Activation-Informed Merging of Large Language Models

Amin Heyrani Nobari[*1], **Kaveh Alim**[*1], **Ali ArjomandBigdeli**[2]
Akash Srivastava[3], **Faez Ahmed**[1], **Navid Azizan**[1]

[1]Massachusetts Institute of Technology
[2]Stony Brook University
[3]MIT-IBM Watson AI Lab & Red Hat AI Innovation

Abstract

Model merging, a method that combines the parameters and embeddings of multiple fine-tuned large language models (LLMs), offers a promising approach to enhance model performance across various tasks while maintaining computational efficiency. This paper introduces Activation-Informed Merging (AIM), a technique that integrates the information from the activation space of LLMs into the merging process to improve performance and robustness. AIM is designed as a flexible, complementary solution that is applicable to any existing merging method. It aims to preserve critical weights from the base model, drawing on principles from continual learning (CL) and model compression. Utilizing a task-agnostic calibration set, AIM selectively prioritizes essential weights during merging. We empirically demonstrate that AIM significantly enhances the performance of merged models across multiple benchmarks. Our findings suggest that considering the activation-space information can provide substantial advancements in the model merging strategies for LLMs with up to 40% increase in benchmark performance. Our code is publicly available at https://github.com/ahnobari/ActivationInformedMerging.

1 Introduction

Foundation models are rapidly becoming the dominant force for building Artificial Intelligence (AI) systems. In many cases, researchers build their machine learning models by starting from pre-trained foundation models and fine-tuning (FT) these pre-trained models for some desired target task [27]. In such a paradigm, numerous fine-tuned models are developed for various tasks. However, an important opportunity is missed, as these fine-tuned task-specialized models typically operate in isolation without leveraging the rich features that each possesses [35]. This fact highlights the importance of a growing area of research focused on combining multiple task-specialized models fine-tuned from the same base foundation model.

In particular, as large language models (LLMs) continue to evolve, it becomes increasingly important to develop methods that can effectively fuse the specialized knowledge of various fine-tuned models derived from the same foundation model. Model merging has shown broad applications, including enhancing accuracy and robustness [40], improving generalization [28], multi-modal models [37], and model alignment to human feedback [29, 30]. Given these benefits, a substantial amount of attention has been devoted to developing more effective merging algorithms for LLMs.

In the vast majority of cases, merging LLMs is done using algorithms that explore the weight space of models and do not leverage the information in the activation space. Activation space information has been widely used to develop model pruning and compression methods, both in the context of

[*]Equal Contribution.

39th Conference on Neural Information Processing Systems (NeurIPS 2025).

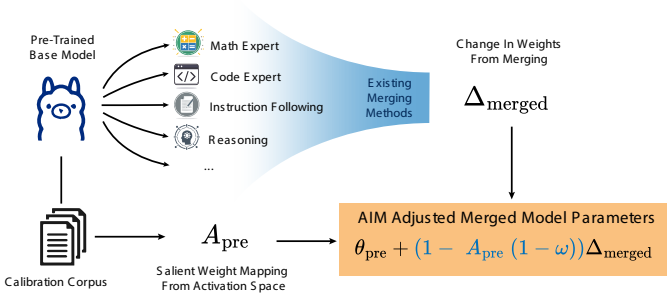

Figure 1: Overview of the proposed activation-informed merging (AIM) in LLMs.

general deep learning methods [12], and more specifically for large language models [23]. However, this direction has remained underexplored for developing more robust merging algorithms. We hypothesize that the activation space information may hold key insights useful for model merging and explore this in our work.

In this paper, we view the problem of merging from a continual learning perspective. Specifically, we explore how FT models and merging them can significantly deviate from the pre-trained base model, potentially leading to overall performance degradation. This is analogous to the common catastrophic forgetting problem in continual learning [10, 35, 39]. Given this perspective, we incorporate the activation space information, taking into account the importance of the base model in preserving its pre-trained capabilities while integrating new knowledge from fine-tuned models. To achieve this, we introduce a new method named Activation-Informed Merging (AIM), which modifies the update step in the merging process to ensure that the most influential weights of the base model, identified through its activations, undergo minimal changes. Figure 1 demonstrates this basic process of AIM.

AIM fundamentally relates to the widely used approach of weight regularization in continual learning [1, 22, 31]. When merging large language models fine-tuned from the same base model, the goal is to maintain the base model's performance while improving the merged model's expertise using the fine-tuned models. Various methods have been proposed to merge fine-tuned LLMs [7, 19, 21, 25, 40, 42, 46, 47]. Although useful, these methods are fragile to outliers and low-quality fine-tuned models and may perform worse than the base model. Hence, we take a new perspective toward merging and adopt the continual learning view to prevent catastrophic forgetting. Our extensive experimental study shows that AIM is a complementary solution that can be applied in conjunction with *any* prior merging methods and consistently improves their performances across all tested benchmarks—including math, code, and instruction following—by up to 40%. Despite its simplicity, AIM shows the effectiveness and importance of the activation space information for more effective model merging.

2 Background and Related Work

LLM Merging. Fine-tuning pre-trained language models has become the prevalent paradigm for building downstream NLP models. Often, fine-tuned models are readily available, but their training data is not due to data privacy or intellectual property concerns [8]. With many fine-tuned checkpoints from the same pre-trained model, LLM merging has emerged as a complementary approach to fine-tuning, combining multiple task-specialized models into a unified architecture. This technique offers several advantages: it reduces storage and inference costs by consolidating expertise into a single model, facilitates compositional generalization by integrating specialized capabilities from different fine-tuned models, and supports decentralized model development by allowing independently trained models to be merged efficiently [43].

Model soup, proposed by Wortsman et al. [40], demonstrates that even simple averaging of the weights of multiple fine-tuned models can enhance accuracy and robustness in image classification and natural language processing applications. Further, Ramé et al. [28] show that model merging can improve out-of-distribution generalization performance. Extending this approach beyond unimodal settings, Sung et al. [37] empirically demonstrate that model merging is also effective in multimodal setups. Beyond generalization benefits, model merging has also been explored for alignment; WARP [30] and WARM [29] introduce weight merging strategies to improve alignment in reinforcement learning from human feedback. WARP demonstrates that merging rewarded policies enhances model quality and alignment, while WARM shows that weight-averaged reward models improve robustness and help mitigate reward hacking.

Recently, many methods have been proposed to go beyond simple weight averaging to merge fine-tuned LLMs into a multitask model by combining their capabilities. Spherical Linear intERPolation (SLERP) [34], originally proposed for animating rotations, which interpolates between two checkpoints, can be seen as a modification to simple weight averaging that finds a spherical path instead of a linear path in models' parameter space. A main shortcoming of this approach is that it does not support merging more than two models. Jang et al. [20] also leverages geometric insights, showing that merging only two fine-tuned models can provide superior in-distribution and out-of-distribution performance compared to ensembling multiple models.

Task Arithmetic [19] generalizes simple weight averaging by introducing task vectors. It suggests moving the base model parameters in the direction of a weighted average of fine-tuned model differences with respect to the base model. Followed by the introduction of task vectors, Model Breadcrumbs [7], Trim, Elect Sign & Merge (TIES merging) [42], and Drop And REscale (DARE) [46] leverage pruning techniques for better and more scalable ways of merging task vectors. WeIght DisENtanglement based merging (WIDEN) [47] takes a more sophisticated approach to model merging by disentangling and analyzing weight components. Another line of work on model merging takes advantage of the information in the model activations of the training data. Matena and Raffel [25] propose Fisher merging, which leverages the Laplace approximation by using the diagonal of each model's Fisher information. Jin et al. [21] attempt to minimize prediction differences between the merged model and the individual models and introduce the Regression Mean (RegMean) method that calculates the optimal weights with respect to Euclidean distance to model predictions.

