MoD: A Distribution-Based Approach for Merging Large Language Models

Quy-Anh Dang[*1,2], Chris Ngo[2]

[1]VNU University of Science, Vietnam
[2]Knovel Engineering Lab, Singapore
dangquyanh150101@gmail.com, chris.ngo@knoveleng.com

Abstract

Large language models (LLMs) have enabled the development of numerous specialized, task-specific variants. However, the maintenance and deployment of these individual models present substantial challenges in terms of resource utilization and operational efficiency. In this work, we propose the *Mixture of Distributions (MoD)* framework, a novel approach for merging LLMs that operates directly on their output probability distributions, rather than on model weights. Unlike traditional weight-averaging methods, MoD effectively preserves the specialized capabilities of individual models while enabling efficient knowledge sharing across tasks. Through extensive experimentation on mathematical reasoning benchmarks using Qwen2.5 models, we demonstrate that MoD significantly outperforms existing model merging techniques across multiple benchmarks. All code, data, and experimental materials are published at `https://github.com/knovel-eng/mod`.

1 Introduction

In this paper, we introduce a novel method for merging large language models (LLMs) called Mixture of Distributions (MoD). MoD extends previous approaches by incorporating a probabilistic framework that optimally balances task specialization with generalization. By constructing a mixture distribution over model parameters, MoD retains the unique strengths of each model while minimizing the risk of catastrophic forgetting. Our experiments with Qwen2.5-1.5B and Qwen2.5-7B models (Team, 2024; Yang et al., 2024) show that MoD significantly outperforms traditional merging techniques, particularly in multitask settings involving mathematics.

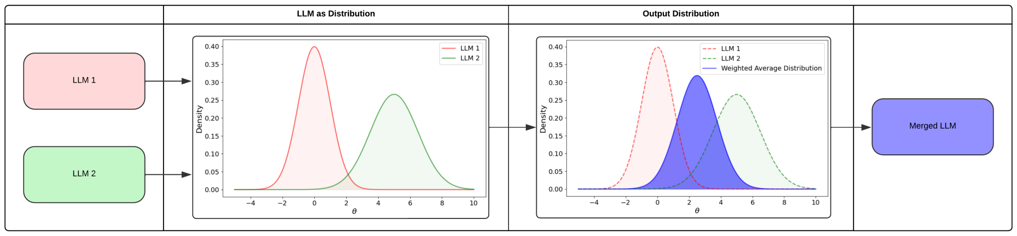
Our main contributions are as follows:

- We propose Mixture of Distributions (MoD), a novel and efficient method for merging LLMs, which demonstrates superior performance over existing approaches.
- We present extensive experimental results with Qwen2.5 models on math-related tasks, underscoring MoD's enhanced effectiveness.
- We release our code to support further research in this area[1].

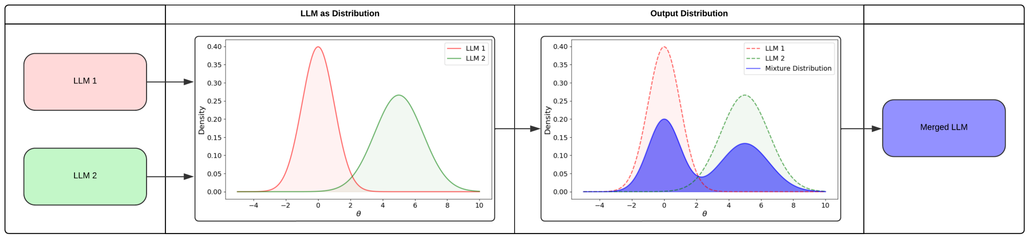
The remainder of this paper details the MoD methodology (Section 2), presents our experimental validation (Section 3), and concludes with a discussion of the results (Section 4).

[*]English Name is Andrew Dang

[1]`https://github.com/knovel-eng/mod`



(a) Weighted Average Method



(b) MoD (Our Method)

Figure 1: Comparison of our **MoD** method with the Weighted Average method. While weighted averaging methods for merging LLMs often produce new distributions that alter the characteristics of the original models (see Fig. 1a), our MoD approach effectively preserves key density structures, accurately maintaining peak densities at $\theta = 0$ and $\theta = 5$ (see Fig. 1b).

2 Methodology

In this section, we present the **Mixture of Distributions (MoD)** method for merging large language models (LLMs) through direct combination of their output probability distributions. By operating in the distribution space rather than interpolating model weights, MoD effectively addresses key challenges, particularly the distortion of density functions commonly observed in traditional weight-based approaches. We establish the mathematical framework, present the underlying motivation, and detail the implementation of our approach.

2.1 Notation and Symbols

Let θ_1 and θ_2 denote the parameter sets (weights) of two large language models (LLMs), where each θ_i follows a multivariate normal distribution, $\theta_i \sim \mathcal{N}(\mu_i, \Sigma_i)$. Given a sequence of input tokens x, let $p_{\theta_1}(x)$ and $p_{\theta_2}(x)$ represent the probability density functions (PDFs) of the two models evaluated at x. Our objective is to derive a unified output distribution $p_\theta(x)$ that preserves the essential characteristics of both original models while providing a coherent merged representation.

