SoupLM: Model Integration in Large Language and Multi-Modal Models

Anonymous ACL submission

Abstract

Training large language models (LLMs) and multimodal LLMs necessitates significant computing resources, and existing publicly available LLMs are typically pre-trained on diverse, privately curated datasets spanning various tasks. For instance, LLaMA, Vicuna, and LLaVA are three LLM variants trained with LLaMA base models using very different training recipes, tasks, and data modalities. The training cost and complexity for such LLM variants grow rapidly. In this work, we propose to use a soup strategy to assemble these LLM variants into a single well-generalized multimodal LLM (SoupLM) in a cost-efficient manner. Assembling these LLM variants efficiently brings knowledge and specialities trained from different domains and data modalities into an integrated one (e.g., chatbot speciality from user-shared conversations for Vicuna, and visual capacity from vision-language data for LLaVA), therefore, to avoid computing costs of repetitive training on several different domains. We propose series of soup strategies to systematically benchmark performance gains across various configurations, and probe the soup behavior across base models in the interpolation space.

1 Introduction

Training large language models (LLMs) (Brown et al., 2020; Achiam et al., 2023; Devlin et al., 2018) presents several significant challenges, such as how to deploy immense size models on infrastructures and make large-scale optimization (Xie et al., 2024; Narayanan et al., 2021), and how to collect and prepare massive training data to match the model size (Swayamdipta et al., 2020; Wang et al., 2022). As a result, the computational cost and other efforts of training such networks is rapidly growing. For example, training a model like LLaMA3-7B (Touvron et al., 2023) requires an extensive amount of computation with carefully defined data and training recipe, not to mention a 70B model demands even more resources and training complexity, measured in thousands of H100 hours (Choquette, 2023). Constraints caused by these substantial computational costs mean that research into new large language models is often restricted to a limited number of teams with extensive resources, which may hinder the community development.

Moreover, while extending the model capacities for multiple domains by transitioning LLMs into large multi-modal models (LMMs), additional challenges arise (Liu et al., 2024b; Zhu et al., 2023; Yan et al., 2021). Training LMMs typically follows the post-training approach, which involves finetuning the base model with a multi-modal instructional tuning dataset (Liu et al., 2024a; Li et al., 2024). For example, LLaVA (Liu et al., 2024b) enable its base Vicuna (Zheng et al., 2023) model to understand visual input by finetuning it on vision-language instruction data. In addition, extending the model with new architecture, such as branch mixing and training (Sukhbaatar et al., 2024) under Mixture-of-Experts (MoE) design (Shazeer et al., 2017), further complicates the process. Overall, as models become more unified and integrate diverse modalities, they face new issues like data and modality drift. Such issues require even more complicated data and optimization recipes, which are more complex than traditional challenges and further increase the multi-modal training costs.

In this context, the concept of model soup emerges as an effective strategy to merge the base model and its finetuned variants. It initially focuses on image classification task (Wortsman et al., 2022). Instead of picking the model with highest validation accuracy, model soup combines tuned models of different hyperparameter configurations, where all variants are trained from the same random initialized model that seen as the base model. The soup strategy obtains a robust model with the highest performance, which can be generalized to several visual backbones like CLIP (Radford et al.,

2021) and ViT (Dosovitskiy et al., 2020). Unlike typical ensemble, the model soup directly merges weights of model variants, resulting in no additional inference and memory costs.

Motivated by the challenges above with model soup inspiration, in this paper, we systematically study how to merge the model variants of different domains in the context of the large language model. More specifically, we focus on language (LLMs) and vision-language (LMMs) domains upon the autoregressive architecture (Radford et al., 2019). We take Vicuna, and its variant LLaVA as two base models for a study case to explore the model integration in LLMs and LMMs, namely, SoupLM. We propose series of soup strategies from naive weight average into finegrained learnable soup, and find SoupLM improves both language and multi-modal task performances as an integrated well-generalized model. Such process has no additional inference cost and requires almost ignorable extra training cost, where naive soup has no training cost and learnable soup has tiny effort to adjust the soup weight. We systematically benchmark extensive evaluations across different soup configurations to fully explore its improvement potential, statistically providing intuitions to find a better soup setting.

We are also curious about the finegrained soup behavior across base models. For example, if the base models are given, what is the learned α distributions under different tuning conditions? Correspondingly, we make detailed analysis upon different settings and further use a simple regularized soup strategy, to initially probe the soup dynamics. To summarize our effort of this paper:

- We propose SoupLM to first investigate the model soup strategy in the context of the autoregressive architecture. SoupLM integrates base models of different domains as a well-generalized multi-modal model, introducing ignorable training and no inference cost.

- We systematically benchmark the learnable soup strategy across various configurations to test the potential performance gain. It observes statistical patterns under the hyperparameter space, and inspires a principle design to derive better soup settings.

- Finegrained soup behaviors are initially probed by learnable and regularized soup, and we find the interpolation distributions are stable under training constraints and certain fine-tuning supervisions. It is expected to inspire more soup mechanism studies to probe its behaviors in an interpretable way.

2 Method

This section introduces *vanilla*, *learnable*, and *regularized* soup strategies for our SoupLM exploration, where *vanilla* initially explores the effectiveness of soup, *learnable* serves as our central method and *regularized* mainly for soup behavior analysis to validate our hypothesis. Given a set of base models with isomorphic model structures $M = \{f(\theta^1), f(\theta^2), ..., f(\theta^n)\}$, where n is the number of base models. Here, the model $f(\cdot)$ generally represents network module at different granularities (e.g., each weight, each MLP block, and the whole model), which varies according to different soup strategies. We keep the model structure $f(\cdot)$ fixed and merge θ^* to obtain a souped model $f(\theta^s)$. The merging also keeps the weight θ^* fixed and only assign a bunch of α to bridge base models. Then, the integrated one is given by

$$f(\theta^s) = \sum_{i=1}^{n} \alpha^i \theta^i, \quad (1)$$

where α is the critical factor of our study and explored by following soup strategies. In this study, we specifically consider two autoregressive Transformer (Vaswani et al., 2017) base models, Vicuna and LLaVA, for the following soup strategies and the number of base models can be easily enlarged. And we ensure $\sum_{i=1}^{n} \alpha^i = 1$ to interpolate weight in linear model space.

2.1 Vanilla Soup

We use vanilla soup as a simple baseline to initially explore if directly combining weights of two base models improves the performance. Herein, $f(\cdot)$ represents the whole model, which is the largest granularity. We manually set different ratios α^1 (e.g., 0.5) for the first base model and use $\alpha^2 = 1 - \alpha^1$ for the second. The vanilla souped model is given by

$$f(\theta^s) = \alpha^1 \theta^1 + (1 - \alpha^1)\theta^2, \quad (2)$$

where we use $\alpha^1 = \{0.1, 0.2, ..., 0.9\}$ in our experiments (see Sec. 3.2)

2.2 Learnable Soup

Instead of merging base models using model-level granularity as vanilla soup, we propose to refine the process by decreasing the soup granularity

Table 1: Summary of five meta sets from language and vision-language domains.

Meta Set	MMMU	LLaVA665K	MMLU	GSM8k	Hellaswag
Number of validation	150	665K	99.8K	7.47K	39.9K
Number of test	900	60 (LLaVA-Bench)	14K	1.32K	10K

to bridge base models in a fine-grained way, which is the central method in this paper. Concretely, we choose each module in Transformer block as a smaller soup unit $f(\cdot)$, such as the *Q, K, V, O* mappings in attention block and *up, down* mappings in MLP block. In addition, we also include all normalization layers, the very first embedding layer, and the last LM head mapping as units for soup. Basically, this process can be seen as a finegrained soup at *per-mapping* granularity.

Rather than manually assignment, we propose to optimize the finegrained α using a tiny development set D. The optimization follows the typical finetuning protocol of autoregressive model to minimize the next token prediction loss, but only tuning the $\alpha_{[*,*]}$ while fixing both base models ($\theta^1_{[*,*]}$, $\theta^2_{[*,*]}$). It integrates the weights in the model space spanned by two base models, which is formally given by:

$$\alpha_{[s,l]} = \arg\min_{\alpha} \mathcal{L}(\alpha_{[s,l]}; \theta^1_{[s,l]}, \theta^2_{[s,l]}, \mathcal{D}), \qquad (3)$$

where s represents different soup units (e.g., Q/up project in attention/MLP) and l means different Transformer layer indices. $\mathcal{L}(\cdot;\cdot)$ is the autoregressive loss. It elaborates the merging process by delicately tuning the soup weights following the data supervision to better take advantages of both base models. Such refinement with smaller soup granularity firstly leads to a more flexible model interpolation space to benefit further performance gain. Furthermore, it provides an access to investigate the functional mechanism of each soup unit by analyzing their merging behaviors. Please note that the learnable soup can be further elaborated by reducing the soup granularity such as neuron or other self-defined units and we keep the per-mapping soup units for this study.

2.3 Regularized Soup

Learnable soup picks smaller granularity and merges base models by fixing the original ones. It also provides an intuitive way to investigate the model merging behaviors in the model space. To do so, we involve a regularization term to elaborate the soup process and point out the merging behavior for analysis. We use L1 normalization on the soup α and augment Eq. 3 as

$$\mathcal{L}_{reg}(\alpha) = \mathcal{L}(\alpha; \theta^1, \theta^2, D) + \lambda \|\alpha\|_1, \qquad (4)$$

where we omit the subscript of $[s,l]$ for α. λ is the regularization strength parameter and $\mathcal{L}_{reg}$ is the final regularized training objective. Other regularization formats (e.g., L2) can be easily extended and we simply consider L1 here. Through adding regularization on the α, its optimized values are constrainted close to its initializations. In this way, we set increasing regularization magnitudes to observe the changes of soup distribution, and validate the hypothesis that model soup performs stable behavior according to the given base models. Different from learnable soup above aiming to exhaust the soup potential, regularized soup is mainly to provide further intuitions of model soup behavior among base models during finetuning.