We refer the reader to Yang et al. [44] for a more comprehensive literature overview. The study provides a detailed discussion of model merging methods and theories, explores their applications in LLMs and multimodal large language models, and highlights future research directions.

Continual Learning. Continual learning strategies can be categorized into five overarching approaches: regularization, where parameters are constrained or updated based on past data; replay, where past data is replayed for the model as it encounters new data; optimization, where the loss function and optimizer are targeted; representation, where new data representations and learned embeddings can be exploited for less forgetting; and architecture, where models and parameters can expand as new data arrives [39]. To avoid catastrophic forgetting of the base model's abilities, we view the model merging problem through the lens of CL, primarily focusing on regularization-based methods. Regularization-based methods penalize deviation from the base model according to some norm [33]. Various methods have been proposed to mitigate catastrophic forgetting, e.g., Aljundi et al. [1], Kirkpatrick et al. [22], Ritter et al. [31]. In particular, Elastic Weight Consolidation (EWC) employs a Fisher Information Matrix to identify and protect parameters crucial for previous tasks by adding a quadratic penalty on the parameter shifts. Integrating CL approaches into the merging framework involves defining a weighted regularization term that selectively constrains parameter updates in critical areas for retaining previously learned tasks. This integration not only mitigates the risk of catastrophic forgetting but also enhances the adaptability and utility of the merged model.

Model Compression. Using activation space information has been shown to be useful in the context of model compression. Frantar and Alistarh [12] show that using a calibration dataset, deep learning models can be quantized and/or pruned efficiently. [23] introduce Activation-aware Weight Quantization (AWQ) for LLM compression and show that protecting only 1% salient weights can greatly reduce quantization error.

Building on these ideas, this paper presents a complementary merging approach that utilizes base model activations and principles from AWQ. Our method efficiently sketches delta parameters, ensuring the base model retains its original capabilities while incorporating expertise from fine-tuned models.

3 Methodology: Activation-Informed Merging

As discussed in Section 2, most existing approaches for merging FT LLMs primarily focus on the weight space of the models being merged. However, it is well established that the activation space of these models contains crucial insights into the degree of importance of different parameters of LLMs. This was shown to be the case, for instance, in the work done by Lin et al. [23] on Activation-aware Weight Quantization (AWQ), outperforming traditional quantization methods by including insight from the activation space of LLMs. Given this, we hypothesize that the activation space of LLMs

likely holds useful clues for model merging as well. Inspired by AWQ, we introduce Activation-Informed Merging (AIM) for merging FT LLMs. In this section, we detail our proposed solution and discuss some of the inner workings of AIM.

3.1 The Merging Problem and Connections to Continual Learning

Consider the merging of N models with parameters $\theta_1, \theta_2, \cdots, \theta_N$ fine-tuned on different tasks from a common pre-trained model with parameters θ_{pre}. For each fine-tuned LLM, we are essentially creating experts on specific tasks that move away from the generalist pre-trained model, hence usually degrading performance in some tasks while improving performance on the task for which the model is fine-tuned. In this sense, each FT model with parameters θ_n can be seen as a model fitted to a new task $\mathcal{D}_n = \{X_n, Y_n\}$ in a continual learning scenario with the potential for catastrophic forgetting on the generalist pre-trained model, which may not perform as well on the specific task but will have a more balanced performance across various tasks. As such, we hypothesize that when merging FT LLMs adapted from the same base model, emphasis on the base model can build better robustness to large performance degradation across numerous tasks while still allowing for capturing each FT expert's capabilities. AIM seeks to achieve this by relaxing the changes to the salient weights of the base model in the final merged model. In this way, AIM is analogous to weight regularization in many continual learning approaches [1, 22, 31, 33, 39]. Notably, the saliency of weights is determined by analyzing the activation space (sensitivities) of the base model rather than just regularizing based on the weight space, similar to [10, 26].

3.2 Activation Space Analysis

AIM determines the saliency of a model's weights by looking at the scale of activations by passing a calibration dataset to the model and recording the scale of activations in each channel. To better understand why, we will analyze how perturbation of the weights of a given model affects the model outputs. Let the original weights be $w \in \mathbb{R}^{N\times M}$ and the perturbation be $\delta w \in \mathbb{R}^{N\times M}$, such that the perturbed weights are $w' = w + \delta w$. The output of a linear layer with input $x \in \mathbb{R}^{1\times N}$ and perturbed weights is

$$y' = xw' = x(w + \delta w) = xw + x(\delta w). \tag{1}$$

The magnitude of the error due to this perturbation, Error $= y' - y = x(\delta w)$, scales with the magnitude of activation x:

$$\|\text{Error}\|_p = \|\operatorname{diag}(x)\delta w\|_p \leq \|\operatorname{diag}(x)\delta w\|_1 \leq \sum_{i=1}^{N} |x_i| \sum_{j=1}^{M} |\delta w_{ij}|. \tag{2}$$

For any specific input channel x_i, the error contribution from perturbation in the i-th row of w, δw_i is amplified by the magnitude of the same channel in the input. As such, one could selectively regularize the weights based on the importance of the input channels, i.e., their magnitudes. In this way, we use a calibration dataset to capture the average magnitude of the input channels for each layer of the base model and determine the saliency of weights in the base model from the activation space.

3.3 Calibration Data and Robustness to Calibration Data

We choose the calibration dataset to be a subset of the validation data from the pile dataset [13], which is similar in distribution to most pre-training datasets. Most notably, this dataset has been utilized in model compression for both quantization in AWQ [23] and pruning in WANDA [36]. This calibration dataset is considered to be fairly diverse and not task-specific, which should allow us to quantify weights' saliency in a robust manner.

Robustness to Calibration Data The use of activation spaces from calibration data has been thoroughly studied in model compression, by Lin et al. [23] and Sun et al. [36]. In both cases, authors conduct extensive studies on the robustness of their methods, which, like ours, rely on the magnitude of activations, and both concur in two important findings. 1) When using calibration data to quantify activation space magnitudes to be used by algorithms for model quantization or pruning, performance is robust to dataset quality. This is in contrast to methods that require fine-tuning or retraining after compression. This observation is made clear in both studies and confirms that sensitivity to the quality of the data is much less in methods that require only activation space analysis without

training. 2) Most notably, ablation studies on the size of the calibration set also show robustness to the size of calibration data, with both studies confirming that only 8-32 sample sequences of length approximately 2048 tokens are enough for model compression algorithms to produce robust outcomes [23, 36]. Despite this robustness, we use the same 256 total sequences (approximately 524K tokens) that the authors of both studies use in their main experiments; however, this robustness to dataset size is noted as an important tool for reducing computational cost and time for running algorithms such as ours. We also study this matter in an ablation study and show that, like those studies, AIM is also robust to dataset size and can be made much more efficient if need be.