2.2 Motivation

Traditional weight-based merging approaches (Matena and Raffel, 2022; Garipov et al., 2018) compute the merged model's parameters through linear combination:

$$\theta = \alpha\theta_1 + (1 - \alpha)\theta_2$$

where $\alpha \in [0, 1]$. However, this approach frequently distorts the original probability distributions. Specifically, linear interpolation between θ_1 and θ_2 can lead to high density assignments at points θ where neither $p_{\theta_1}(x)$ nor $p_{\theta_2}(x)$ initially exhibited significant probability mass. This phenomenon results in poor generalization performance on downstream tasks, as illustrated in Figure 2.

To address these limitations, we propose a distribution-centric merging method that operates directly on the models' output PDFs. The Mixture of Distributions (MoD) method ensures preservation of both models' probabilistic properties while maintaining their fundamental density structures.

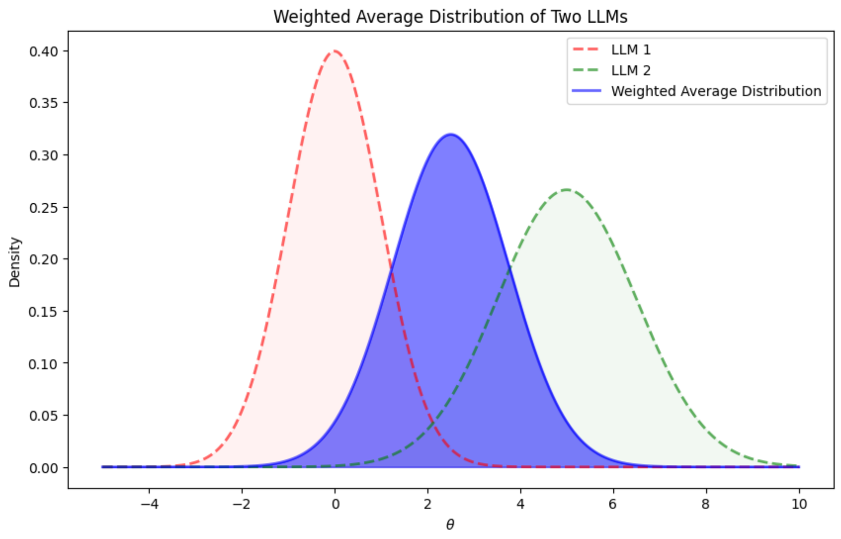


Figure 2: Distribution distortion in weighted averaging methods, demonstrating failure to preserve maximum density at $\theta = 0$ despite high density in the red distribution.

2.3 Mixture of Distributions (MoD)

The MoD framework directly combines the output distributions of the constituent models. Rather than merging parameters, we define the unified output distribution as a weighted combination of probability densities:

$$p_\theta(x) = \alpha p_{\theta_1}(x) + (1 - \alpha) p_{\theta_2}(x)$$

where $\alpha \in [0, 1]$. Here, $p_{\theta_1}(x)$ and $p_{\theta_2}(x)$ represent the probability density functions of models 1 and 2 at point x, with α and $(1 - \alpha)$ serving as mixture weights.

Solving for Mixture Weights The determination of optimal mixture weights requires solving:

$$\begin{gathered} f : \mathbb{R}^n \times \mathbb{R}^n \to \mathbb{R}^n \\ \theta = f(\theta_1, \theta_2) \end{gathered} \tag{1}$$

where f represents the mapping function that identifies appropriate mixture weights while maintaining distributional dimensionality. We approach this through quantile function analysis. The quantile function $Q(p)$ identifies the value θ_{specific} such that:

$$Q(p) = \inf\{\theta_{\text{specific}} \in \mathbb{R} : P(\theta \leq \theta_{\text{specific}}) = p\}$$

where $P(\theta \leq \theta_{\text{specific}})$ represents the cumulative distribution function (CDF) of the mixture distribution, and θ_{specific} denotes a specific value in the distribution of θ. To address the computational complexity of quantile function optimization, we employ a threshold-based approach. We normalize θ_1 within $[0, 1]$ as $\theta_{1-\text{normalize}}$ and choose α as a threshold that governs distributional contributions:

$$\theta = \begin{cases} \theta_1, & \text{if } \theta_{1-\text{normalize}} < \alpha \\ \theta_2, & \text{otherwise.} \end{cases}$$

This formulation ensures selective integration of significant distributional components.

Maximizing Density at Key Points A core advantage of MoD is its preservation of original density structures while emphasizing critical distributional regions. Unlike weight-averaging methods, which often generate spurious density peaks, MoD maintains density characteristics at crucial points x through dynamic adjustment of mixture weights based on input sequences (Figure 3).