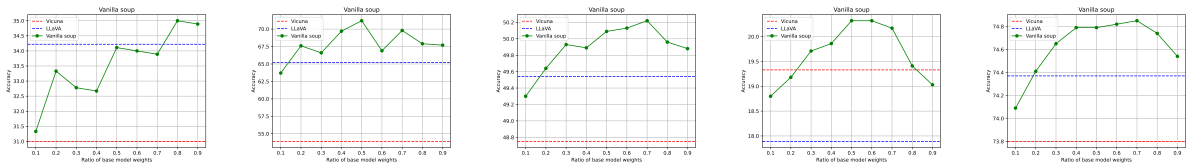
3 Experiments

3.1 Principle Design

Since we study series finegrained soup strategies based on multi-modal models with massive parameters, it is critical to propose a feasible path to manage the hyperparameter spaces for a reasonable exploration pipeline. Therefore, we briefly introduce *base models*, *meta sets*, and *soup strategies*, then elaborate them in the following sections.

Base Models

We specifically consider vision-language domains and choose representative Vicuna (Zheng et al., 2024) and its visual variants LLaVA (Liu et al., 2024b) as two base models. Vicuna is finetuned from LLaMA (Touvron et al., 2023) using human conversation instruction, which enable it with chatbot function. LLaVA is further finetuned from Vicuna using vision-language instructions, therefore, the model can understand visual input and interact with users by language. Basically, they are both variants from original LLaMA, sharing the isomorphical structures on language decoder, and



(a) MMMU (b) LLaVA-Bench (c) MMLU (d) GSM8K (e) Hellaswag

Figure 1: Vanilla soup evaluations on five meta sets, including MMMU, LLaVA-Bench for multi-modality, and MMLU, GSM8K, Hellaswag for language. The x-axis shows increasing soup ratio from 0.1 to 0.9 of (α^1) of LLaVA. The y-axis means the evaluation performance. Green dots serve as soup performances. Two base models are shown in blue and red lines. We find vanilla soup generally outperforms baselines, and direct average with $\alpha^1 = 0.5$ often obtains better results except for the MMMU dataset.

their weights are consistently optimized step-by-step. Such consistencies benefits to further explore model interpolation upon these two models. Specifically, we use their 7B and V1.5 version to represent language and multi-modal domains. Among our experiments, we fix two base models and only investigate the interpolation weight α based on different soup strategies. We also fix the visual encoder and alignment MLP of LLaVA for both training and test. Please note the base model candidates can be easily generalized into other domains (e.g., audio and video) and multiple (>2) base models, but we only take language and vision-language ones in our study.

Meta Sets

Various evaluation benchmarks are designed for both language and vision-language models from different purposes, we choose a few representative ones as our meta (development) sets for benchmarking. Such meta sets fulfil: 1) they are well-prepared and robust evaluation datasets for certain general purposes, 2) they cover both language and vision-language multi-modal domains, 3) they contain training and corresponding test set. In this study, we choose MMMU (Yue et al., 2023), LLaVA665K (Liu et al., 2023a) for vision-language domain; MMLU (Hendrycks et al., 2021), GSM8K (Cobbe et al., 2021), and Hellaswag (Zellers et al., 2019) for language domain. We use their given training set for finetuning and test set for evaluation[1]. The meta sets information is summarized in Tab. 1

Soup Strategies

We study a series of soup strategies that interpolate two base models while fixing their original weights based on five meta sets. At first, we simply use vanilla soup as a initial baseline (Sec. 2.1) to test if such a naive method improves performance on 5 meta sets without complicated experimental designs. Then, we expound learnable soup (Sec. 2.2) as the central role in our experiments to 1) fully explore the soup potential for performance gain, 2) statistically depict the soup performance patterns under multiple hyperparameter dimensions. Finally, other than pursuing better performance, we deploy regularized soup (Sec. 2.3) to intuitively probe the stability of soup behavior under various regularized training scenarios.

3.2 Vanilla Soup

Our exploration begins with the simplest vanilla soup. Given Vicuna and LLaVA as base models, we set $\alpha^1 = \{0.1, 0.2, ..., 0.9\}$ (α^2 correspondingly obtained by Eq. 2) to merge them and test on meta sets. Fig. 1 shows the soup performance (green dots) and two base models as baselines (blue and red lines). We conclude 1) LLaVA naturally improves vision-language tasks (MMMU and LLaVA-Bench), as it is visually finetuned. Further, since the visual finetuning also contain language partition, it also enhances two general language-only tasks (MMLU and Hellaswag), but not for GSM8K which is more specific in math. 2) Vanilla soup performs generally better than two baselines proving the soup strategy effectiveness. 3) For 4 out of 5 meta sets (except MMMU), the trending of vanilla soup performance shows half-half average of base models obtains better results compared with other ratios, especially certain extreme cases (e.g., $\alpha^1 = 0.1, 0.9$). However, this is not for MMMU which highly relies on the visual finetuning for improvement. We track the performance comparison in Tab. 2

[1]LLaVA665K is the instruction finetuning data for LLaVA without corresponding test set, we regard the LLaVA-Bench (Liu et al., 2024b) as its in-domain test set.

Table 2: Performance summary of different soup strategies on five meta sets. It includes two base model baselines and records the best performance of three soup strategies among various configurations.

Model	MMMU	LLaVA-Bench	MMLU	GSM8k	Hellaswag
Vicuna-7B-v1.5	31.00	53.90	48.75	19.33	73.80
LLaVA-7B-v1.5	34.22	65.20	49.54	17.89	74.37
Vanilla Soup*	34.89	71.20	50.22	20.32	74.85
Single Meta-Set*	**35.78**	**72.10**	51.24	21.15	**74.86**
Pair Meta-Set*	35.11	-	**51.65**	**21.38**	74.82

3.3 Learnable Soup

After vanilla soup as a simple proof-of-concept validation, we then go into details of learnable soup method, where we elaborate extensive ablation study. This ablation aims to firstly find if such fine-grained soup can 1) further obtain performance gain compared with vanilla soup, and 2) find statistical soup patterns across several hyperparameter dimensions, helping to understand the soup sensitivity under different settings. Specifically, given five meta sets for finetuning and evaluation, we cover 1) datasets, 2) epoch, 3) learning rate, 4) sample number, 5) sample ratio, and 6) activation aspects for ablations. It is hard to systematically discover the global oracle setting, as all dimensions are entangled together. Therefore, we heuristically design a path to search for the best combination from several rounds of ablation study. Along with them, we summarize the soup performance patterns in a statistical way.

First Round

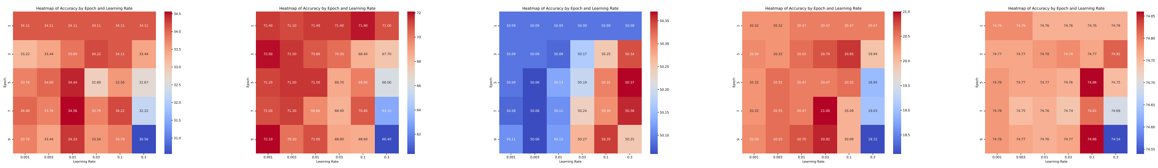
We begin with searching for the best meta sets combination by: 1) using each individual meta set to finetune, 2) fixing the total sample number as 1000, 3) ablating the epoch from 1 to 9, 4) ablating the learning rate from 0.001 to 0.3, 5) evaluating on 5 meta sets. We representatively show a bunch of visualization in Fig. 2, which uses MMMU as finetuning set. The rest visualizations are supplemented in Fig. 7 in appendix due to the limited space. Corresponding performances are also tracked in Tab. 2. To summarize all visualizations, we calculate the mean and maximum performance of 5 meta sets across epochs and learning rates in Tab. 3. We conclude 1) finegrained learnable soup outperforms vanilla soup for each evaluation task, obtaining further performance gain compared with two baselines. However, the best results of each meta set are based on different hyperparameter settings. Due to the different properties of training and evaluation sets, the soup performance varies significantly among them. 2) There are clear trends of performance changes with ablated learning rates and epochs (color changes in heatmap plots), indicating a clear hyperparameter patterns at least within one meta set, but may change across meta sets. 3) The soup patterns dramatically differs across different training-evaluation sets combination. For example, MM-MM observes the best combination in the middle with the worst at bottom right corner, but MM-ML shows completely different clues. 4) Based on the results in Tab. 3, we find LLaVA665K is better than MMMU to be chosen in multi-modal domain. MMLU and Hellaswag show their advantages in language-only domain. Considering, MMLU follows the multiple-choice task instead of typical natural language, thus we choose MMLU instead of Hellaswag.

As a summary, the first round ablation results in 1) learnable soup further improves the evaluation performance, 2) soup performance patterns change dramatically across different finetuning and test set combinations, but show clear pattern given a fixed training and test pair, and 3) overall, we use LLaVA665K and MMLU as training sets for following ablation rounds.

Second Round

Using LLaVA665K and MMLU as meta sets, we conduct the second round ablation study. It aims to find the best hyperparameter setting including 1) learning rate, 2) epoch, 3) sample number, and 4) activation. Concretely, we 1) fixing the training data as LLaVA665K and MMLU, 2) ablating sample numbers from 10 to 1000, 2) ablating learning rate from 0.001 to 0.3, 3) ablating epoch from 1 to 9, 4) ablating activation using *sigmoid*, *linear*, *clamp*, and *softmax* options[2]. Please note, from

[2]The implementation details of activation: We initialize the α as 0, 0.5, 0.5, and (0.5, 0.5) for sigmoid, linear, clamp, and softmax, respectively, where we finetune α^1 and α^2 for softmax and only learn α^1 and $\alpha^2 = 1 - \alpha^1$ for the rest.



(a) MM-MM (b) MM-L (c) MM-ML (d) MM-G (e) MM-H

Figure 2: Representative MMMU single set evaluation. MM, L, ML, G, and H represent MMMU, LLaVA-Bench, MMLU, GSK8K, and Hellaswag, respectively. For each heatmap, x/y axis means ablated learning rates and epochs. Different colors show the performance variances on evaluation sets.