3.4 Adaptable Relaxation Scheme

As discussed, we introduce an adaptable relaxation scheme based on the activations of the base model, which we wish not to stray away from significantly. To make the scheme adaptable to any merging algorithm in the weight space, we formulate our relaxation scheme in terms of the changes in the weights. Given a task-agnostic representative calibration corpus D. We can pass this corpus through the model and accumulate the activations from each token in the dataset. This will yield the average magnitude of activations for each channel in all layers of the model. As previously noted, averaging the inputs over the calibration set yields a vector $x \in \mathbb{R}^{1\times N}$ for each linear layer. Constructing a diagonal matrix from the absolute values of this vector gives $\mathrm{diag}(|x|) \in \mathbb{R}^{N\times N}$. By normalizing $\mathrm{diag}(|x|)$ using its maximum value, we obtain a diagonal matrix $A_{\text{pre}} \in [0,1]^{N\times N}$. This matrix serves to modulate weight updates based on input saliency, allowing for consistent control of merging through a universal hyperparameter ω. Next, we define the action of any given merging method by the changes that it applies to the model weights with respect to each of the fine-tuned models being merged (i.e., $\theta_1, \theta_2, \ldots, \theta_n$). Specifically, we denote the weight update contributed by a fine-tuned model (with parameters θ_i) to the model parameters by Δ_i (e.g., $\Delta_i = \frac{\theta_i - \theta_{\text{pre}}}{n}$ for weight averaging). Now, we propose an adaptive relaxation scheme that adjusts the final model. For simplicity of notation, let θ refer to weights of a linear layer (i.e., an $N \times M$ matrix), then the AIM relaxation scheme can be written as:

$$\delta w_{\text{AIM}} = \theta_{\text{merged}} - \theta_{\text{pre}} = (1 - A_{\text{pre}}(1-\omega)) \sum_{i=1}^{N} \lambda_i \Delta_i, \tag{3}$$

where δw_{AIM} is the relaxation change and the subscript *pre* refers to the pre-trained model and ω is the relaxation factor that controls how much relaxation is applied (an ω of 0.0 would revert the most salient weight to the base weights and an ω of 1.0 applies no relaxation), effectively ω is scaling error/deviation from the base model, and λ_i are the weight factors for each Δ_i. Note that λ_i is internal to each merging algorithm and not part of the AIM relaxation; however, in general, one could selectively apply this if desired. In this work, since we do not explicitly look at the merging algorithm's inner workings, we can simply fuse the terms $\sum_{i=1}^{N} \lambda_i \Delta_i$ into a single algorithm-agnostic term Δ_{merged} and simplify the relaxation scheme to:

$$\theta_{\text{AIM}} = \theta_{\text{pre}} + (1 - A_{\text{pre}}(1-\omega))\Delta_{\text{merged}}. \tag{4}$$

Note that in general A_{pre} is not a single matrix, rather a mapping of activations for all model parameters obtained as we described prior. In our experiments, we apply this relaxation scheme to several different merging methods and explore how the hyperparameter ω affects the merged model's behavior, and present these results in Section 5.

Sensitivity-Based Formulation An alternative way of choosing the importance scores for changing the model weights—which has been used, e.g., in continual learning [10, 26], out-of-distribution detection [32], and meta-learning [2]—is to use the partial derivatives of the (base) model with respect to the parameters, i.e., *sensitivities*, which correspond to a scaled version of the activations. This is in contrast with using the activations directly, common in model compression. Formally, let f_θ denote the model with parameters θ, and $\theta[j]$ be one of the weights; then $g_{\theta[j]} = |\frac{\partial f}{\partial \theta[j]}|$ determines how sensitive the output is to perturbing the weight $\theta[j]$. We develop a sensitivity-based formulation of AIM by incorporating sensitivity scores calculated from the gradients and replacing activations with sensitivity scores. We consider $f_\theta(x)$ to be the logits of the model with parameter θ and $\mathcal{L}$ to be the entropy function, similar to Farajtabar et al. [10]'s suggested approach for classification problems. Now let $G_{\text{pre}} = |\frac{\partial \mathcal{L}}{\partial \theta}|$ be the magnitude of the gradient for all parameters of the pretrained model for samples in a calibration corpus D. Then the sensitivity-based formulation can be written as follows:

$$\theta_{\text{AIM,G}} = \theta_{\text{pre}} + (1 - G_{\text{pre}}(1 - \omega))\Delta_{\text{merged}}. \quad (5)$$

In other words, the sensitivity-based formulation of the problem suggests that regularization of weights through relaxation should be done based on model gradients. However, we note that computing gradients for a calibration corpus can be computationally expensive and will require significantly more memory and compute resources than storing activations, which is akin to performing inference on the model. To verify that using activations retains the same performance and fidelity as this formulation, we conduct our experiments for both AIM and the sensitivity-based formulation. We present the results of the continual learning view in Appendix C, and we see that when comparing results of AIM (Table 1) and the sensitivity-based formulation (Table 4), the performance boost of both approaches is very similar, with AIM being computationally more efficient.

4 Experimental Setup and Evaluation Metrics

We conduct two separate experiments with AIM: 1) we apply AIM to 5 different merging methods including the two latest works on the topic with different numbers of experts being merged and report the performance of the models on 6 different benchmarks; 2) we conduct an ablation study on the ω parameter in AIM and analyze how ω affects each of the merging methods in a scenario where 3 different experts are being merged.

4.1 Selection of FT Expert LLMs

To understand how AIM reacts with different merging methods, we conduct experiments with merging different experts fine-tuned from the same base model. The set of experts we use is the same set of experts used by the two latest LLM merging algorithms in the literature, namely DARE [46] and WIDEN [47], which use the same three experts fine-tuned from Llama-2-13b [38]. These experts include the WizardLM-13B [41] model fine-tuned for instruction following, WizardMath-13B [24] fine-tuned for superior mathematical reasoning, and llama-2-13b-code-alpaca, which serves as the code expert [4]. See Appendix A for more details.

4.2 Merging Methods Implementations

In our experiments, we implement the latest merging methods in the literature for LLM merging. These include newly developed DARE and WIDEN [46, 47] methods as well as some of the long-established approaches of TIES merging [42], and task arithmetic [19]. For all merging methods except WIDEN, we use the comprehensive MergeKit implementations developed by Goddard et al. [15], and for WIDEN, we use the publicly available implementation provided by the authors of the paper.

We note that in many of the merging algorithms, many hyperparameters can be adjusted. In these cases, we use the author-recommended values where available and the default parameters recommended by Goddard et al. [15]. Note that it is possible to perform a grid search on these hyperparameters to find optimal values for each benchmark; however, this would essentially be over-fitting on benchmarks and does not provide any value to our analysis of the proposed complementary relaxation scheme, which applies adjustments to the merged models. For reproducibility, all of our checkpoints and code to reproduce the results will be made publicly available.

4.3 Benchmarks Used For Evaluations

Given that the expert models we use in our experiments involve fine-tuning on instruction following, mathematical reasoning, and code generation, we use several common benchmarks for each of these tasks. Specifically, we measure model performance on language understanding with the MMLU [16] benchmark, instruction following with IFEval [48] benchmark, code generation with HumanEval [5] and MBPP [3] benchmarks, and mathematical reasoning with the MATH [17] and GSM8K [6] benchmarks. For all benchmarks, we use the latest versions and up-to-date implementations developed by Gao et al. [14] except for mathematical reasoning, for which we use the chain of thought prompting used by Luo et al. [24] to replicate the results of the original model as closely as possible. The code we use for these benchmark results will also be made publicly available for reproducibility.

In addition to the common benchmarks that we use to evaluate merged models, we also propose a new evaluation metric for LLM merging (or merging of different experts in general), which we believe helps better contextualize the value added by any given merging algorithm and which we discuss in the following section.