3 Experiment

Datasets To evaluate the performance and efficiency of our merged model, we focus on a range of mathematics-focused benchmark datasets. Specifically, we use 12 datasets representing diverse

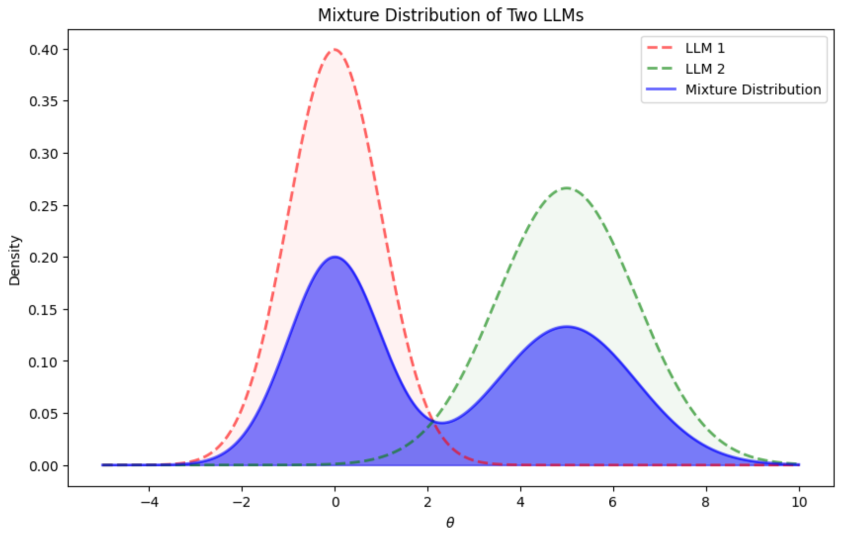


Figure 3: MoD successfully preserves maximum density characteristics at $\theta = 0$, demonstrating effective distribution merging compared to traditional approaches.

aspects of mathematical reasoning and problem-solving: GSM8K (Cobbe et al., 2021), MATH (Hendrycks et al., 2021c), SVAMP (Patel et al., 2021), ASDiv (Miao et al., 2020), MAWPS (Patel et al., 2021), CARP En (Zhang et al., 2023a), GaoKao 2023 En (Zhang et al., 2023b), OlympiadBench (He et al., 2024), College Math (Tang et al., 2024), MMLU STEM (Hendrycks et al., 2021b,a), AIME24[2], AMC23[3].

The details of these datasets, including the number of samples, are summarized in Table 1.

Table 1: Datasets and Number of Samples for Evaluation

Dataset	**#Num Samples**
GSM8K	1319
MATH	5000
College Math	2818
SVAMP	1000
ASDiv	2215
MAWPS	2065
CARP En	976
GaoKao 2023 En	385
OlympiadBench	675
MMLU STEM	3018
AIME24	30
AMC23	40

Metrics We report 5-shot pass@1 (Song et al., 2022; Chen et al., 2021) performance for MMLU (STEM) and zero-shot pass@1 performance on the remaining benchmarks (Kojima et al., 2023).

We compare the performance of our MoD method with several established model-merging techniques, including Linear (Matena and Raffel, 2022), Task-Arithmetic (Ilharco et al., 2023), TIES (Yadav et al., 2023), DARE (Yu et al., 2024), and SLERP (Shoemake, 1985). These methods represent widely used and advanced approaches for merging large language models, implemented using the Mergekit package (Goddard et al., 2024)[4].

[2]`https://huggingface.co/datasets/AI-MO/aimo-validation-aime`
[3]`https://huggingface.co/datasets/AI-MO/aimo-validation-amc`
[4]`https://github.com/arcee-ai/mergekit`

Experimental Evaluation We conducted extensive experiments to evaluate the effectiveness of MoD by merging two variants of Large Language Models (LLMs): Qwen-2.5 Instruct and Qwen-2.5 Math Instruct, each available in 1.5B and 7B parameter versions. The general-purpose Qwen-2.5 Instruct model serves as the base model with a density of 0.9, while the mathematics-specialized Qwen-2.5 Math Instruct contributes with a density of 0.1 across all experimental configurations. This combination was specifically chosen to demonstrate MoD's capability in merging models with complementary task-specific strengths.

Table 2: Performance comparison of different methods across mathematical benchmarks by merging Qwen2.5-1.5B-Instruct and Qwen2.5-1.5B-Math-Instruct

Method	Benchmarks											
	GSM8K	MATH	College Math	SVAMP	ASDiv	MAWPS	CARP En	GaoKao 2023 En	Olympiad Bench	MMLU STEM	AIME24	AMC23
Linear	39.3	11.7	9.5	61.1	64.1	76.1	23.6	14.3	3.1	34.8	0.0	2.5
Task-Arithmetic	47.2	27.9	18.5	74.3	79.6	85.5	40.4	28.1	7.0	19.4	0.0	17.5
TIES	16.5	12.0	9.5	46.4	50.9	54.1	18.8	10.9	2.4	5.8	0.0	0.0
DARE	51.5	22.1	13.9	64.3	69.8	78.8	34.2	21.8	6.1	45.1	3.3	10.0
SLERP	47.3	20.9	13.9	58.1	64.8	70.3	31.8	21.8	6.2	42.4	0.0	7.5
MoD (Our)	**74.5**	**55.8**	**38.0**	**85.1**	**88.0**	**95.1**	**56.0**	**47.0**	**20.6**	**59.5**	**10.0**	**27.5**

Baseline Models Our evaluation results for the 1.5B parameter models, presented in Table 2, demonstrate MoD's superior performance across all benchmarks. On fundamental mathematical tasks such as GSM8K, MoD achieves 74.5% accuracy, surpassing the previous state-of-the-art method DARE by a substantial margin of 23 percentage points. The performance differential becomes even more pronounced on complex benchmarks like MATH, where MoD attains 55.8% accuracy compared to Task-Arithmetic's 27.9%. Notably, MoD exhibits robust performance on both elementary and advanced mathematical reasoning tasks, achieving 95.1% on MAWPS and 88.0% on ASDiv. The method's generalization capabilities are further evidenced by strong performance on specialized benchmarks, including CARP En (56.0%) and the challenging Olympiad Bench (20.6%). In contrast, baseline methods including Linear, Task-Arithmetic, TIES, and SLERP demonstrate significant limitations, particularly on competitive mathematics benchmarks, with several methods failing to achieve measurable performance on AIME24, and TIES showing 0% accuracy on AMC23.