Table 3: Statistical summary of first round ablation: mean/max accuracy across epochs and learning rates

Meta\Eval	MMMU	LLaVA-Bench	MMLU	GSM8k	Hellaswag	Sum
MMMU	33.68/34.56	69.77/72.10	50.17/50.37	20.29/21.00	74.77/74.86	248.68/252.89
LLaVA665k	34.63/35.78	69.55/72.10	49.99/50.20	20.13/21.08	74.72/74.84	**249.02/254.00**
MMLU	32.48/34.89	64.56/71.30	50.53/51.13	19.46/21.15	74.58/74.81	**241.61**/253.28
GSM8K	31.65/34.33	60.97/71.80	50.37/51.24	19.24/21.00	74.51/74.86	236.74/253.23
Hellaswag	31.64/35.11	60.02/71.80	50.21/51.03	18.98/21.00	74.35/74.84	235.20/**253.78**

this round, we only evaluate four meta sets except for LLaVA-Bench here due to its massive request of OpenAI API. We show the representative visualization in Fig. 3 and the rest visualizations are supplemented in the appendix (Fig. 8) due to the limited space. We conclude 1) using LLaVA665K and MMLU as paired meta sets further improve the performance but not significantly. Similarly, the best setting for each evaluation task varies, indicating the soup process is sensitive to specific test set. 2) The performance changes are still clear given a fixed finetuning and test combination across learning rate, epoch, and activation, however, not consistent while varying the number of samples. Especially for MMLU task, the trend changes reversely as the number of sample increases. 3) The activation choice affects performances by a large margin such as the linear activation dramatically affect the performance, and overall the other options perform better than linear. We track the pair meta sets results in Tab. 2 and we search the best setting based on overall performance on meta sets. The statistical summary is given by Fig. 9 in appendix and we choose the best setting with 3 epoch, 50 sample, 0.1 learning rate, and softmax activation.

As a summary, given LLaVA665K and MMLU as meta sets, the second round ablation search the epoch, sample number, learning rate, and activations. We find the little performance gain compared with the first round and the soup performances vary across differen settings. The best overall setting is picked for the next round ablation.

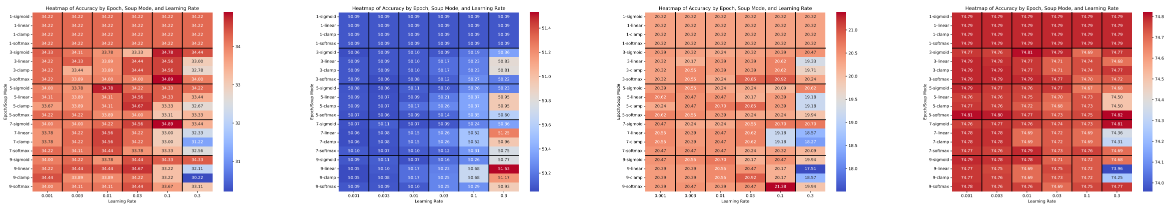
Third Round

We finally make ablation on the ratio of given meta sets as the last round. Given the setting from first and second round, we adjust the sample ratio from LLaVA665K and MMLU from 5-95 to 95-5 to test if the ratio is a sensitive factor for evaluation. Performance variances are shown in Fig. 4. We conclude there are no clear trend according to the sample ratio based on the given setting, except for the MMLU task. Overall, the 50-50 ratio achieves the averagely better results than others. Through the three rounds heuristic ablations, we benchmark the soup performance on 5 meta sets, covering several hyperparameter configurations and fully exploring the model soup potential. Statistically, we find the better configurations and provide intuitions of the hyperparameter properties for SoupLM.

More Evaluations

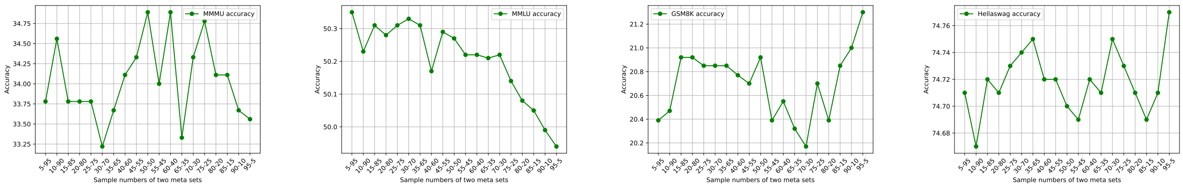
Using the best soup setting from three rounds ablation, we evaluate its soup performance on more diverse evaluation tasks other than given five meta sets. We choose Winoground (Thrush et al., 2022), PiQA (Bisk et al., 2020), MathQA (Amini et al., 2019), BoolQA (Clark et al., 2019), and BBH (Suzgun et al., 2022) for language and POPE (Li et al.,

For sigmoid, linear, and clamp, we apply sigmoid, keep it the same, or clamp (from 0 to 1) operation on α^1, then, obtain $\alpha^2 = 1 - \alpha^1$. For softmax, we directly apply softmax operation on α^1 and α^2.



(a) 50:MM (b) 50:ML (c) 50:G (d) 50:H

Figure 3: Second round ablation for epoch, sample number, learning rate, and activation. MM, ML, G, H are for MMMU, MMLU, GSM8K, Hellaswag. Colors show performance changes. X-axis is learning rate. Y-axis is number of epoch and activation function. Here, we use 50 samples for LLaVA665K and 50 samples for MMLU.



(a) MMMU (b) MMLU (c) GSM8K (d) Hellaswag

Figure 4: Ratio ablation on MMMU, MMLU, GSM8K, and Hellaswag on LLaVA665K and MMLU meta sets.

2023) and MM-Bench (Liu et al., 2023b) for vision-language domains. Tab. 4 in appendix shows model soup generally outperforms baselines, but may also drop the performance for certain tasks, which may due to severe domain drift such as MathQA.

4 Soup Behavior

Beside of discussing performance gain, we initially study the soup behavior based on empirical results (Sec. 3) and regularized soup (Sec. 2.3). We are curious if the soup dynamics follow certain patterns under different training constraints and supervisions. We first probe such behavior through visualizing the learned α from different meta sets. Since we are only curious about its distribution, we tune the α with 0.3 learning rate, 9 epochs, and 1000 samples to ensure it is fully optimized. We visualize an exemplar case of key mapping across the language decoder layers (Fig. 5). We visualize the rest of visualizations in the appendix (Sec. A.5) including other mappings, normalization layers, etc. Furthermore, we set series of regularization magnitudes to observe if the soup behavior varies under training constraints. We visualize the regularized soup of key mapping with 0.0001 magnitude in Fig. 6 and leave the rest magnitudes in appendix (Sec. A.6). Figures show how the two base models are integrated into the souped model. Through the x-axis, they show different meta sets across different layers. Y-axis indicates the learned ratios between Vicuna and LLaVA. If the ratio is more than 0.5, meaning the corresponging base model dominates the soup process for this mapping, we color it as green, otherwise, as red. According to these figures, we observe 1) for some layers, the color distributions are very neat across different meta set, while for some others, these consistencies are not stable. 2) For the α under regularized soup, we find the soup trends are not vulnerable, only generally close to the initial value 0.5 as constrained by the regularization. 3) Please note different mappings may show varied distributions and see more cases in the appendix. Overall, we draw the conclusions that the soup behaviors are not vulnerable under regularized constraints, and show consistency across certain layers but may vary different layers and mappings. In this study, we initially probe the soup behavior to provide intuitions by visualizations, and hope it inspires more model interpolation mechanism explorations.

5 Related Work

5.1 Large Language and Multi-Modal Models

Large-scale language models (LLMs) show that large-scale pretraining enables model with strong language capacity with massive knowledge (Radford et al., 2018, 2019; Brown et al., 2020; Devlin et al., 2018; Liu et al., 2019; Touvron et al.,

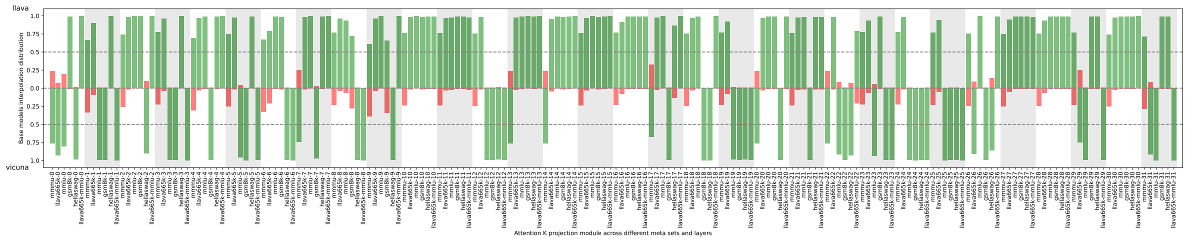


Figure 5: Learned alpha distribution on LLaVA-Vicuna model space of key mapping across different meta sets and Transformer layers. This set of α is tuned on 9 epochs, 0.3 learning rate, and 1000 samples. Certain layers show stable consistency across different meta sets.

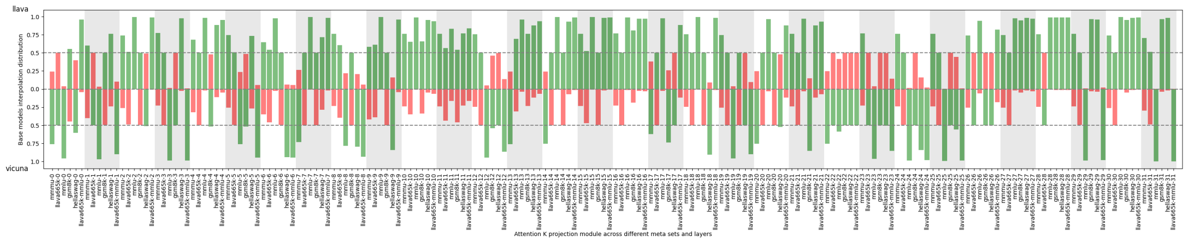


Figure 6: Learned alpha distribution with 0.0001 regularization. It follows the same finetuning settings as figure above. The regularization limits the α values close to the initial 0.5 but shows the same α distribution with the unregularized one.

2023). Downstream finetuning improves task performances and aligns the model behavior with human preference (Ouyang et al., 2022; Zheng et al., 2024; Zhang et al., 2023; Taori et al., 2023; Wang et al., 2022; Ziegler et al., 2019; Stiennon et al., 2020). Centered around pretrained LLMs, their model variants are widely extended to other domains by finetuning with instruction datasets (Achiam et al., 2023; Reid et al., 2024; Huang et al., 2024; Xu et al., 2023; Liu et al., 2024b; Lin et al., 2023). Instead of finetuning a pretrained LM, multi-modal capacity can be also obtained simultaneously by training a unified model from scratch (Lu et al., 2022, 2023; Luo et al., 2020; Tang et al., 2024; Pan et al., 2023; Jin et al., 2023; Koh et al., 2024). SoupLM proposes to efficiently assemble model variants to deliver a well-generalized one without extra training cost.