4.4 Measuring Performance From an Optimization Perspective

We note that in most cases, LLMs are not meant to operate as narrow expert models, unlike a large portion of deep learning applications where models are trained to perform very specific tasks such as classification or regression. LLMs, in contrast, are generalist language models aiming to assist across a wide variety of tasks and applications. As such, LLMs can be viewed from a multi-objective optimization perspective. In merging scenarios specifically, multiple expert models are brought together to create a merged model aiming to find the balance of performance across the different expertise of the fine-tuned models. In this sense, each expert can be thought of as optimized for a specific objective. This perspective lends itself rather well to a multi-objective optimization view of the problem. Given this, only looking at how each model performs on each of the benchmarks does not give us a full picture of the multi-objective goal of merging.

To obtain a more comprehensive view of merging performance, we propose a **hypervolume-based metric** that quantifies the contribution of the merged model to the **multi-objective frontier** of FT LLMs. Consider a performance space defined over N benchmarks, where each model's performance is represented as a point in an N-dimensional space. The performance on each benchmark is normalized to the range $[0, 1]$, where 0 represents the worst-case (reference point), and 1 represents the best possible performance with 100% accuracy on the benchmark in question.

Let $\mathbf{r} = (r_1, r_2, \ldots, r_N)$ denote the reference point in this space, which we set to $(0, 0, \ldots, 0)$ to ensure hypervolume calculations are consistently defined. Given a set S of FT models and the pre-trained base model, let $S^* \subseteq S$ denote the subset of models that are **Pareto-optimal**, i.e., models that are not dominated by any other model in S. The hypervolume of this set, denoted as $\mathrm{HV}(S^*)$, is defined as:

$$\mathrm{HV}(S^*) = \lambda \left(\bigcup_{\mathbf{x} \in S^*} \mathrm{dom}(\mathbf{x}, \mathbf{r}) \right), \tag{6}$$

where $\lambda(\cdot)$ denotes the Lebesgue measure (i.e., volume in $\mathbb{R}^N$), and $\mathrm{dom}(\mathbf{x}, \mathbf{r})$ represents the hypervolume dominated by $\mathbf{x}$ with respect to the reference point $\mathbf{r}$.

When a merged model M is introduced, the new set becomes $S' = S \cup \{M\}$, and the updated Pareto-optimal set is denoted as S'^*. Given this set including the merged model, we can measure the added value of the merged model from a multi-objective perspective as the normalized hypervolume gain (HV Gain) as a result of adding this merged model:

$$\text{HV Gain} = \sqrt[d]{\mathrm{HV}(S'^*) - \mathrm{HV}(S^*)} \tag{7}$$

Where d is the number of dimensions/benchmarks. Since hypervolume is computed only over Pareto-optimal models, we have $\mathrm{HV}(S'^*) \geq \mathrm{HV}(S^*)$, ensuring that HV Gain ≥ 0. This metric provides an aggregated measure of merging effectiveness, capturing trade-offs across multiple benchmarks rather than focusing on isolated improvements, thus providing a full picture of merging performance. In our experiments, we track *HV Gain* along with the 6 aforementioned benchmarks.

5 Experiments and Results

In this section, we present our results on applying AIM to different merging methods as well as an ablation study on how the ω hyperparameter affects the performance of each merging method. All experiments are run using 4 H100 GPUs, and each set of benchmarks takes roughly 15 minutes to run for each checkpoint.

5.1 AIM Applied to Various Merging Approaches

To demonstrate the effectiveness of AIM, we conduct experiments on 5 different merging methods under 4 different scenarios. As mentioned before, we use 3 different FT LLM experts in our experiments. As such, we merge models using each merging method for all 4 possible permutations

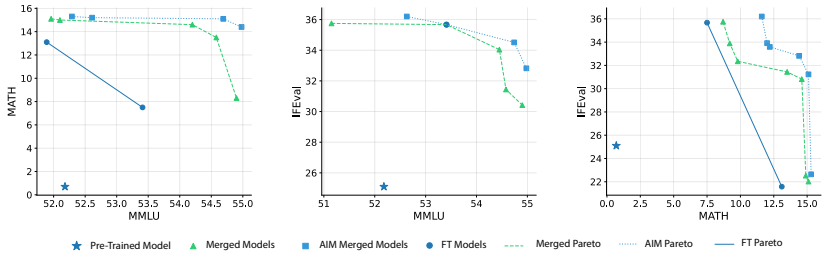

Figure 2: The Pareto fronts of models under different scenarios. Note that the points in these plots represent all models benchmarked in Table 1, for better readability, we only visualize the dominating points in each case. The measured increases in HV Gain when AIM is applied can be clearly seen in the Pareto frontier shifting further forward when AIM is applied compared to when only a population of merged models is evaluated.

Table 1: Benchmark Results Across Various Merging Methods. Percentage changes are shown relative to models merged without AIM. The highest-performing fine-tuned and base models are highlighted in yellow, and the best-performing merged models are marked in blue. The results demonstrate that applying AIM significantly enhances the performance of merged models.