Table 3: Performance comparison of different methods across mathematical benchmarks by merging Qwen2.5-7B-Instruct and Qwen2.5-7B-Math-Instruct

Method	Benchmarks											
	GSM8K	MATH	College Math	SVAMP	ASDiv	MAWPS	CARP En	GaoKao 2023 En	Olympiad Bench	MMLU STEM	AIME24	AMC23
Linear	91.9	70.7	45.6	92.7	95.1	97.9	58.8	62.1	35.0	56.9	**13.3**	**47.5**
Task-Arithmetic	72.5	40.8	24.0	84.9	89.8	92.8	47.2	37.7	12.0	32.5	0.0	10.0
TIES	53.3	35.8	22.9	75.3	81.5	86.5	40.3	31.4	10.5	25.6	0.0	15.0
DARE	90.9	71.6	45.3	92.2	95.1	97.5	59.0	60.8	34.8	55.7	**13.3**	42.5
SLERP	91.5	72.2	46.0	92.2	94.9	98.0	59.8	62.3	36.4	**58.1**	**13.3**	**47.5**
MoD (Our)	**92.4**	**75.4**	**47.0**	**94.5**	**95.4**	**98.1**	**60.6**	**64.2**	**37.6**	51.0	**13.3**	**47.5**

The results for the 7B parameter models, detailed in Table 3, further validate MoD's effectiveness across diverse mathematical tasks. MoD establishes new state-of-the-art benchmarks on fundamental tests, achieving 92.4% on GSM8K and 75.4% on MATH. This superior performance extends to practical applications, with exceptional results on MAWPS (98.1%) and ASDiv (95.4%). The method demonstrates particular strength in specialized domains, achieving 64.2% on GaoKao 2023 En and 60.6% on CARP En, substantially outperforming established methods such as SLERP and DARE. MoD's capability in advanced mathematical reasoning is further demonstrated by its leading performance on Olympiad Bench (37.6%). While maintaining competitive performance on standardized tests (AIME24: 13.3%, AMC23: 47.5%), MoD's consistent superiority across varied mathematical tasks underscores its robust architecture and strong generalization capabilities.

LLM Merging Competition However, in the *LLM Merging Competition: Building LLMs Efficiently through Merging*[5], one of the competition's strict rules specifies that all models used must have been uploaded **before May 31st, 2024**. Based on prior experimental evaluations on Qwen2.5 models and several external comparisons, we selected Starling-7B (Zhu et al., 2023) as the base model to merge with Hermes-2-Pro-Mistral-7B[6]. The resulting merged model served as our final submission. Notably, during the inference phase, our method required only 15 GB of a 48 GB GPU, demonstrating its efficiency and resource effectiveness.

4 Conclusion

In this paper, we introduced Mixture of Distributions (MoD), a novel approach for merging Large Language Models that preserves and leverages the strengths of constituent models through probabilistic distribution combination. Our method demonstrates significant advantages over existing parameter-merging techniques by maintaining critical density characteristics while enabling selective integration of model capabilities. The experimental results across diverse mathematical benchmarks validate MoD's effectiveness, achieving state-of-the-art performance on both fundamental and advanced tasks. Our findings suggest that distribution-based merging approaches offer a promising direction for developing more capable and adaptable language models, particularly in specialized domains requiring precise knowledge integration.

References

Samuel K. Ainsworth, Jonathan Hayase, and Siddhartha Srinivasa. 2023. Git re-basin: Merging models modulo permutation symmetries.

Rachit Bansal, Bidisha Samanta, Siddharth Dalmia, Nitish Gupta, Shikhar Vashishth, Sriram Ganapathy, Abhishek Bapna, Prateek Jain, and Partha Talukdar. 2024. Llm augmented llms: Expanding capabilities through composition.

Mark Chen, Jerry Tworek, Heewoo Jun, Qiming Yuan, Henrique Ponde de Oliveira Pinto, Jared Kaplan, Harri Edwards, Yuri Burda, Nicholas Joseph, Greg Brockman, Alex Ray, Raul Puri, Gretchen Krueger, Michael Petrov, Heidy Khlaaf, Girish Sastry, Pamela Mishkin, Brooke Chan, Scott Gray, Nick Ryder, Mikhail Pavlov, Alethea Power, Lukasz Kaiser, Mohammad Bavarian, Clemens Winter, Philippe Tillet, Felipe Petroski Such, Dave Cummings, Matthias Plappert, Fotios Chantzis, Elizabeth Barnes, Ariel Herbert-Voss, William Hebgen Guss, Alex Nichol, Alex Paino, Nikolas Tezak, Jie Tang, Igor Babuschkin, Suchir Balaji, Shantanu Jain, William Saunders, Christopher Hesse, Andrew N. Carr, Jan Leike, Josh Achiam, Vedant Misra, Evan Morikawa, Alec Radford, Matthew Knight, Miles Brundage, Mira Murati, Katie Mayer, Peter Welinder, Bob McGrew, Dario Amodei, Sam McCandlish, Ilya Sutskever, and Wojciech Zaremba. 2021. Evaluating large language models trained on code.