5.2 Model Soup

Model soup (weight averaging) is widely used to study optimization process (Ahmadianfar et al., 2022; Bansal et al., 2011). Many works study how it works on improving neural network capacity or analyze the model behavior (Nowlan and Hinton, 2018; Blundell et al., 2015). For large-scale networks, model soup is firstly studied by (Wortsman et al., 2022). It benchmarks the soup method on image classification task on different backbones, and obtain free performance gain with no inference cost, which is critical for large-scale models. Soup strategy also benefits to enhance adapter structure (Chronopoulou et al., 2023), personalized finetuning (Jang et al., 2023), continue training (Akiba et al., 2024), etc, for language models. Different from existing works, our work explores model soup for large language and vision-language models in a cross-domain fashion with more general purposes.

6 Conclusion

We propose SoupLM to first explore the model soup strategy in autoregressive large language models (LLMs) and large multi-modal models (LMMs). This study takes Vicuna and LLaVA as a study case to 1) propose series soup strategies to fully explore the model soup potential pursuing performance gain, 2) statistically benchmark learnable soup capacity across systematically designed configuration space and observe comprehensive hyperparameter patterns, 3) initially probe the soup behavior to observe its consistent property across configurations and regularizations. SoupLM efficiently assembles isomorphical model variants into a well-generalized one that handles multiple domains, with no inference and ignorable training costs. It inspires to fast integrate and iterate large-scale models with multiple domain capacities while avoiding costly additional training efforts.

7 Limitations

We propose SoupLM to merge LLM and LMM into a well-generalized model that handles both language and vision-language domains. However, due to the massive computational requirements to benchmark the model soup for large-scale models, 1) we only take two base models with 7B model size as a study case, which can be easily extended into more general cases, 2) we only provide a heuristic design to benchmark the soup performance on base models, since it is almost not feasible to find the oracle setting among several configuration dimensions. We leave more general studies in our future work.

References

Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, and 1 others. 2023. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*.

Iman Ahmadianfar, Ali Asghar Heidari, Saeed Noshadian, Huiling Chen, and Amir H Gandomi. 2022. Info: An efficient optimization algorithm based on weighted mean of vectors. *Expert Systems with Applications*, 195:116516.

Takuya Akiba, Makoto Shing, Yujin Tang, Qi Sun, and David Ha. 2024. Evolutionary optimization of model merging recipes. *arXiv preprint arXiv:2403.13187*.

Aida Amini, Saadia Gabriel, Peter Lin, Rik Koncel-Kedziorski, Yejin Choi, and Hannaneh Hajishirzi. 2019. Mathqa: Towards interpretable math word problem solving with operation-based formalisms. *arXiv preprint arXiv:1905.13319*.

Jagdish Chand Bansal, PK Singh, Mukesh Saraswat, Abhishek Verma, Shimpi Singh Jadon, and Ajith Abraham. 2011. Inertia weight strategies in particle swarm optimization. In *2011 Third world congress on nature and biologically inspired computing*, pages 633–640. IEEE.

Yonatan Bisk, Rowan Zellers, Jianfeng Gao, Yejin Choi, and 1 others. 2020. Piqa: Reasoning about physical commonsense in natural language. In *Proceedings of the AAAI conference on artificial intelligence*, volume 34, pages 7432–7439.

Charles Blundell, Julien Cornebise, Koray Kavukcuoglu, and Daan Wierstra. 2015. Weight uncertainty in neural network. In *International conference on machine learning*, pages 1613–1622. PMLR.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, and 1 others. 2020. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901.

Jack Choquette. 2023. Nvidia hopper h100 gpu: Scaling performance. *IEEE Micro*.

Alexandra Chronopoulou, Matthew E Peters, Alexander Fraser, and Jesse Dodge. 2023. Adaptersoup: Weight averaging to improve generalization of pretrained language models. *arXiv preprint arXiv:2302.07027*.

Christopher Clark, Kenton Lee, Ming-Wei Chang, Tom Kwiatkowski, Michael Collins, and Kristina Toutanova. 2019. Boolq: Exploring the surprising difficulty of natural yes/no questions. *arXiv preprint arXiv:1905.10044*.

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*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, and 1 others. 2020. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*.

Leo Gao, Jonathan Tow, Baber Abbasi, Stella Biderman, Sid Black, Anthony DiPofi, Charles Foster, Laurence Golding, Jeffrey Hsu, Alain Le Noac'h, Haonan Li, Kyle McDonell, Niklas Muennighoff, Chris Ociepa, Jason Phang, Laria Reynolds, Hailey Schoelkopf, Aviya Skowron, Lintang Sutawika, and 5 others. 2023. A framework for few-shot language model evaluation.

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)*.

Rongjie Huang, Mingze Li, Dongchao Yang, Jiatong Shi, Xuankai Chang, Zhenhui Ye, Yuning Wu, Zhiqing Hong, Jiawei Huang, Jinglin Liu, and 1 others. 2024. Audiogpt: Understanding and generating speech, music, sound, and talking head. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pages 23802–23804.

Joel Jang, Seungone Kim, Bill Yuchen Lin, Yizhong Wang, Jack Hessel, Luke Zettlemoyer, Hannaneh Hajishirzi, Yejin Choi, and Prithviraj Ammanabrolu. 2023. Personalized soups: Personalized large language model alignment via post-hoc parameter merging. *arXiv preprint arXiv:2310.11564*.

Yang Jin, Kun Xu, Liwei Chen, Chao Liao, Jianchao Tan, Bin Chen, Chenyi Lei, An Liu, Chengru Song, Xiaoqiang Lei, and 1 others. 2023. Unified language-vision pretraining with dynamic discrete visual tokenization. *arXiv preprint arXiv:2309.04669*.

Jing Yu Koh, Daniel Fried, and Russ R Salakhutdinov. 2024. Generating images with multimodal language models. *Advances in Neural Information Processing Systems*, 36.

Chunyuan Li, Cliff Wong, Sheng Zhang, Naoto Usuyama, Haotian Liu, Jianwei Yang, Tristan Naumann, Hoifung Poon, and Jianfeng Gao. 2024. Llava-med: Training a large language-and-vision assistant for biomedicine in one day. *Advances in Neural Information Processing Systems*, 36.

Yifan Li, Yifan Du, Kun Zhou, Jinpeng Wang, Wayne Xin Zhao, and Ji-Rong Wen. 2023. Evaluating object hallucination in large vision-language models. *arXiv preprint arXiv:2305.10355*.

Bin Lin, Bin Zhu, Yang Ye, Munan Ning, Peng Jin, and Li Yuan. 2023. Video-llava: Learning united visual representation by alignment before projection. *arXiv preprint arXiv:2311.10122*.

Haotian Liu, Chunyuan Li, Yuheng Li, and Yong Jae Lee. 2023a. Improved baselines with visual instruction tuning.

Haotian Liu, Chunyuan Li, Yuheng Li, and Yong Jae Lee. 2024a. Improved baselines with visual instruction tuning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 26296–26306.

Haotian Liu, Chunyuan Li, Qingyang Wu, and Yong Jae Lee. 2024b. Visual instruction tuning. *Advances in neural information processing systems*, 36.

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

Yuan Liu, Haodong Duan, Yuanhan Zhang, Bo Li, Songyang Zhang, Wangbo Zhao, Yike Yuan, Jiaqi Wang, Conghui He, Ziwei Liu, and 1 others. 2023b. Mmbench: Is your multi-modal model an all-around player? *arXiv preprint arXiv:2307.06281*.

Jiasen Lu, Christopher Clark, Sangho Lee, Zichen Zhang, Savya Khosla, Ryan Marten, Derek Hoiem, and Aniruddha Kembhavi. 2023. Unified-io 2: Scaling autoregressive multimodal models with vision, language, audio, and action. *arXiv preprint arXiv:2312.17172*.

Jiasen Lu, Christopher Clark, Rowan Zellers, Roozbeh Mottaghi, and Aniruddha Kembhavi. 2022. Unified-io: A unified model for vision, language, and multi-modal tasks. In *The Eleventh International Conference on Learning Representations*.

Huaishao Luo, Lei Ji, Botian Shi, Haoyang Huang, Nan Duan, Tianrui Li, Jason Li, Taroon Bharti, and Ming Zhou. 2020. Univl: A unified video and language pre-training model for multimodal understanding and generation. *arXiv preprint arXiv:2002.06353*.

Deepak Narayanan, Mohammad Shoeybi, Jared Casper, Patrick LeGresley, Mostofa Patwary, Vijay Korthikanti, Dmitri Vainbrand, Prethvi Kashinkunti, Julie Bernauer, Bryan Catanzaro, and 1 others. 2021. Efficient large-scale language model training on gpu clusters using megatron-lm. In *Proceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis*, pages 1–15.

Steven J Nowlan and Geoffrey E Hinton. 2018. Simplifying neural networks by soft weight sharing. In *The Mathematics of Generalization*, pages 373–394. CRC Press.

Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, and 1 others. 2022. Training language models to follow instructions with human feedback. *Advances in neural information processing systems*, 35:27730–27744.

Xichen Pan, Li Dong, Shaohan Huang, Zhiliang Peng, Wenhu Chen, and Furu Wei. 2023. Kosmos-g: Generating images in context with multimodal large language models. *arXiv preprint arXiv:2310.02992*.

Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, and 1 others. 2021. Learning transferable visual models from natural language supervision. In *International conference on machine learning*, pages 8748–8763. PMLR.

Alec Radford, Karthik Narasimhan, Tim Salimans, Ilya Sutskever, and 1 others. 2018. Improving language understanding by generative pre-training.

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

Machel Reid, Nikolay Savinov, Denis Teplyashin, Dmitry Lepikhin, Timothy Lillicrap, Jean-baptiste Alayrac, Radu Soricut, Angeliki Lazaridou, Orhan Firat, Julian Schrittwieser, and 1 others. 2024. Gemini 1.5: Unlocking multimodal understanding across millions of tokens of context. *arXiv preprint arXiv:2403.05530*.

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*.

Nisan Stiennon, Long Ouyang, Jeffrey Wu, Daniel Ziegler, Ryan Lowe, Chelsea Voss, Alec Radford, Dario Amodei, and Paul F Christiano. 2020. Learning to summarize with human feedback. *Advances in Neural Information Processing Systems*, 33:3008–3021.