Method	Model(s)	AIM	HumanEval	MBPP	MMLU	MATH	GSM8K	IFEval	HV Gain
Base Models									
-	Base	-	17.07	27.80	52.18	0.70	4.20	25.10	-
-	Code	-	17.07	31.60	52.91	6.00	24.10	26.25	-
-	Instruction Tuned	-	26.83	34.80	53.41	7.50	43.40	35.67	-
-	Math	-	15.24	27.60	51.89	13.10	59.10	21.58	-
Merged Models									
DARE Task Arithmetic	Code + Instruction Tuned	No	26.83	34.40	53.53	8.40	45.80	33.42	0.27
		Yes	29.27 (+9.09%)	36.00 (+4.65%)	54.18 (+1.21%)	8.30 (-1.19%)	46.20 (+0.87%)	32.00 (-4.25%)	0.28 (+2.49%)
	Code + Math	No	16.46	28.60	51.96	15.10	64.70	22.02	0.23
		Yes	15.85 (-3.71%)	29.60 (+3.50%)	52.50 (+1.04%)	14.80 (-1.99%)	64.10 (-0.93%)	21.91 (-0.50%)	0.23 (-1.65%)
	Instruction Tuned + Math	No	5.49	19.00	51.08	9.80	54.30	32.35	0.18
		Yes	12.20 (+122.22%)	28.20 (+48.42%)	52.72 (+3.21%)	12.90 (+31.63%)	62.20 (+14.55%)	31.96 (-1.21%)	0.26 (+40.71%)
	Code + Instruction Tuned + Math	No	11.59	19.60	50.89	9.10	49.70	33.20	0.16
		Yes	15.85 (+36.76%)	27.00 (+37.76%)	52.59 (+3.34%)	12.20 (+34.07%)	60.70 (+22.13%)	33.59 (+1.17%)	0.23 (+40.59%)
DARE Ties	Code + Instruction Tuned	No	30.49	35.20	53.40	8.60	46.20	33.28	0.28
		Yes	30.49	36.80 (+4.55%)	54.02 (+1.16%)	8.60	47.20 (+2.16%)	33.16 (-0.36%)	0.29 (+1.63%)
	Code + Math	No	17.07	27.40	51.92	14.90	63.60	22.53	0.23
		Yes	17.68 (+3.57%)	29.00 (+5.84%)	52.61 (+1.33%)	15.20 (+2.01%)	63.90 (+0.47%)	21.10 (-6.35%)	0.24 (+4.00%)
	Instruction Tuned + Math	No	8.54	23.80	51.39	9.20	54.10	33.89	0.20
		Yes	15.85 (+85.60%)	30.20 (+26.89%)	52.89 (+2.92%)	11.60 (+26.09%)	57.80 (+6.84%)	35.63 (+5.13%)	0.26 (+31.22%)
	Code + Instruction Tuned + Math	No	13.41	21.20	51.15	8.70	51.50	35.75	0.17
		Yes	19.51 (+45.49%)	28.60 (+34.91%)	52.63 (+2.89%)	11.60 (+33.33%)	57.00 (+10.68%)	36.20 (+1.26%)	0.24 (+41.28%)
Task Arithmetic	Code + Instruction Tuned	No	29.27	33.80	53.44	8.60	47.10	31.60	0.28
		Yes	29.88 (+2.08%)	35.80 (+5.92%)	54.12 (+1.27%)	7.80 (-9.30%)	46.60 (-1.06%)	32.01 (+1.30%)	0.28 (+0.61%)
	Code + Math	No	18.29	28.60	52.10	15.00	64.70	21.92	0.24
		Yes	17.68 (-3.34%)	29.20 (+2.10%)	52.52 (+0.81%)	14.60 (-2.67%)	64.50 (-0.31%)	21.54 (-1.73%)	0.24 (-2.65%)
	Instruction Tuned + Math	No	4.27	20.20	51.50	10.00	54.20	31.31	0.18
		Yes	8.54 (+100.00%)	26.40 (+30.69%)	52.83 (+2.58%)	12.80 (+28.00%)	61.30 (+13.10%)	32.62 (+4.18%)	0.24 (+34.52%)
	Code + Instruction Tuned + Math	No	11.59	19.60	51.20	9.00	52.70	32.87	0.16
		Yes	15.24 (+31.49%)	27.40 (+39.80%)	52.63 (+2.79%)	12.00 (+33.33%)	58.10 (+10.25%)	33.91 (+3.16%)	0.22 (+31.97%)
Ties Merging	Code + Instruction Tuned	No	16.46	23.60	52.70	2.70	5.40	24.48	0.00
		Yes	15.24 (-7.41%)	24.20 (+2.54%)	53.15 (+0.85%)	2.60 (-3.70%)	5.20 (-3.70%)	22.87 (-6.58%)	0.05 (+inf%)
	Code + Math	No	15.85	26.80	51.86	14.30	62.60	21.63	0.20
		Yes	15.85	28.60 (+6.72%)	52.29 (+0.83%)	15.30 (+6.99%)	63.80 (+1.92%)	22.64 (+4.67%)	0.23 (+13.55%)
	Instruction Tuned + Math	No	28.05	34.60	54.45	8.70	44.70	34.04	0.23
		Yes	27.44 (-2.17%)	35.00 (+1.16%)	54.74 (+0.53%)	9.30 (+6.90%)	46.10 (+3.13%)	34.51 (+1.38%)	0.25 (+6.38%)
	Code + Instruction Tuned + Math	No	21.34	29.20	53.97	6.30	29.20	26.95	0.11
		Yes	20.73 (-2.86%)	29.20	54.46 (+0.91%)	5.70 (-9.52%)	23.70 (-18.84%)	25.98 (-3.60%)	0.11 (+4.33%)
WIDEN	Code + Instruction Tuned	No	26.22	35.60	54.90	8.30	45.00	30.42	0.27
		Yes	25.61 (-2.33%)	34.60 (-2.81%)	54.97 (+0.13%)	8.20 (-1.20%)	44.10 (-2.00%)	31.60 (+3.88%)	0.26 (-0.93%)
	Code + Math	No	17.07	29.40	53.35	14.20	64.40	24.02	0.24
		Yes	17.07	29.60 (+0.68%)	53.36 (+0.02%)	14.30 (+0.70%)	62.20 (-3.42%)	23.95 (-0.29%)	0.24 (-1.22%)
	Instruction Tuned + Math	No	24.39	30.40	54.20	14.60	66.00	30.82	0.30
		Yes	23.78 (-2.50%)	32.00 (+5.26%)	54.69 (+0.90%)	15.10 (+3.42%)	68.20 (+3.33%)	31.23 (+1.33%)	0.31 (+2.54%)
	Code + Instruction Tuned + Math	No	25.00	33.20	54.58	13.50	64.20	31.44	0.29
		Yes	26.83 (+7.32%)	32.80 (-1.20%)	54.98 (+0.73%)	14.40 (+6.67%)	64.00 (-0.31%)	32.82 (+4.39%)	0.30 (+4.70%)

of these expert LLMs. Then we apply AIM to all merged models and measure the performance of each model in all 6 benchmarks. We also report the HV gain for each merged model compared to the population of the base model and the models being merged (in cases with 2 models, the population will only include the models used for merging). These results are presented in Table 1. For this experiment, we used $\omega = 0.4$, which we found to be the best balance of performance among the various merging methods we use. This choice was informed by our analysis in Section 5.2. In Table 1, we have highlighted the gain/loss of performance for each benchmark due to AIM and we can see that in the vast majority of cases, AIM causes a significant performance boost, with an Average Change of 13% (ignoring the Inf value) and more than 40% HV Gain in 20% cases, further highlighted by the fact that the top performers on each benchmark, as well as the largest hypervolume gain, are all in models merged with AIM. We observe HumanEval (10 out of 20) and MBPP (17 out of 20) often see large boosts with AIM, especially when merging Instruction Tuned models with others. Some merges also reveal small drops in GSM8K or IFEval even when other benchmarks improve, reflecting the inherent trade-offs in merging specialized models. Overall, a clear majority **(80%) of merges exhibit improved HV Gain under AIM**, reinforcing that the method often enhances multi-task performance overall. We can further visualize this increase in hypervolume by looking at how AIM pushes the Pareto frontier. Figure 2 shows how applying AIM to existing merging methods extends the Pareto optimal frontier, which we also quantitatively measured using HV gain. These results showcase the

efficacy of the proposed method across a variety of merging methods and reinforce the hypothesis that the activation space encompasses useful insight for merging.

To validate AIM's generalizability beyond the Llama-2 family, we conducted a new experiment on a different architecture and modality: the Qwen 2.5-VL-7B vision-language model. We merged two distinct experts, Video-R1 model for video reasoning [11] and CAD-Coder model for image-to-code-based CAD geometry generation [9], with their instruct base model.

We evaluated this merge on 6 benchmarks: IFEval and MMLU for instruct base model capabilities, Video MMMU [18] and VSI-Bench [45] for video reasoning, and two CAD-Coder benchmarks for code generation from rendered and real images [9]. As shown in Table 2, AIM consistently improves the underlying merge methods across these diverse tasks and improves the Hypervolume Gain under all four merging methods. This demonstrates that AIM's benefits are not architecture-specific and generalize effectively to multimodal models.

For merging, therefore, we have two experts, one for CAD and one for video reasoning, with a base model that is instruction-tuned. Therefore, to assess performance, we perform the IF-Eval and MMLU benchmarks (the base model knowledge and instruction following) as well as expert benchmarks for video reasoning, Video MMMU [18] and VSI-Bench [45], and CAD generation benchmarks of CAD-Coder benchmark (on rendered CAD images) [9] and CAD-Coder Real benchmark (on real 3D printed images) [9]. As shown in Table 2 below, AIM consistently improves the performance of the underlying merging method across diverse tasks while also showing that the overall hypervolume gain is raised in all merging scenarios, demonstrating AIM's benefits are not limited to the Llama-2 architecture, and AIM does provide a path towards higher quality merging with little computational and data overhead.

Table 2: AIM generalizes across architectures and modalities. We evaluated merging two Qwen 2.5-VL-7B experts (Video-R1 and CADCoder, with an instruction-tuned base). AIM consistently improves performance across benchmarks, increasing the multi-task Hypervolume Gain in all cases.