Daixuan Cheng, Shaohan Huang, and Furu Wei. 2024. Adapting large language models via reading comprehension. In *The Twelfth International Conference on Learning Representations*.

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

MohammadReza Davari and Eugene Belilovsky. 2024. Model breadcrumbs: Scalable upcycling of finetuned foundation models via sparse task vectors merging. In *ICML 2024 Workshop on Foundation Models in the Wild*.

[5]https://llm-merging.github.io/index
[6]https://huggingface.co/NousResearch/Hermes-2-Pro-Mistral-7B

Jonathan Frankle, Gintare Karolina Dziugaite, Daniel Roy, and Michael Carbin. 2020. Linear mode connectivity and the lottery ticket hypothesis. In *Proceedings of the 37th International Conference on Machine Learning*, volume 119 of *Proceedings of Machine Learning Research*, pages 3259–3269. PMLR.

Timur Garipov, Pavel Izmailov, Dmitrii Podoprikhin, Dmitry P Vetrov, and Andrew G Wilson. 2018. Loss surfaces, mode connectivity, and fast ensembling of dnns. In *Advances in Neural Information Processing Systems*, volume 31. Curran Associates, Inc.

Charles Goddard, Shamane Siriwardhana, Malikeh Ehghaghi, Luke Meyers, Vlad Karpukhin, Brian Benedict, Mark McQuade, and Jacob Solawetz. 2024. Arcee's mergekit: A toolkit for merging large language models.

Chaoqun He, Renjie Luo, Yuzhuo Bai, Shengding Hu, Zhen Thai, Junhao Shen, Jinyi Hu, Xu Han, Yujie Huang, Yuxiang Zhang, Jie Liu, Lei Qi, Zhiyuan Liu, and Maosong Sun. 2024. OlympiadBench: A challenging benchmark for promoting AGI with olympiad-level bilingual multimodal scientific problems. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 3828–3850, Bangkok, Thailand. Association for Computational Linguistics.

Dan Hendrycks, Collin Burns, Steven Basart, Andrew Critch, Jerry Li, Dawn Song, and Jacob Steinhardt. 2021a. Aligning ai with shared human values. *Proceedings of the International Conference on Learning Representations (ICLR)*.

Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. 2021b. Measuring massive multitask language understanding. *Proceedings of the International Conference on Learning Representations (ICLR)*.

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

Gabriel Ilharco, Marco Tulio Ribeiro, Mitchell Wortsman, Ludwig Schmidt, Hannaneh Hajishirzi, and Ali Farhadi. 2023. Editing models with task arithmetic. In *The Eleventh International Conference on Learning Representations*.

Moritz Imfeld, Jacopo Graldi, Marco Giordano, Thomas Hofmann, Sotiris Anagnostidis, and Sidak Pal Singh. 2024. Transformer fusion with optimal transport. In *International Conference on Learning Representations*.

Albert Q. Jiang, Alexandre Sablayrolles, Arthur Mensch, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Florian Bressand, Gianna Lengyel, Guillaume Lample, Lucile Saulnier, LÃl'lio Renard Lavaud, Marie-Anne Lachaux, Pierre Stock, Teven Le Scao, Thibaut Lavril, Thomas Wang, TimothÃl'e Lacroix, and William El Sayed. 2023. Mistral 7b.

Takeshi Kojima, Shixiang Shane Gu, Machel Reid, Yutaka Matsuo, and Yusuke Iwasawa. 2023. Large language models are zero-shot reasoners.

Yanis Labrak, Adrien Bazoge, Emmanuel Morin, Pierre-Antoine Gourraud, Mickael Rouvier, and Richard Dufour. 2024. BioMistral: A collection of open-source pretrained large language models for medical domains. In *Findings of the Association for Computational Linguistics ACL 2024*, pages 5848–5864, Bangkok, Thailand and virtual meeting. Association for Computational Linguistics.

Michael S Matena and Colin A Raffel. 2022. Merging models with fisher-weighted averaging. In *Advances in Neural Information Processing Systems*, volume 35, pages 17703–17716. Curran Associates, Inc.

Shen-yun Miao, Chao-Chun Liang, and Keh-Yih Su. 2020. A diverse corpus for evaluating and developing English math word problem solvers. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 975–984, Online. Association for Computational Linguistics.

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

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

Behnam Neyshabur, Hanie Sedghi, and Chiyuan Zhang. 2021. What is being transferred in transfer learning?

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.

Rafael Rafailov, Archit Sharma, Eric Mitchell, Christopher D Manning, Stefano Ermon, and Chelsea Finn. 2023. Direct preference optimization: Your language model is secretly a reward model. In *Thirty-seventh Conference on Neural Information Processing Systems*.

Alexandre Ramé, Kartik Ahuja, Jianyu Zhang, Matthieu Cord, Léon Bottou, and David Lopez-Paz. 2022. Model ratatouille: Recycling diverse models for out-of-distribution generalization. *arXiv preprint arXiv:2212.10445*.