Sainbayar Sukhbaatar, Olga Golovneva, Vasu Sharma, Hu Xu, Xi Victoria Lin, Baptiste Rozière, Jacob Kahn, Daniel Li, Wen-tau Yih, Jason Weston, and 1 others. 2024. Branch-train-mix: Mixing expert llms into a mixture-of-experts llm. *arXiv preprint arXiv:2403.07816*.

Mirac Suzgun, Nathan Scales, Nathanael Schärli, Sebastian Gehrmann, Yi Tay, Hyung Won Chung, Aakanksha Chowdhery, Quoc V Le, Ed H Chi, Denny Zhou, and 1 others. 2022. Challenging big-bench tasks and whether chain-of-thought can solve them. *arXiv preprint arXiv:2210.09261*.

Swabha Swayamdipta, Roy Schwartz, Nicholas Lourie, Yizhong Wang, Hannaneh Hajishirzi, Noah A Smith, and Yejin Choi. 2020. Dataset cartography: Mapping and diagnosing datasets with training dynamics. *arXiv preprint arXiv:2009.10795*.

Zineng Tang, Ziyi Yang, Chenguang Zhu, Michael Zeng, and Mohit Bansal. 2024. Any-to-any generation via composable diffusion. *Advances in Neural Information Processing Systems*, 36.

Rohan Taori, Ishaan Gulrajani, Tianyi Zhang, Yann Dubois, Xuechen Li, Carlos Guestrin, Percy Liang, and Tatsunori B Hashimoto. 2023. Alpaca: A strong, replicable instruction-following model. *Stanford Center for Research on Foundation Models. https://crfm. stanford. edu/2023/03/13/alpaca. html*, 3(6):7.

Tristan Thrush, Ryan Jiang, Max Bartolo, Amanpreet Singh, Adina Williams, Douwe Kiela, and Candace Ross. 2022. Winoground: Probing vision and language models for visio-linguistic compositionality. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5238–5248.

Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, and 1 others. 2023. Llama: Open and efficient foundation language models. *arXiv preprint arXiv:2302.13971*.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *Advances in neural information processing systems*, 30.

Yizhong Wang, Yeganeh Kordi, Swaroop Mishra, Alisa Liu, Noah A Smith, Daniel Khashabi, and Hannaneh Hajishirzi. 2022. Self-instruct: Aligning language models with self-generated instructions. *arXiv preprint arXiv:2212.10560*.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, and 3 others. 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 1 others. 2022. Model soups: averaging weights of multiple fine-tuned models improves accuracy without increasing inference time. In *International conference on machine learning*, pages 23965–23998. PMLR.

Sang Michael Xie, Hieu Pham, Xuanyi Dong, Nan Du, Hanxiao Liu, Yifeng Lu, Percy S Liang, Quoc V Le, Tengyu Ma, and Adams Wei Yu. 2024. Doremi: Optimizing data mixtures speeds up language model pretraining. *Advances in Neural Information Processing Systems*, 36.

Runsen Xu, Xiaolong Wang, Tai Wang, Yilun Chen, Jiangmiao Pang, and Dahua Lin. 2023. Pointllm: Empowering large language models to understand point clouds. *arXiv preprint arXiv:2308.16911*.

Wilson Yan, Yunzhi Zhang, Pieter Abbeel, and Aravind Srinivas. 2021. Videogpt: Video generation using vq-vae and transformers. *arXiv preprint arXiv:2104.10157*.

Xiang Yue, Yuansheng Ni, Kai Zhang, Tianyu Zheng, Ruoqi Liu, Ge Zhang, Samuel Stevens, Dongfu Jiang, Weiming Ren, Yuxuan Sun, and 1 others. 2023. Mmmu: A massive multi-discipline multimodal understanding and reasoning benchmark for expert agi. *arXiv preprint arXiv:2311.16502*.

Rowan Zellers, Ari Holtzman, Yonatan Bisk, Ali Farhadi, and Yejin Choi. 2019. Hellaswag: Can a machine really finish your sentence? In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*.

Renrui Zhang, Jiaming Han, Chris Liu, Peng Gao, Aojun Zhou, Xiangfei Hu, Shilin Yan, Pan Lu, Hongsheng Li, and Yu Qiao. 2023. Llama-adapter: Efficient fine-tuning of language models with zero-init attention. *arXiv preprint arXiv:2303.16199*.

Lianmin Zheng, Wei-Lin Chiang, Ying Sheng, Siyuan Zhuang, Zhanghao Wu, Yonghao Zhuang, Zi Lin, Zhuohan Li, Dacheng Li, Eric Xing, and 1 others. 2024. Judging llm-as-a-judge with mt-bench and chatbot arena. *Advances in Neural Information Processing Systems*, 36.

Lianmin Zheng, Wei-Lin Chiang, Ying Sheng, Siyuan Zhuang, Zhanghao Wu, Yonghao Zhuang, Zi Lin, Zhuohan Li, Dacheng Li, Eric. P Xing, Hao Zhang, Joseph E. Gonzalez, and Ion Stoica. 2023. Judging llm-as-a-judge with mt-bench and chatbot arena. *Preprint*, arXiv:2306.05685.

Deyao Zhu, Jun Chen, Xiaoqian Shen, Xiang Li, and Mohamed Elhoseiny. 2023. Minigpt-4: Enhancing vision-language understanding with advanced large language models. *arXiv preprint arXiv:2304.10592*.

Daniel M Ziegler, Nisan Stiennon, Jeffrey Wu, Tom B Brown, Alec Radford, Dario Amodei, Paul Christiano, and Geoffrey Irving. 2019. Fine-tuning language models from human preferences. *arXiv preprint arXiv:1909.08593*.

A Supplementary Material

A.1 More Implementation Details

Our experiments are conducted on A6000 GPUs. We borrow the code of LLaVA and use its provided model checkpoints for Vicuna and LLaVA base models, and the LLaVA665K instruction dataset. We directly use the training split of meta sets from Huggingface (Wolf et al., 2020). For language evaluation, we leverage on the organized lm-evaluation-harness (Gao et al., 2023) codebase, and for vision-language tasks, we follow the evaluation instruction from LLaVA or use their official evaluation protocols. Our exploration is mainly based on 7B model with their V1.5 version, but it can easily extended to larger model size and other versions of models.

A.2 More Evaluation Performances

Due to the limited space in the main draft, we provide more evaluation performances on language and vision-language domains (**More Evaluations** section in Sec. 3) in Tab. 4

A.3 Complete First Round Ablation Visualizations

We provide complete first round ablation visualizations in Fig. 7. It contains the complete finetuning and test set combinations, which is discussed in the **First Round** section in Sec. 3.

A.4 Complete Second Round Ablation Visualizations

We provide complete second round ablation visualization in Fig. 8. It contains the complete number of samples settings from 10 to 1000, which is discussed in the **Second Round** section in Sec. 3.

The statistical summary of the second round ablation is shown in Fig. 9, used to choose the best hyperparameter combanitions of the second round ablation.

A.5 Complete α Distribution Visualizations

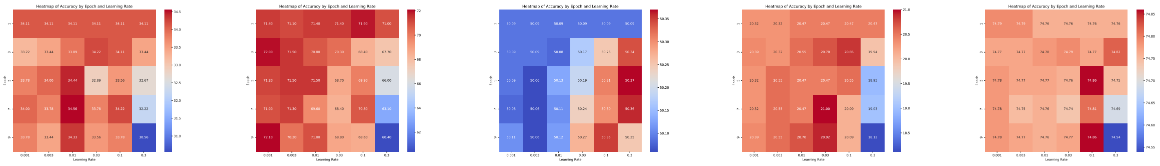
We provide complete α distribution visualizations for different mappings in Fig. 10, Fig. 11, and Fig. 12. They include the mappings of attention, MLP, and normalization blocks, which are discussed in Sec 4. We also include visualizations of other mappings in Fig. 13.

A.6 Complete Regularized α Distribution Visualizations

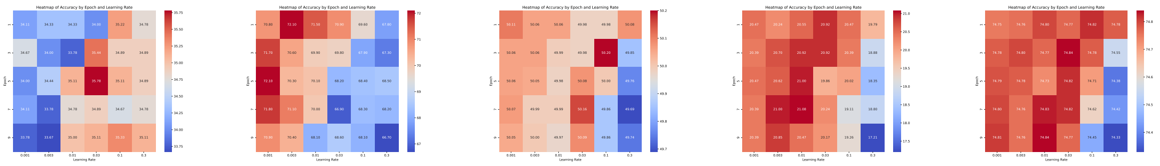
We provide complete regularized α distribution visualizations in Fig. 14, Fig. 15, Fig. 16, Fig. 17. They include 0.0001 and 0.001 regularization magnitudes for attention and MLP blocks, which are discussed in Sec. 4.

Table 4: More evaluations on language and vision-language evaluation benchmarks.

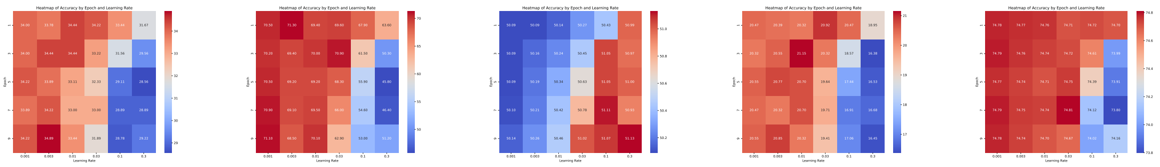
Model	Winogrande	PiQA	MathQA	BoolQA	BBH	POPE	MM-Bench
Vicuna-7B-v1.5	69.46	77.26	27.14	80.95	42.79	80.03	1.98
LLaVA-7B-v1.5	70.64	77.53	**28.11**	81.71	42.14	85.86	**64.69**
Vanilla-Soup ($\alpha^1 = 0.5$)	70.71	**77.80**	27.37	**82.57**	43.51	86.76	62.29
Meta-Soup	**70.72**	77.48	27.27	82.45	**43.96**	**86.90**	61.86



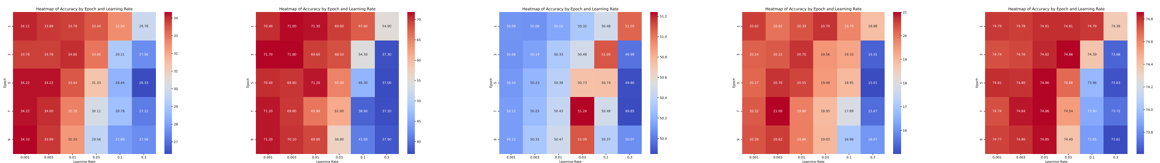
(a) MM-MM (b) MM-L (c) MM-ML (d) MM-G (e) MM-H