Method	AIM	IFEval	MMLU	Video VSI	Video MMMU	CAD test100	CAD R400	HV Gain
Base Models								
Instruct	-	0.68	0.67	0.21	0.47	0.04	0.05	-
Video-R1	-	0.64	0.61	0.38	0.49	0.00	0.03	-
CADCoder	-	0.27	0.66	0.00	0.00	0.61	0.32	-
Merged Models								
TIES	No	0.57	0.68	0.23	0.41	0.61	0.32	0.44
	Yes	0.58 (+1.75%)	0.68	0.23	0.45 (+9.76%)	0.65 (+6.56%)	0.32	0.45 (+2.26%)
DARE	No	0.54	0.67	0.25	0.42	0.54	0.32	0.43
	Yes	0.56 (+3.70%)	0.67	0.26 (+4.00%)	0.45 (+7.14%)	0.55 (+1.85%)	0.32	0.44 (+3.47%)
DARE Task Arithmetic	No	0.25	0.64	0.25	0.43	0.54	0.33	0.38
	Yes	0.56 (+124.00%)	0.67 (+4.69%)	0.26 (+4.00%)	0.45 (+4.65%)	0.54	0.34 (+3.03%)	0.45 (+17.42%)
WIDEN	No	0.35	0.66	0.08	0.26	0.61	0.33	0.32
	Yes	0.35	0.66	0.14 (+75.00%)	0.32 (+23.08%)	0.64 (+4.92%)	0.33	0.36 (+13.24%)

5.2 Ablation study

To understand the effects of changing ω in AIM, we conduct an ablation study on the case of merging all three expert LLMs. For this study, we apply AIM with $\omega \in \{0.0, 0.2, 0.4, 0.6, 0.8\}$ and run the benchmarks on each merged model with each value of ω. For brevity, we do not report all benchmark results for each value here; instead, we track the hypervolume gain (The full set of results are presented in Appendix B). Specifically, to better visualize the effect of ω, we measure the relative change in HV Gain compared to no AIM (i.e., $\omega = 1.0$). We present these results in Figure 3. In most merging methods, we see that decreasing ω to even 0 benefits the model performance. However, in TIES merging particularly, we see that decreasing ω beyond 0.4 seems to degrade performance, and setting ω to the most extreme case of 0.0 does see some degradation in WIDEN as well. Given this, it seems that in these experiments, a value of 0.4 balances the performance gains in methods responding well to AIM and the potential degradation of methods that benefit less from AIM. However, given this observation that in some cases pushing ω to 0 still yields benefits, there may be some value in exploring non-linear scaling of activation magnitudes and non-linear relaxation schemes that could further boost performance in some cases.

To empirically validate the sensitivity to the calibration set, we conducted an ablation study on the calibration set size. We applied AIM to the DARE TIES merge of all three Llama-2 experts from Table 1, varying the number of calibration blocks from 1 to 256. As shown in Figure 4, the results demonstrate that AIM is highly robust. The Hypervolume Gain (HV Gain) increases substantially with just one block and stabilizes by 8 blocks (sourced from the Pile corpus Gao et al. [13]). This confirms that the data overhead for AIM is minimal and the method is highly practical.

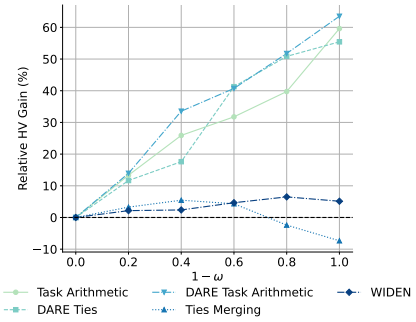

Figure 3: The Impact of the Relaxation Factor ω on Merged Model Performance. This figure plots the relative change in HV-Gain compared to scenarios without AIM. The x-axis represents $1 - \omega$, reflecting that decreasing ω results in more relaxation. The plot indicates that for some tasks, smaller values of ω continue to yield benefits. An ω of 0.4-0.6 appears to strike a balance.

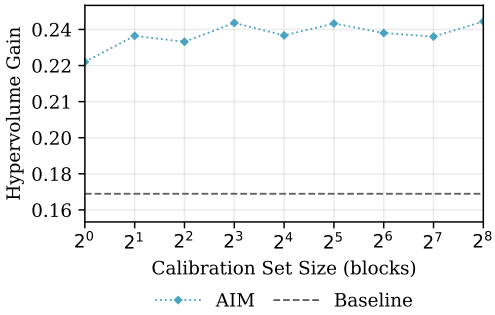

Figure 4: AIM's Robustness to Calibration Set Size. HV-Gain is plotted against the number of calibration blocks (log scale) for the DARE TIES merge. The dashed line is the baseline performance without AIM. Significant improvement is achieved with small data, and performance stabilizes at only 8 blocks.

6 Conclusion and Outlook

In this work, we introduced Activation-Informed Merging (AIM) as a complementary algorithm to existing model merging techniques for large language models (LLMs). We hypothesized that the activation space of LLMs harbors useful information that is often overlooked in model merging, as most existing methods operate purely on the weight space. To explore this potential information in the activation space, we viewed the problem from a continual learning perspective and proposed leveraging the activation space information from a task-agnostic calibration set. This approach selectively preserves critical weights from the pre-trained model, mitigating catastrophic forgetting while incorporating knowledge from fine-tuned models, yielding overall higher-performing models.

Through extensive empirical evaluations across multiple merging methods and benchmark tasks, and model architectures (including the Llama-2 and Qwen-VL families), we demonstrated that AIM consistently improves performance, often yielding superior results in comparison to the original merging methods it was applied to. These results empirically confirm our hypothesis on the importance of the activation space. Notably, AIM boosted merged model performance by up to **40%** in some cases, underscoring the crucial role and the potential of activation information in merging methods. Furthermore, our ablation study confirmed AIM's robustness, showing significant gains even with minimal calibration data (as few as 8 blocks), highlighting the method's practical applicability. Our findings strongly highlight the necessity and benefit of incorporating activation-informed strategies when merging multiple fine-tuned models.

Moving forward, our findings open up several promising directions for future research. First, our results indicate that even aggressively preserving salient weights of the pre-trained model is effective across many merging scenarios. This highlights the promise for more advanced activation-informed strategies and non-linear relaxation methods to potentially further enhance performance. Beyond the pre-trained activations explored in this work, there is room to improve existing merging methods by leveraging the broader activation space of the models being merged. So far, AIM has only considered the activations of the pre-trained model, while the activations of the expert LLMs remain unexplored. Future research should focus on developing methods that also encompass information from the expert model activations. Additionally, in future works, more theoretically grounded approaches for incorporating the activation space of LLMs in merging should be developed and tested. These integrations will hold great value in improving the quality and performance of merging methods in an increasingly competitive and ever more efficient landscape of LLMs, which could benefit from smaller and more efficient yet more powerful models.

Overall, AIM serves as a robust and adaptable augmentation to existing LLM merging techniques, offering a principled way to incorporate activation information for more effective model fusion. By prioritizing the activation-aware perspective, we take a step towards more stable, efficient, and generalizable merged models, demonstrated across different architectures and modalities, that better leverage the strengths of multiple fine-tuned experts.

Acknowledgments and Disclosure of Funding

This work was supported in part by the MIT-IBM Watson AI Lab.

