z.; Zhang, H.; Zheng, L.; Zhuang, S.; Zhuang, Y.; Gonzalez, J. E.; Stoica, I.; and Xing, E. P. 2023. Vicuna: An Open-Source Chatbot Impressing GPT-4 with 90%* ChatGPT Quality.

Clark, C.; Lee, K.; Chang, M.; Kwiatkowski, T.; Collins, M.; and Toutanova, K. 2019. BoolQ: Exploring the Surprising Difficulty of Natural Yes/No Questions. In *NAACL-HLT*.

Deshpande, A.; Achille, A.; Ravichandran, A.; Li, H.; Zancato, L.; Fowlkes, C.; Bhotika, R.; Soatto, S.; and Perona, P. 2021. A linearized framework and a new benchmark for model selection for fine-tuning. *CoRR*, abs/2102.00084.

Ding, D.; Mallick, A.; Wang, C.; Sim, R.; Mukherjee, S.; Ruhle, V.; Lakshmanan, L. V.; and Awadallah, A. H. 2024. Hybrid LLM: Cost-Efficient and Quality-Aware Query Routing. *arXiv preprint arXiv:2404.14618*.

Ding, N.; Chen, X.; Levinboim, T.; Changpinyo, S.; and Soricut, R. 2022. PACTran: PAC-Bayesian Metrics for Estimating the Transferability of Pretrained Models to Classification Tasks. In *ECCV*.

Du, Z.; Qian, Y.; Liu, X.; Ding, M.; Qiu, J.; Yang, Z.; and Tang, J. 2022. GLM: General Language Model Pretraining with Autoregressive Blank Infilling. In *ACL*, 320–335.

Dubey, A.; Jauhri, A.; Pandey, A.; et al. 2024. The Llama 3 Herd of Models.

Feng, T.; Shen, Y.; and You, J. 2024. GraphRouter: A Graph-based Router for LLM Selections. *CoRR*, abs/2410.03834.

Gunasekar, S.; Zhang, Y.; Aneja, J.; et al. 2023. Textbooks Are All You Need. *CoRR*, abs/2306.11644.

Hendrycks, D.; Burns, C.; Basart, S.; Zou, A.; Mazeika, M.; Song, D.; and Steinhardt, J. 2020. Measuring massive multitask language understanding. *arXiv preprint arXiv:2009.03300*.

Hendrycks, D.; Burns, C.; Basart, S.; Zou, A.; Mazeika, M.; Song, D.; and Steinhardt, J. 2021. Measuring Massive Multitask Language Understanding. In *ICLR*.

Hu, Q. J.; Bieker, J.; Li, X.; Jiang, N.; Keigwin, B.; Ranganath, G.; Keutzer, K.; and Upadhyay, S. K. 2024. ROUTERBENCH: A Benchmark for Multi-LLM Routing System. *arXiv preprint arXiv:2403.12031*.

Jiang, A. Q.; Sablayrolles, A.; Mensch, A.; Bamford, C.; Chaplot, D. S.; de las Casas, D.; Bressand, F.; Lengyel, G.; Lample, G.; Saulnier, L.; Lavaud, L. R.; Lachaux, M.-A.; Stock, P.; Scao, T. L.; Lavril, T.; Wang, T.; Lacroix, T.; and Sayed, W. E. 2023. Mistral 7B. arXiv:2310.06825.

Jiang, A. Q.; Sablayrolles, A.; Roux, A.; Mensch, A.; Savary, B.; Bamford, C.; Chaplot, D. S.; Casas, D. d. l.; Hanna, E. B.; Bressand, F.; et al. 2024. Mixtral of experts. *arXiv preprint arXiv:2401.04088*.

Karpukhin, V.; Oguz, B.; Min, S.; Lewis, P. S. H.; Wu, L.; Edunov, S.; Chen, D.; and Yih, W. 2020. Dense Passage Retrieval for Open-Domain Question Answering. In *EMNLP*, 6769–6781. Association for Computational Linguistics.

Kembhavi, A.; Salvato, M.; Kolve, E.; Seo, M.; Hajishirzi, H.; and Farhadi, A. 2016. A diagram is worth a dozen images. In *ECCV*, 235–251.

Li, Y.; Jia, X.; Sang, R.; Zhu, Y.; Green, B.; Wang, L.; and Gong, B. 2021. Ranking neural checkpoints. In *CVPR*.

Liu, Y.; Duan, H.; Zhang, Y.; Li, B.; Zhang, S.; Zhao, W.; Yuan, Y.; Wang, J.; He, C.; Liu, Z.; Chen, K.; and Lin, D. 2024. MMBench: Is Your Multi-modal Model an All-around Player?

Lu, K.; Yuan, H.; Lin, R.; Lin, J.; Yuan, Z.; Zhou, C.; and Zhou, J. 2023. Routing to the expert: Efficient reward-guided ensemble of large language models. *arXiv preprint arXiv:2311.08692*.

Lu, X.; Liusie, A.; Raina, V.; Zhang, Y.; and Beauchamp, W. 2024. Blending Is All You Need: Cheaper, Better Alternative to Trillion-Parameters LLM. *arXiv preprint arXiv:2401.02994*.

Nguyen, C. V.; Hassner, T.; Archambeau, C.; and Seeger, M. 2020. LEEP: A New Measure to Evaluate Transferability of Learned Representations. In *ICML*.

OpenAI. 2022. ChatGPT. https://openai.com/blog/chatgpt.

OpenAI. 2023. GPT-4 Technical Report. *CoRR*, abs/2303.08774.

Pándy, M.; Agostinelli, A.; Uijlings, J. R. R.; Ferrari, V.; and Mensink, T. 2022. Transferability Estimation using Bhattacharyya Class Separability. In *CVPR*.