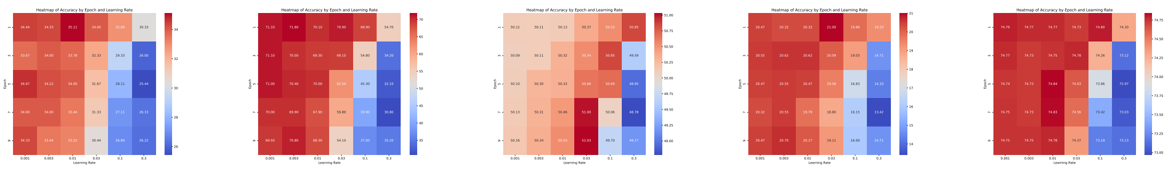
(f) L-MM (g) L-L (h) L-ML (i) L-G (j) L-H



(k) ML-MM (l) ML-L (m) ML-ML (n) ML-G (o) ML-H

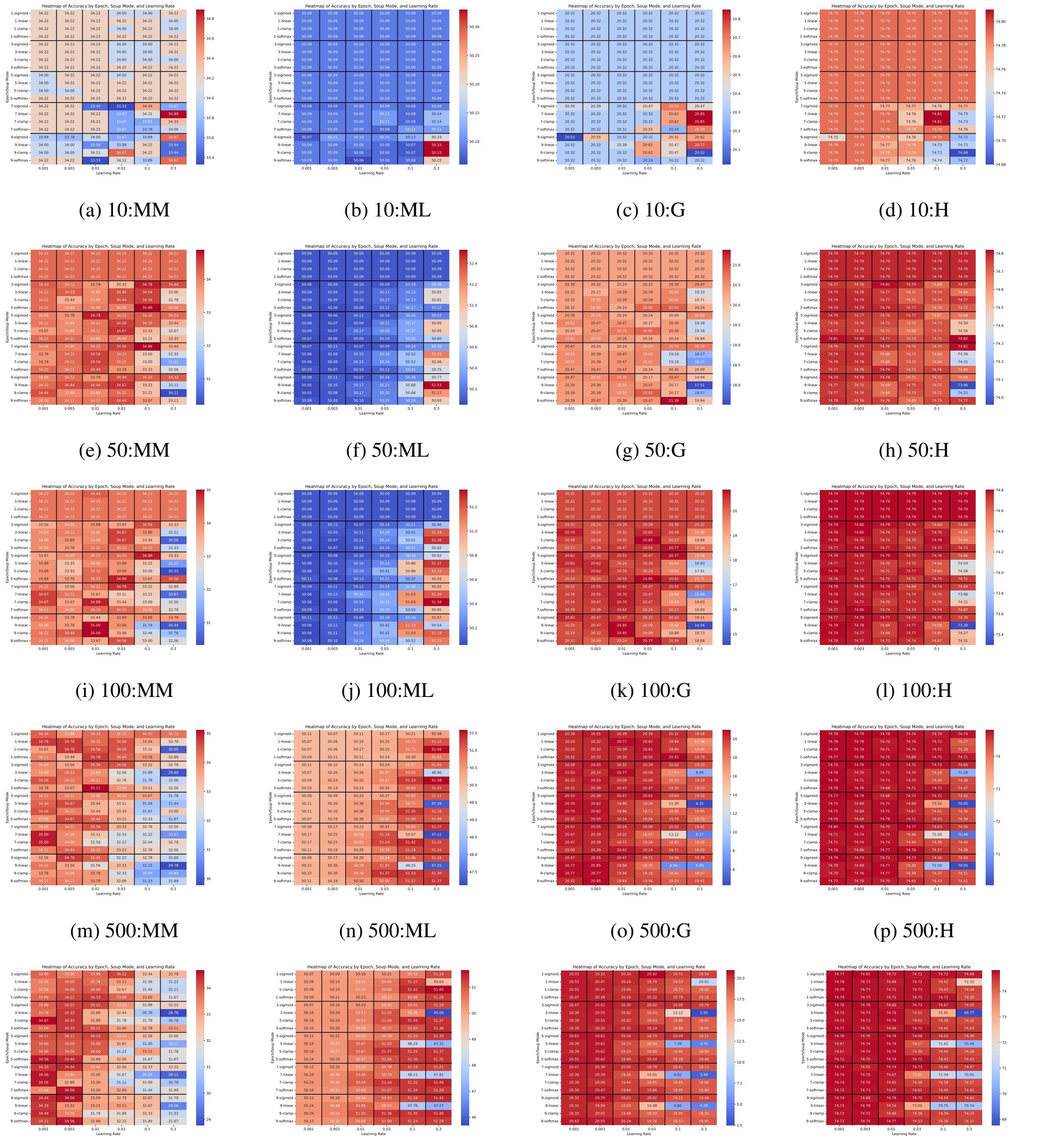


(p) G-MM (q) G-L (r) G-ML (s) G-G (t) G-H



(u) H-MM (v) H-L (w) H-ML (x) H-G (y) H-H

Figure 7: Complete visualization results of the first round ablation for each individual meta set.



(a) 10:MM (b) 10:ML (c) 10:G (d) 10:H

(e) 50:MM (f) 50:ML (g) 50:G (h) 50:H

(i) 100:MM (j) 100:ML (k) 100:G (l) 100:H

(m) 500:MM (n) 500:ML (o) 500:G (p) 500:H

(q) 1000:MM (r) 1000:ML (s) 1000:G (t) 1000:H

Figure 8: Complete visualization results of the second round ablation for number of samples, epochs, learning rates, and soup activation under LLaVA665K-MMLU meta sets.

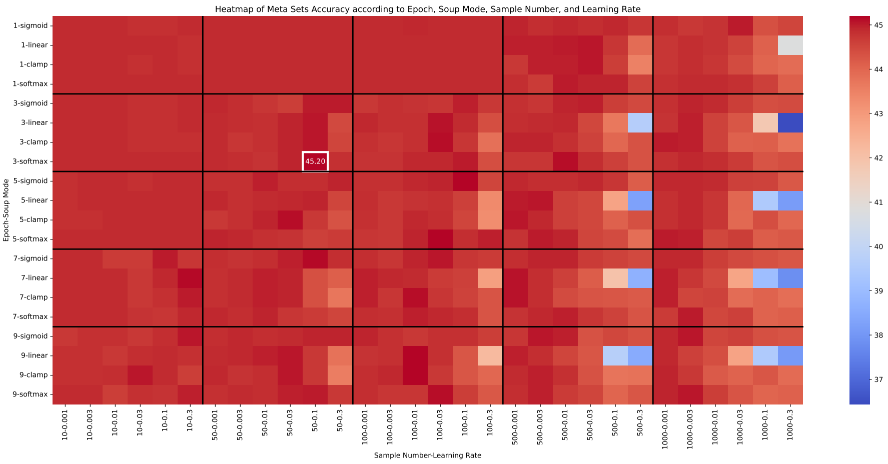
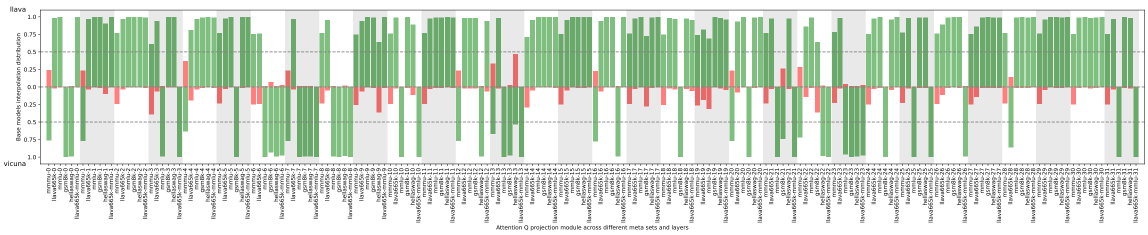
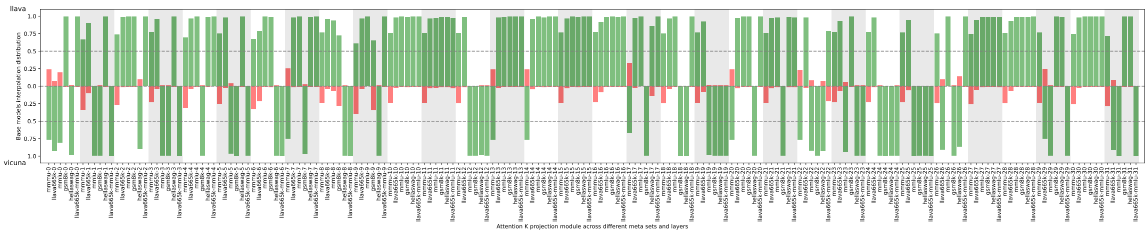


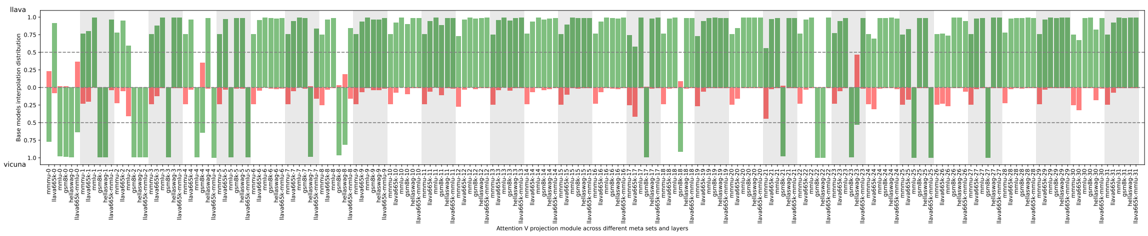
Figure 9: Heatmap visualization of the statistical summary of the second round ablation. The best setting with the highest performance is shown in the white box.