References

[1] Rahaf Aljundi, Francesca Babiloni, Mohamed Elhoseiny, Marcus Rohrbach, and Tinne Tuytelaars. Memory aware synapses: Learning what (not) to forget. In *Proceedings of the European conference on computer vision (ECCV)*, pages 139–154, 2018.

[2] Cesar Almecija, Apoorva Sharma, and Navid Azizan. Uncertainty-aware meta-learning for multimodal task distributions. *arXiv preprint arXiv:2210.01881*, 2022.

[3] Jacob Austin, Augustus Odena, Maxwell Nye, Maarten Bosma, Henryk Michalewski, David Dohan, Ellen Jiang, Carrie Cai, Michael Terry, Quoc Le, et al. Program synthesis with large language models. *arXiv preprint arXiv:2108.07732*, 2021.

[4] Sahil Chaudhary. Code alpaca: An instruction-following llama model for code generation. https://github.com/sahil280114/codealpaca, 2023.

[5] Mark Chen, Jerry Tworek, Heewoo Jun, Qiming Yuan, Henrique Ponde De Oliveira Pinto, Jared Kaplan, Harri Edwards, Yuri Burda, Nicholas Joseph, Greg Brockman, et al. Evaluating large language models trained on code. *arXiv preprint arXiv:2107.03374*, 2021.

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

[7] MohammadReza Davari and Eugene Belilovsky. Model breadcrumbs: Scaling multi-task model merging with sparse masks, 2024.

[8] Jesse Dodge, Gabriel Ilharco, Roy Schwartz, Ali Farhadi, Hannaneh Hajishirzi, and Noah Smith. Fine-tuning pretrained language models: Weight initializations, data orders, and early stopping, 2020.

[9] Anna C Doris, Md Ferdous Alam, Amin Heyrani Nobari, and Faez Ahmed. Cad-coder: An open-source vision-language model for computer-aided design code generation. *arXiv preprint arXiv:2505.14646*, 2025.

[10] Mehrdad Farajtabar, Navid Azizan, Alex Mott, and Ang Li. Orthogonal gradient descent for continual learning. In *International Conference on Artificial Intelligence and Statistics*, pages 3762–3773. PMLR, 2020.

[11] Kaituo Feng, Kaixiong Gong, Bohao Li, Zonghao Guo, Yibing Wang, Tianshuo Peng, Junfei Wu, Xiaoying Zhang, Benyou Wang, and Xiangyu Yue. Video-r1: Reinforcing video reasoning in mllms. *arXiv preprint arXiv:2503.21776*, 2025.

[12] Elias Frantar and Dan Alistarh. Optimal brain compression: A framework for accurate post-training quantization and pruning. *Advances in Neural Information Processing Systems*, 35: 4475–4488, 2022.

[13] Leo Gao, Stella Biderman, Sid Black, Laurence Golding, Travis Hoppe, Charles Foster, Jason Phang, Horace He, Anish Thite, Noa Nabeshima, Shawn Presser, and Connor Leahy. The pile: An 800gb dataset of diverse text for language modeling, 2020.

[14] 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, Eric Tang, Anish Thite, Ben Wang, Kevin Wang, and Andy Zou. A framework for few-shot language model evaluation, 2024.

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

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

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

[18] Kairui Hu, Penghao Wu, Fanyi Pu, Wang Xiao, Yuanhan Zhang, Xiang Yue, Bo Li, and Ziwei Liu. Video-mmmu: Evaluating knowledge acquisition from multi-discipline professional videos. *arXiv preprint arXiv:2501.13826*, 2025.

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

[20] Dong-Hwan Jang, Sangdoo Yun, and Dongyoon Han. Model stock: All we need is just a few fine-tuned models. In *European Conference on Computer Vision*, pages 207–223. Springer, 2025.

[21] Xisen Jin, Xiang Ren, Daniel Preotiuc-Pietro, and Pengxiang Cheng. Dataless knowledge fusion by merging weights of language models. In *The Eleventh International Conference on Learning Representations*, 2023.

[22] James Kirkpatrick, Razvan Pascanu, Neil Rabinowitz, Joel Veness, Guillaume Desjardins, Andrei A Rusu, Kieran Milan, John Quan, Tiago Ramalho, Agnieszka Grabska-Barwinska, et al. Overcoming catastrophic forgetting in neural networks. *Proceedings of the national academy of sciences*, 114(13):3521–3526, 2017.

[23] Ji Lin, Jiaming Tang, Haotian Tang, Shang Yang, Wei-Ming Chen, Wei-Chen Wang, Guangxuan Xiao, Xingyu Dang, Chuang Gan, and Song Han. Awq: Activation-aware weight quantization for llm compression and acceleration, 2024.

[24] Haipeng Luo, Qingfeng Sun, Can Xu, Pu Zhao, Jianguang Lou, Chongyang Tao, Xiubo Geng, Qingwei Lin, Shifeng Chen, Yansong Tang, and Dongmei Zhang. Wizardmath: Empowering mathematical reasoning for large language models via reinforced evol-instruct, 2025.

[25] Michael S Matena and Colin A Raffel. Merging models with fisher-weighted averaging. *Advances in Neural Information Processing Systems*, 35:17703–17716, 2022.

[26] Youngjae Min, Kwangjun Ahn, and Navid Azizan. One-pass learning via bridging orthogonal gradient descent and recursive least-squares. In *2022 IEEE 61st Conference on Decision and Control (CDC)*, pages 4720–4725. IEEE, 2022.

[27] Aldo Pareja, Nikhil Shivakumar Nayak, Hao Wang, Krishnateja Killamsetty, Shivchander Sudalairaj, Wenlong Zhao, Seungwook Han, Abhishek Bhandwaldar, Guangxuan Xu, Kai Xu, Ligong Han, Luke Inglis, and Akash Srivastava. Unveiling the secret recipe: A guide for supervised fine-tuning small llms, 2024.

[28] Alexandre Ramé, Kartik Ahuja, Jianyu Zhang, Matthieu Cord, Léon Bottou, and David Lopez-Paz. Model ratatouille: Recycling diverse models for out-of-distribution generalization. In *International Conference on Machine Learning*, pages 28656–28679. PMLR, 2023.

[29] Alexandre Rame, Nino Vieillard, Leonard Hussenot, Robert Dadashi-Tazehozi, Geoffrey Cideron, Olivier Bachem, and Johan Ferret. WARM: On the benefits of weight averaged reward models. In *Proceedings of the 41st International Conference on Machine Learning*, pages 42048–42073. PMLR, 2024.

[30] Alexandre Ramé, Johan Ferret, Nino Vieillard, Robert Dadashi, Léonard Hussenot, Pierre-Louis Cedoz, Pier Giuseppe Sessa, Sertan Girgin, Arthur Douillard, and Olivier Bachem. Warp: On the benefits of weight averaged rewarded policies, 2024.

[31] Hippolyt Ritter, Aleksandar Botev, and David Barber. Online structured laplace approximations for overcoming catastrophic forgetting. *Advances in Neural Information Processing Systems*, 31, 2018.

[32] Apoorva Sharma, Navid Azizan, and Marco Pavone. Sketching curvature for efficient out-of-distribution detection for deep neural networks. In *Uncertainty in artificial intelligence*, pages 1958–1967. PMLR, 2021.

[33] Haizhou Shi, Zihao Xu, Hengyi Wang, Weiyi Qin, Wenyuan Wang, Yibin Wang, Zifeng Wang, Sayna Ebrahimi, and Hao Wang. Continual learning of large language models: A comprehensive survey. *arXiv preprint arXiv:2404.16789*, 2024.