Baptiste RoziÃÍre, Jonas Gehring, Fabian Gloeckle, Sten Sootla, Itai Gat, Xiaoqing Ellen Tan, Yossi Adi, Jingyu Liu, Romain Sauvestre, Tal Remez, JÃľrÃľmy Rapin, Artyom Kozhevnikov, Ivan Evtimov, Joanna Bitton, Manish Bhatt, Cristian Canton Ferrer, Aaron Grattafiori, Wenhan Xiong, Alexandre DÃľfossez, Jade Copet, Faisal Azhar, Hugo Touvron, Louis Martin, Nicolas Usunier, Thomas Scialom, and Gabriel Synnaeve. 2024. Code llama: Open foundation models for code.

Victor Sanh, Albert Webson, Colin Raffel, Stephen H. Bach, Lintang Sutawika, Zaid Alyafeai, Antoine Chaffin, Arnaud Stiegler, Teven Le Scao, Arun Raja, Manan Dey, M Saiful Bari, Canwen Xu, Urmish Thakker, Shanya Sharma Sharma, Eliza Szczechla, Taewoon Kim, Gunjan Chhablani, Nihal Nayak, Debajyoti Datta, Jonathan Chang, Mike Tian-Jian Jiang, Han Wang, Matteo Manica, Sheng Shen, Zheng Xin Yong, Harshit Pandey, Rachel Bawden, Thomas Wang, Trishala Neeraj, Jos Rozen, Abheesht Sharma, Andrea Santilli, Thibault Fevry, Jason Alan Fries, Ryan Teehan, Tali Bers, Stella Biderman, Leo Gao, Thomas Wolf, and Alexander M. Rush. 2022. Multitask prompted training enables zero-shot task generalization.

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

Yisheng Song, Ting Wang, Subrota K Mondal, and Jyoti Prakash Sahoo. 2022. A comprehensive survey of few-shot learning: Evolution, applications, challenges, and opportunities.

Zhengyang Tang, Xingxing Zhang, Benyou Wang, and Furu Wei. 2024. Mathscale: Scaling instruction tuning for mathematical reasoning. In *Forty-first International Conference on Machine Learning*.

Norman Tatro, Pin-Yu Chen, Payel Das, Igor Melnyk, Prasanna Sattigeri, and Rongjie Lai. 2020. Optimizing mode connectivity via neuron alignment. In *Advances in Neural Information Processing Systems*, volume 33, pages 15300–15311. Curran Associates, Inc.

Qwen Team. 2024. Qwen2.5: A party of foundation models.

Fanqi Wan, Xinting Huang, Deng Cai, Xiaojun Quan, Wei Bi, and Shuming Shi. 2024. Knowledge fusion of large language models.

Liyuan Wang, Xingxing Zhang, Hang Su, and Jun Zhu. 2024. A comprehensive survey of continual learning: Theory, method and application. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 46(8):5362–5383.

Yufei Wang, Wanjun Zhong, Liangyou Li, Fei Mi, Xingshan Zeng, Wenyong Huang, Lifeng Shang, Xin Jiang, and Qun Liu. 2023. Aligning large language models with human: A survey. *arXiv preprint arXiv:2307.12966*.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Remi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander Rush. 2020. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45, Online. Association for Computational Linguistics.

Mitchell Wortsman, Gabriel Ilharco, Samir Ya Gadre, Rebecca Roelofs, Raphael Gontijo-Lopes, Ari S Morcos, Hongseok Namkoong, Ali Farhadi, Yair Carmon, Simon Kornblith, and Ludwig Schmidt. 2022. Model soups: averaging weights of multiple fine-tuned models improves accuracy without increasing inference time. In *Proceedings of the 39th International Conference on Machine Learning*, volume 162 of *Proceedings of Machine Learning Research*, pages 23965–23998. PMLR.

Chaoyi Wu, Weixiong Lin, Xiaoman Zhang, Ya Zhang, Yanfeng Wang, and Weidi Xie. 2023. Pmc-llama: Towards building open-source language models for medicine.

Chengyue Wu, Yukang Gan, Yixiao Ge, Zeyu Lu, Jiahao Wang, Ye Feng, Ying Shan, and Ping Luo. 2024. LLaMA pro: Progressive LLaMA with block expansion. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 6518–6537, Bangkok, Thailand. Association for Computational Linguistics.

Prateek Yadav, Derek Tam, Leshem Choshen, Colin Raffel, and Mohit Bansal. 2023. TIES-merging: Resolving interference when merging models. In *Thirty-seventh Conference on Neural Information Processing Systems*.

An Yang, Baosong Yang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Zhou, Chengpeng Li, Chengyuan Li, Dayiheng Liu, Fei Huang, Guanting Dong, Haoran Wei, Huan Lin, Jialong Tang, Jialin Wang, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Ma, Jin Xu, Jingren Zhou, Jinze Bai, Jinzheng He, Junyang Lin, Kai Dang, Keming Lu, Keqin Chen, Kexin Yang, Mei Li, Mingfeng Xue, Na Ni, Pei Zhang, Peng Wang, Ru Peng, Rui Men, Ruize Gao, Runji Lin, Shijie Wang, Shuai Bai, Sinan Tan, Tianhang Zhu, Tianhao Li, Tianyu Liu, Wenbin Ge, Xiaodong Deng, Xiaohuan Zhou, Xingzhang Ren, Xinyu Zhang, Xipin Wei, Xuancheng Ren, Yang Fan, Yang Yao, Yichang Zhang, Yu Wan, Yunfei Chu, Yuqiong Liu, Zeyu Cui, Zhenru Zhang, and Zhihao Fan. 2024. Qwen2 technical report. *arXiv preprint arXiv:2407.10671*.