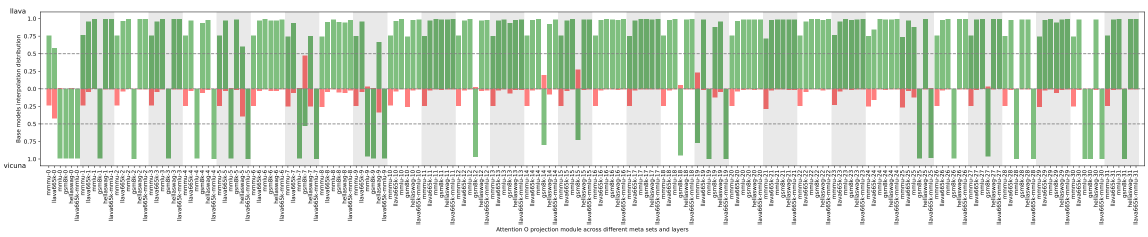
(a) Attention Q mapping.



(b) Attention K mapping.

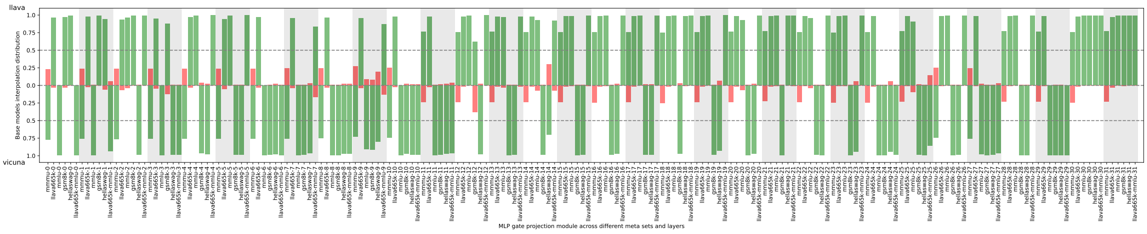


(c) Attnention V mapping.

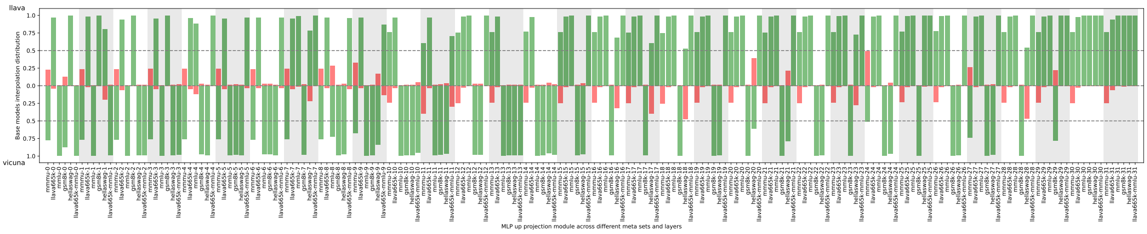


(d) Attention O mapping.

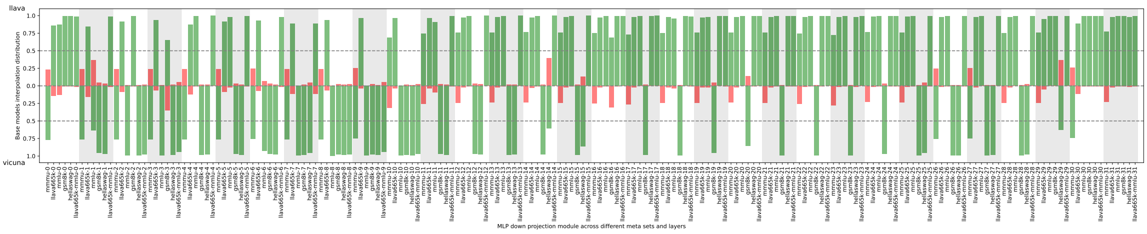
Figure 10: α distribution visualizations for attention.



(a) MLP gate mapping.

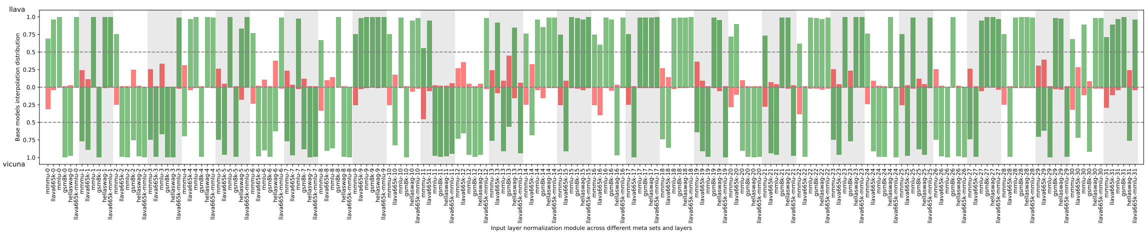


(b) MLP up mapping.

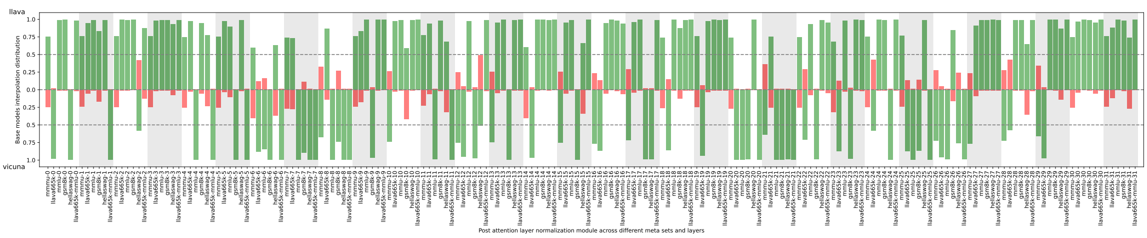


(c) MLP down mapping.

Figure 11: α distribution visualizations for MLP.



(a) Input layernorm.



(b) Post attention layernorm.

Figure 12: α distribution visualizations for layernorm.

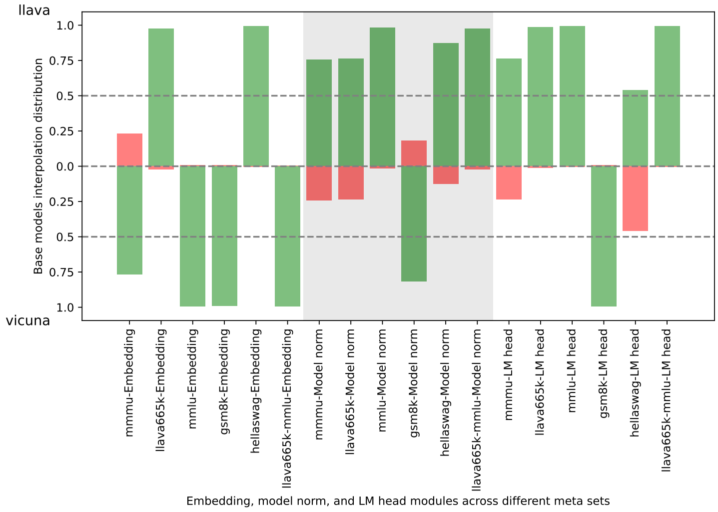
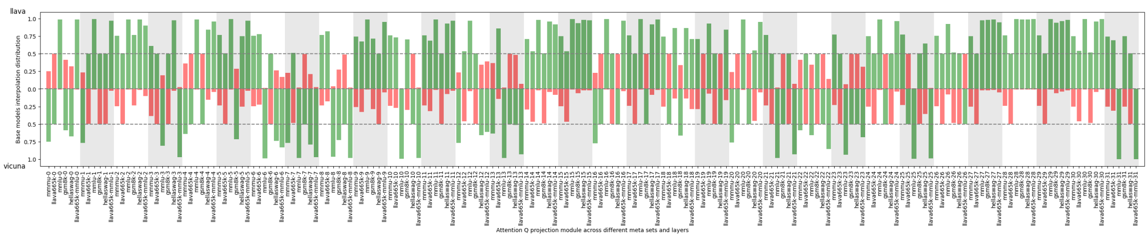
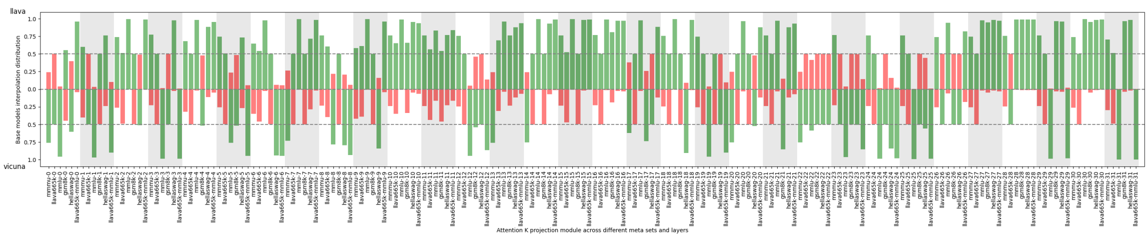


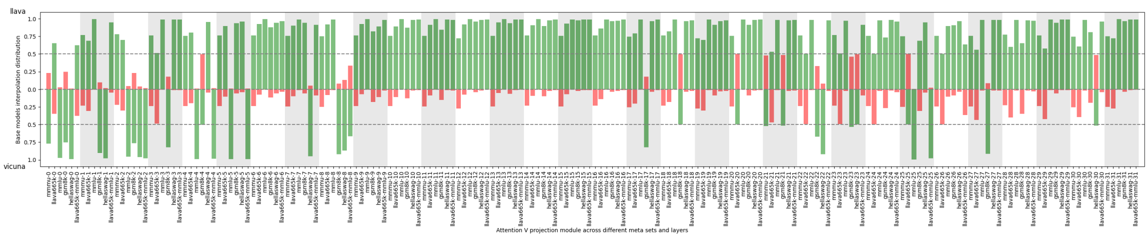
Figure 13: α distribution visualizations of other mappings.



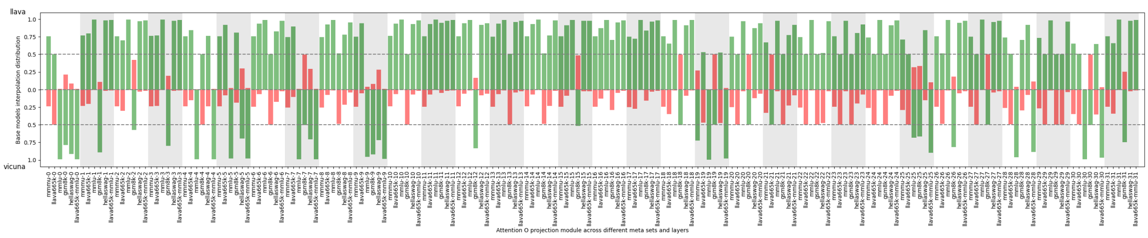
(a) Attention Q mapping.