Reid, M.; Savinov, N.; Teplyashin, D.; et al. 2024. Gemini 1.5: Unlocking multimodal understanding across millions of tokens of context. *arXiv preprint arXiv:2403.05530*.

Sakaguchi, K.; Bras, R. L.; Bhagavatula, C.; and Choi, Y. 2020. WinoGrande: An Adversarial Winograd Schema Challenge at Scale. In *AAAI*, 8732–8740.

Shnitzer, T.; Ou, A.; Silva, M.; Soule, K.; Sun, Y.; Solomon, J.; Thompson, N.; and Yurochkin, M. 2023. Large language model routing with benchmark datasets. *arXiv preprint arXiv:2309.15789*.

Srivatsa, K.; Maurya, K. K.; and Kochmar, E. 2024. Harnessing the Power of Multiple Minds: Lessons Learned from LLM Routing. *arXiv preprint arXiv:2405.00467*.

Tan, B.; Zhu, Y.; Liu, L.; Xing, E. P.; Hu, Z.; and Chen, J. 2023. Cappy: Outperforming and Boosting Large Multi-Task LMs with a Small Scorer. *CoRR*, abs/2311.06720.

Tan, Y.; Li, Y.; and Huang, S.-L. 2021. OTCE: A Transferability Metric for Cross-Domain Cross-Task Representations. In *CVPR*.

Tan, Z.; Liu, J.; Bi, X.; Tan, P.; Zheng, Q.; Liu, H.; Xie, Y.; Zou, X.; Yu, Y.; and Zhou, Z. 2024. Beimingwu: A Learnware Dock System. In Baeza-Yates, R.; and Bonchi, F., eds., *SIGKDD*, 5773–5782. ACM.

Team, G.; Anil, R.; Borgeaud, S.; Wu, Y.; Alayrac, J.-B.; Yu, J.; Soricut, R.; Schalkwyk, J.; Dai, A. M.; Hauth, A.; et al. 2023. Gemini: a family of highly capable multimodal models. *arXiv preprint arXiv:2312.11805*.

Touvron, H.; Martin, L.; Stone, K.; et al. 2023a. Llama 2: Open Foundation and Fine-Tuned Chat Models. arXiv:2307.09288.

Touvron, H.; Martin, L.; Stone, K.; et al. 2023b. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*.

Tran, A. T.; Nguyen, C. V.; and Hassner, T. 2019. Transferability and hardness of supervised classification tasks. In *ICCV*.

Wang, A.; Singh, A.; Michael, J.; Hill, F.; Levy, O.; and Bowman, S. R. 2019. GLUE: A Multi-Task Benchmark and Analysis Platform for Natural Language Understanding. In *ICLR*.

Wei, J.; Bosma, M.; Zhao, V.; Guu, K.; Yu, A. W.; Lester, B.; Du, N.; Dai, A. M.; and Le, Q. V. 2022. Finetuned Language Models are Zero-Shot Learners. In *ICLR*.

Xiao, S.; Liu, Z.; Zhang, P.; and Muennighoff, N. 2023. C-Pack: Packaged Resources To Advance General Chinese Embedding. arXiv:2309.07597.

Yang, A.; Yang, B.; Hui, B.; et al. 2024. Qwen2 Technical Report. arXiv:2407.10671.

You, K.; Liu, Y.; Wang, J.; and Long, M. 2021. LogME: Practical Assessment of Pre-trained Models for Transfer Learning. In *ICML*.

You, K.; Liu, Y.; Zhang, Z.; Wang, J.; Jordan, M. I.; and Long, M. 2022. Ranking and Tuning Pre-trained Models: A New Paradigm for Exploiting Model Hubs. *Journal of Machine Learning Research*, 23.

Young, A.; Chen, B.; Li, C.; et al. 2024. Yi: Open Foundation Models by 01.AI. *CoRR*, abs/2403.04652.

Zeng, A.; Xu, B.; Wang, B.; et al. 2024. ChatGLM: A Family of Large Language Models from GLM-130B to GLM-4 All Tools. *CoRR*, abs/2406.12793.

Zhang, P.; Xiao, S.; Liu, Z.; Dou, Z.; and Nie, J. 2023a. Retrieve Anything To Augment Large Language Models. *CoRR*, abs/2310.07554.

Zhang, X.; Zhang, Y.; Long, D.; Xie, W.; Dai, Z.; Tang, J.; Lin, H.; Yang, B.; Xie, P.; Huang, F.; et al. 2024a. mGTE: Generalized Long-Context Text Representation and Reranking Models for Multilingual Text Retrieval. *arXiv preprint arXiv:2407.19669*.

Zhang, Y.; Huang, T.; Ding, Y.; Zhan, D.; and Ye, H. 2023b. Model Spider: Learning to Rank Pre-Trained Models Efficiently. In *NeurIPS*.

Zhang, Y.-K.; Lu, S.; Li, Y.; Ma, Y.; Chen, Q.-G.; Xu, Z.; Luo, W.; Zhang, K.; Zhan, D.-C.; and Ye, H.-J. 2024b. Wings: Learning Multimodal LLMs without Text-only Forgetting.

Zhou, Z. 2016. Learnware: on the future of machine learning. *Frontiers Comput. Sci.*, 10(4): 589–590.

Zhou, Z.; Shi, J.; Song, P.; Yang, X.; Jin, Y.; Guo, L.; and Li, Y. 2024. LawGPT: A Chinese Legal Knowledge-Enhanced Large Language Model. *CoRR*, abs/2406.04614.

Zhou, Z.; and Tan, Z. 2024. Learnware: small models do big. *Sci. China Inf. Sci.*, 67(1).