Le Yu, Bowen Yu, Haiyang Yu, Fei Huang, and Yongbin Li. 2024. Language models are super mario: Absorbing abilities from homologous models as a free lunch. In *International Conference on Machine Learning*. PMLR.

Beichen Zhang, Kun Zhou, Xilin Wei, Wayne Xin Zhao, Jing Sha, Shijin Wang, and Ji-Rong Wen. 2023a. Evaluating and improving tool-augmented computation-intensive math reasoning. *arXiv preprint arXiv:2306.02408*.

Peiyuan Zhang, Guangtao Zeng, Tianduo Wang, and Wei Lu. 2024. Tinyllama: An open-source small language model.

Xiaotian Zhang, Chunyang Li, Yi Zong, Zhengyu Ying, Liang He, and Xipeng Qiu. 2023b. Evaluating the performance of large language models on gaokao benchmark.

Banghua Zhu, Evan Frick, Tianhao Wu, Hanlin Zhu, and Jiantao Jiao. 2023. Starling-7b: Improving llm helpfulness harmlessness with rlaif.

Fuzhen Zhuang, Zhiyuan Qi, Keyu Duan, Dongbo Xi, Yongchun Zhu, Hengshu Zhu, Hui Xiong, and Qing He. 2021. A comprehensive survey on transfer learning. *Proceedings of the IEEE*, 109(1):43–76.

A Background

In the past year, we have witnessed significant advancements in open-source large language models (LLMs), many of which are available on the Hugging Face model hub (Wolf et al., 2020). These models are trained on datasets containing trillions of tokens and range from 1 to 70 billion parameters (Minaee et al., 2024; Zhang et al., 2024). The diversity of open-source checkpoints is remarkable, with a broad classification into pretrained models (Zhuang et al., 2021) and models fine-tuned for instruction-following across a range of domains, such as coding (RoziÃÍre et al., 2024) and medical applications (Wu et al., 2023). However, fine-tuning separate models for each specific task poses two key challenges:

1. Each task-specific model must be stored and deployed independently, leading to increased storage and deployment costs.
2. Independently trained models are unable to share insights across tasks, limiting their ability to enhance task-specific performance or generalize to other domains (Sanh et al., 2022; Ramé et al., 2022; Yu et al., 2024).

Training these models from scratch is resource-intensive, as illustrated by the Mistral-7B model, which incurred costs between 2 to 3 million USD (Jiang et al., 2023). Further fine-tuning of pretrained models often results in catastrophic forgetting (Wang et al., 2024), where the model's original generalization capabilities degrade, impairing its performance across multiple tasks (Cheng et al., 2024; Wu et al., 2024). Moreover, aligning models to human preferences demands substantial effort and data collection, making it impractical for most research teams to replicate (Wang et al., 2023; Rafailov et al., 2023).

These challenges bring into focus the critical question of how to best utilize existing pretrained checkpoints for research and practical applications. In this context, model merging has emerged as a promising approach, combining parameters from multiple task-specific models into a single, unified model. This technique enables multitask and continual learning while minimizing catastrophic forgetting, all without the steep costs of training models from scratch (Yadav et al., 2023).

B Related Work

Model Merging Recent advances in large language models (LLMs) have highlighted model merging as a crucial strategy for combining the capabilities of multiple models into a unified system (Ainsworth et al., 2023; Goddard et al., 2024; Labrak et al., 2024). This approach has gained prominence for its ability to enhance multitask performance and enable continual learning without requiring costly retraining procedures. Initial investigations in this domain explored weight averaging techniques, which directly combined parameters of models sharing identical architectures and initializations (Matena and Raffel, 2022; Garipov et al., 2018). While these methods demonstrated promising results, they revealed significant limitations when applied to models trained on heterogeneous tasks or initialized differently, prompting the development of more sophisticated approaches.

Merging Techniques The theoretical foundation for many modern merging approaches stems from Linear Mode Connectivity (LMC) (Frankle et al., 2020), which demonstrates that models fine-tuned from a common pretrained checkpoint often permit linear interpolation while maintaining performance integrity (Nagarajan and Kolter, 2021; Neyshabur et al., 2021). This insight has led to the development of several practical methodologies. Model Soups (Wortsman et al., 2022) and weight averaging techniques (Matena and Raffel, 2022; Garipov et al., 2018) offer elegant solutions for merging models with shared initialization. Task Arithmetic (Ilharco et al., 2023) extends this framework by introducing task vectors, demonstrating that arithmetic operations on the differences between fine-tuned models and their base model yield semantically meaningful results. More recent approaches, including Trim, Elect Sign Merge (TIES merging) (Yadav et al., 2023), Model Breadcrumbs (Davari and Belilovsky, 2024), and Drop And REscale (DARE) (Yu et al., 2024), have introduced sophisticated methods for sparsifying and combining task vectors, enabling the integration of multiple models while preserving their individual capabilities. The application of Spherical Linear intERPolation (SLERP) (Shoemake, 1985) represents a significant advancement over simple weight averaging, revealing that spherical paths often present lower loss barriers compared to direct linear interpolation.