(b) Attention K mapping.

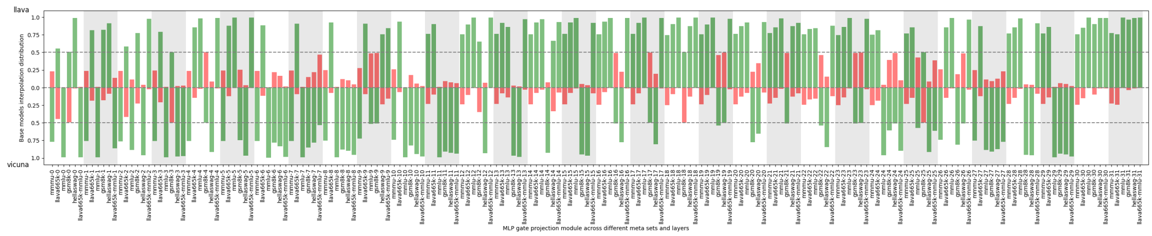


(c) Attnention V mapping.

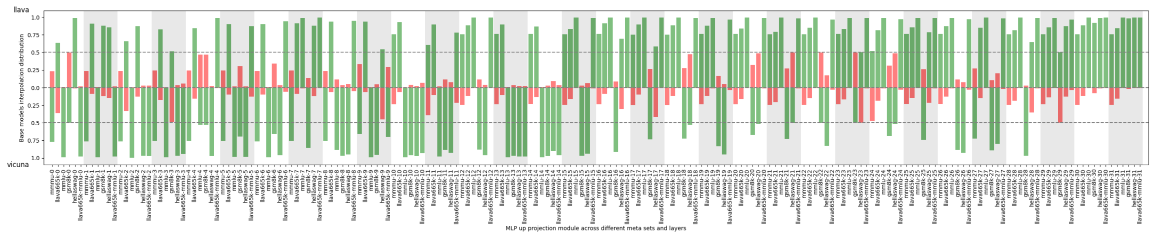


(d) Attention O mapping.

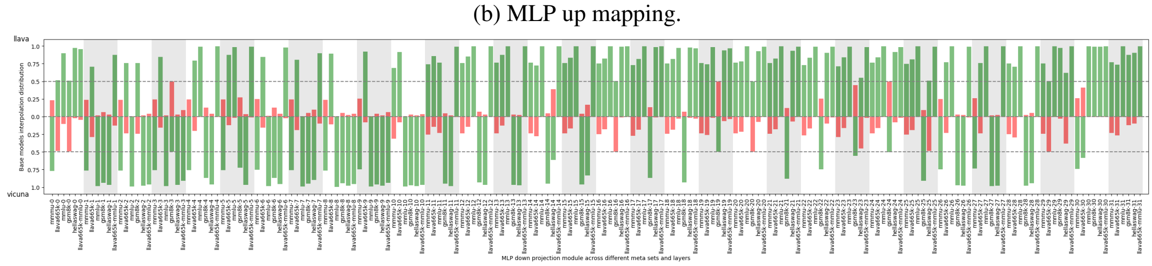
Figure 14: Regularized (0.0001) α distribution visualizations for attention.



(a) MLP gate mapping.

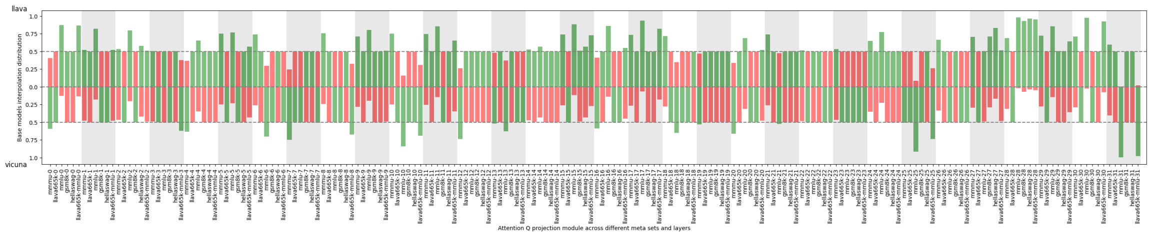


(b) MLP up mapping.

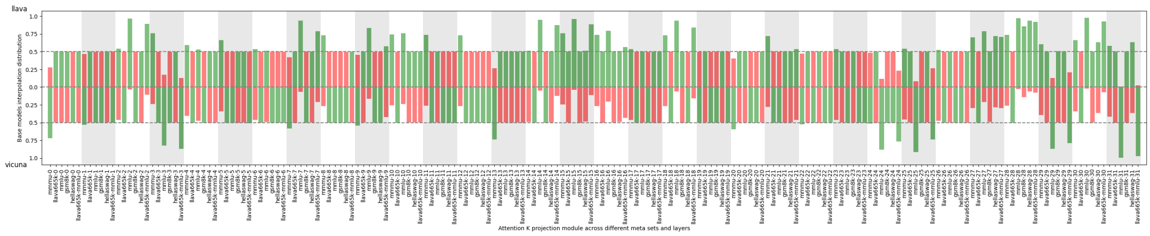


(c) MLP down mapping.

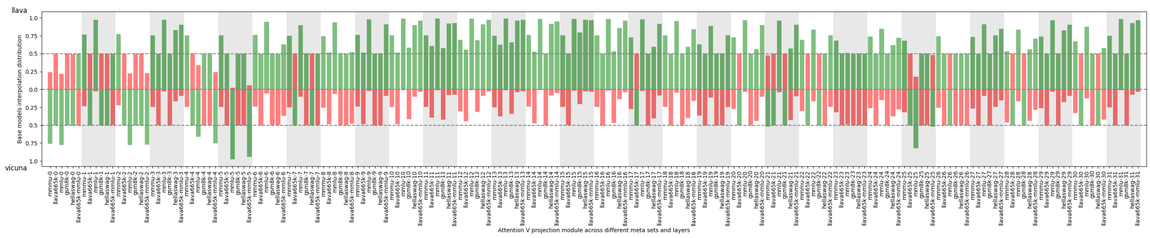
Figure 15: Regularized (0.0001) α distribution visualizations for MLP.



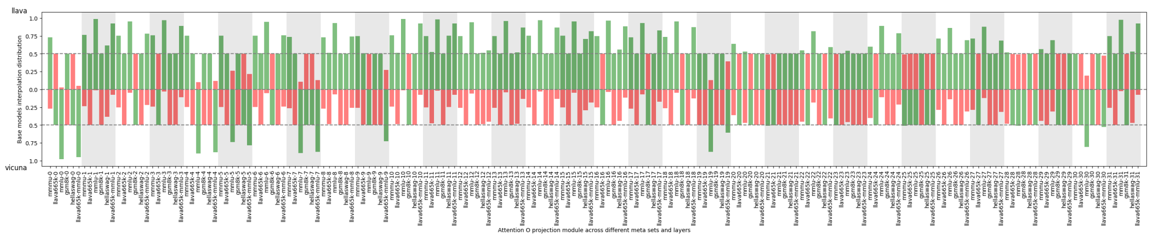
(a) Attention Q mapping.



(b) Attention K mapping.

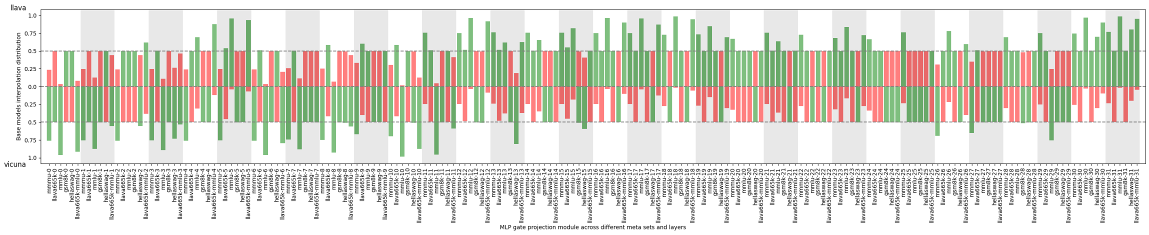


(c) Attnention V mapping.

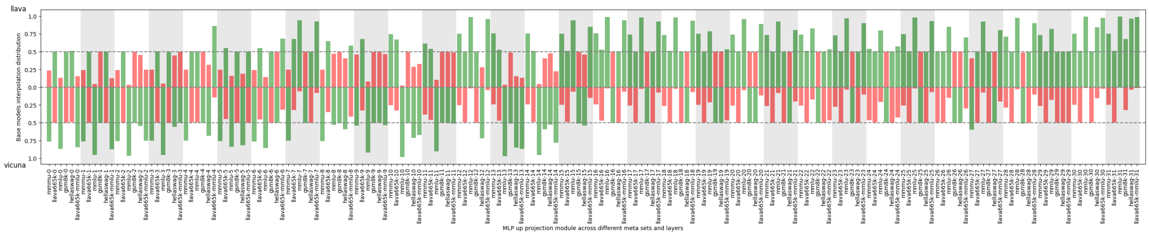


(d) Attention O mapping.

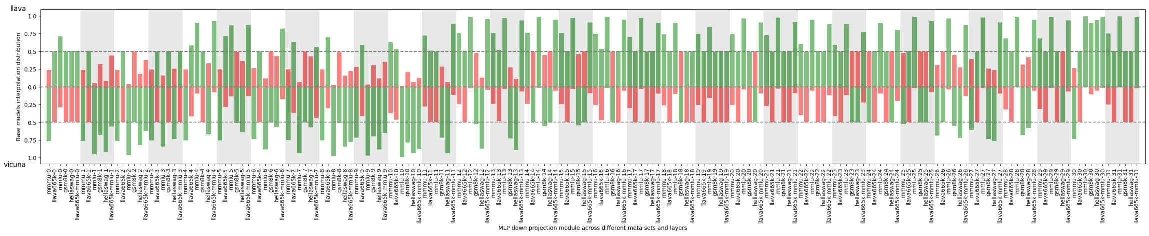
Figure 16: Regularized (0.001) α distribution visualizations for attention.



(a) MLP gate mapping.



(b) MLP up mapping.



(c) MLP down mapping.

Figure 17: Regularized (0.001) α distribution visualizations for MLP.