The challenge of merging independently trained models with different initializations presents a more complex scenario. Git-Rebasin (Ainsworth et al., 2023) addresses this challenge by exploiting neural networks' permutation symmetry, enabling the alignment of neurons across independently trained models to facilitate effective merging. Complementary approaches such as Optimizing Mode Connectivity via Neuron Alignment (Tatro et al., 2020) and Optimal Transport Fusion (OTFusion) (Imfeld et al., 2024) have further developed this concept, demonstrating enhanced capabilities in reducing interpolation barriers between models with distinct random initializations.

Recent research has pushed the boundaries of model merging by exploring the integration of models with heterogeneous architectures. The Composition to Augment Language Models (CALM) approach (Bansal et al., 2024) leverages cross-attention mechanisms to integrate models with diverse neural architectures, marking a significant advancement in the field. Similarly, the FUSELLM framework (Wan et al., 2024) focuses on aligning probabilistic distributions across different language models, facilitating the fusion of models with varying output characteristics. While these methods incur higher computational costs and may require additional pretraining, they represent important progress toward creating more versatile and adaptable models.

In this paper, we introduce Mixture of Distributions (MoD), a novel approach that shifts the paradigm from weight interpolation to probabilistic output combination. Our method leverages the probability density functions of large language models, enabling a more nuanced integration that preserves the distinctive strengths of each model. The sections detail the methodology of MoD (Section 2), present our experimental validation (Section 3), provide conclusions (Section 4).

C Limitations and Future Work

While MoD demonstrates superior performance compared to existing methods, we acknowledge some limitations in our current study. First, our experimental validation is primarily confined to the mathematical domain, which, while comprehensive, may not fully represent the method's generalizability across other specialized fields. Second, our current approach employs a simplified strategy for determining mixture weights, which may not capture optimal combinations for all scenarios.

These limitations suggest several promising directions for future research. First, extending the evaluation of MoD to diverse domains beyond mathematics would provide valuable insights into the method's robustness and general applicability. Second, developing more sophisticated approaches for determining optimal mixture weights could potentially enhance the method's performance further. Additionally, investigating the theoretical foundations of distribution-based merging approaches could lead to more principled strategies for model combination and integration. These directions would contribute to a deeper understanding of model merging techniques and their applications in developing more capable language models.

D YAML Configuration

This appendix details the configuration parameters implemented across all methodologies in this study, specifically for the 1.5B parameter model variant. These configurations are similar to those employed in the 7B parameter implementation.

```
base_model: Qwen/Qwen2.5-1.5B-Instruct

experts:
  - source_model: Qwen/Qwen2.5-1.5B-Instruct
  - source_model: Qwen/Qwen2.5-Math-1.5B-Instruct

model_kwargs:
  - device_map: cuda
  - low_cpu_mem_usage: True
  - trust_remote_code: True
```

```
weights: [0.9, 0.1]
```

Configuration 1: MoD Method (Our)

```
models:
  - model: Qwen/Qwen2.5-1.5B-Instruct
    parameters:
      weight: 0.9
  - model: Qwen/Qwen2.5-Math-1.5B-Instruct
    parameters:
      weight: 0.1
merge_method: linear
dtype: float16
```

Configuration 2: Linear Method

```
models:
  - model: Qwen/Qwen2.5-1.5B-Instruct
    # No parameters necessary for base model
  - model: Qwen/Qwen2.5-Math-1.5B-Instruct
    parameters:
      density: 0.9
      weight: 0.1
merge_method: dare_ties
base_model: Qwen/Qwen2.5-1.5B-Instruct
parameters:
  int8_mask: true
dtype: bfloat16
```

Configuration 3: DARE Method

```
models:
  - model: Qwen/Qwen2.5-1.5B-Instruct
  - model: Qwen/Qwen2.5-Math-1.5B-Instruct
    parameters:
      density: 0.9
      weight: 0.1
merge_method: ties
base_model: Qwen/Qwen2.5-1.5B-Instruct
parameters:
  normalize: true
dtype: bfloat16
```

Configuration 4: TIES Method

```
models:
  - model: Qwen/Qwen2.5-1.5B-Instruct
    parameters:
      weight: 0.9

  - model: Qwen/Qwen2.5-Math-1.5B-Instruct
    parameters:
      weight: 0.1

base_model: Qwen/Qwen2.5-1.5B-Instruct
merge_method: task_arithmetic
parameters:
  normalize: true
  int8_mask: true

dtype: bfloat16
```

Configuration 5: Task Arithmetic Method

```
slices:
  - sources:
      - model: Qwen/Qwen2.5-1.5B-Instruct
        layer_range: [0, 28]
      - model: Qwen/Qwen2.5-Math-1.5B-Instruct
        layer_range: [0, 28]
merge_method: slerp
base_model: Qwen/Qwen2.5-1.5B-Instruct
parameters:
  t:
    - filter: self_attn
      value: 0.1
    - filter: mlp
      value: 0.1
    - value: 0.1
dtype: bfloat16
```

Configuration 6: SLERP Method