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

[35] Shivchander Sudalairaj, Abhishek Bhandwaldar, Aldo Pareja, Kai Xu, David D. Cox, and Akash Srivastava. Lab: Large-scale alignment for chatbots, 2024.

[36] Mingjie Sun, Zhuang Liu, Anna Bair, and J. Zico Kolter. A simple and effective pruning approach for large language models, 2024.

[37] Yi-Lin Sung, Linjie Li, Kevin Lin, Zhe Gan, Mohit Bansal, and Lijuan Wang. An empirical study of multimodal model merging. In *The 2023 Conference on Empirical Methods in Natural Language Processing*, 2023.

[38] Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, Dan Bikel, Lukas Blecher, Cristian Canton Ferrer, Moya Chen, Guillem Cucurull, David Esiobu, Jude Fernandes, Jeremy Fu, Wenyin Fu, Brian Fuller, Cynthia Gao, Vedanuj Goswami, Naman Goyal, Anthony Hartshorn, Saghar Hosseini, Rui Hou, Hakan Inan, Marcin Kardas, Viktor Kerkez, Madian Khabsa, Isabel Kloumann, Artem Korenev, Punit Singh Koura, Marie-Anne Lachaux, Thibaut Lavril, Jenya Lee, Diana Liskovich, Yinghai Lu, Yuning Mao, Xavier Martinet, Todor Mihaylov, Pushkar Mishra, Igor Molybog, Yixin Nie, Andrew Poulton, Jeremy Reizenstein, Rashi Rungta, Kalyan Saladi, Alan Schelten, Ruan Silva, Eric Michael Smith, Ranjan Subramanian, Xiaoqing Ellen Tan, Binh Tang, Ross Taylor, Adina Williams, Jian Xiang Kuan, Puxin Xu, Zheng Yan, Iliyan Zarov, Yuchen Zhang, Angela Fan, Melanie Kambadur, Sharan Narang, Aurelien Rodriguez, Robert Stojnic, Sergey Edunov, and Thomas Scialom. Llama 2: Open foundation and fine-tuned chat models, 2023.

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

[40] Mitchell Wortsman, Gabriel Ilharco, Samir Ya Gadre, Rebecca Roelofs, Raphael Gontijo-Lopes, Ari S Morcos, Hongseok Namkoong, Ali Farhadi, Yair Carmon, Simon Kornblith, et al. 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, 2022.

[41] Can Xu, Qingfeng Sun, Kai Zheng, Xiubo Geng, Pu Zhao, Jiazhan Feng, Chongyang Tao, Qingwei Lin, and Daxin Jiang. WizardLM: Empowering large pre-trained language models to follow complex instructions. In *The Twelfth International Conference on Learning Representations*, 2024.

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

[43] Prateek Yadav, Tu Vu, Jonathan Lai, Alexandra Chronopoulou, Manaal Faruqui, Mohit Bansal, and Tsendsuren Munkhdalai. What matters for model merging at scale?, 2024.

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

[45] Jihan Yang, Shusheng Yang, Anjali W Gupta, Rilyn Han, Li Fei-Fei, and Saining Xie. Thinking in space: How multimodal large language models see, remember, and recall spaces. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pages 10632–10643, 2025.

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

[47] Le Yu, Bowen Yu, Haiyang Yu, Fei Huang, and Yongbin Li. Extend model merging from fine-tuned to pre-trained large language models via weight disentanglement. *arXiv preprint arXiv:2408.03092*, 2024.

[48] Jeffrey Zhou, Tianjian Lu, Swaroop Mishra, Siddhartha Brahma, Sujoy Basu, Yi Luan, Denny Zhou, and Le Hou. Instruction-following evaluation for large language models, 2023.

A Reproducibility Details

Here we provide the specific details for reproducing the results presented in the paper.

Checkpoints: Firstly, we specify the publicly available checkpoints we use in our experiments. Below is a list of checkpoints used and the links to the publicly available weights for these models:

- **Base Model:** https://huggingface.co/unsloth/llama-2-13b
- **Code Model:** https://huggingface.co/layoric/llama-2-13b-code-alpaca
- **Math Model:** https://huggingface.co/vanillaOVO/WizardMath-13B-V1.0
- **Instruction Tuned Model:** https://huggingface.co/WizardLMTeam/WizardLM-13B-V1.2

A Note On Weights: The weights we use in our experiments may not be exactly identical to the weights used in the experiments by Yu et al. [46] and Yu et al. [47], since the referenced weights for WizardMath-13B are no longer available publicly, instead we use a publicly available copy of the model.

Code and Data: Aside from the checkpoint, we provide our code and the link to the publicly available calibration data we use in our work. Our code is publicly available at https://github.com/ahnobari/ActivationInformedMerging and the calibration data can be found at https://huggingface.co/datasets/mit-han-lab/pile-val-backup.

B Ablation Detailed Results

Here we present the full results of the ablation study we conducted. Table 3 includes the granular values for all benchmarks we ran for different values of ω.

Table 3: Performance metrics for different methods with varying ω values.

Method	ω	HumanEval	MBPP	MMLU	MATH	GSM8K	IFEval	HV Gain
Ties Merging	0.0	19.51	29.0	54.32	4.6	17.6	26.24	0.1015
	0.2	19.51	28.6	54.35	5.3	21.2	25.37	0.1069
	0.4	20.73	29.2	54.46	5.7	23.7	25.98	0.1143
	0.6	20.12	27.8	54.25	6.6	32.1	23.9	0.1155
	0.8	20.73	27.8	54.09	6.6	33.7	24.07	0.1131
	1.0	21.34	29.2	53.97	6.3	29.2	26.95	0.1096
DARE Task Arithmetic	0.0	18.90	29.4	53.42	13.8	60.5	35.49	0.2623
	0.2	16.46	29.0	52.98	12.9	61.5	34.97	0.2434
	0.4	15.85	27.0	52.59	12.2	60.7	33.67	0.2257
	0.6	15.24	27.0	52.18	11.8	58.1	33.95	0.2142
	0.8	14.02	22.2	51.53	9.9	54.0	32.69	0.1828
	1.0	11.59	19.6	50.89	9.1	49.7	33.2	0.1604
DARE Ties	0.0	25.61	29.6	53.31	12.0	59.4	34.64	0.2669
	0.2	21.34	29.4	53.11	12.1	58.7	36.76	0.2590
	0.4	19.51	28.6	52.63	11.6	57.0	36.2	0.2426
	0.6	15.85	26.0	52.22	10.1	54.2	34.82	0.2019
	0.8	14.02	24.0	51.6	10.0	53.1	35.51	0.1917
	1.0	13.41	21.2	51.15	8.7	51.5	35.75	0.1717
Task Arithmetic	0.0	18.90	29.4	53.42	13.8	60.5	35.49	0.2623
	0.2	15.24	27.6	52.97	13.0	59.8	35.27	0.2296
	0.4	15.24	27.4	52.63	12.0	58.1	33.88	0.2165
	0.6	15.24	25.4	52.13	11.4	57.4	33.29	0.2069
	0.8	13.41	21.8	51.61	10.2	56.2	32.74	0.1862
	1.0	11.59	19.6	51.20	9.0	52.7	32.95	0.1643
WIDEN	0.0	27.44	33.2	55.26	14.0	64.9	32.39	0.3027
	0.2	28.05	33.0	55.16	14.2	65.6	32.39	0.3066
	0.4	26.83	32.8	54.98	14.4	64.0	32.76	0.3013
	0.6	25.61	33.4	54.77	14.2	63.0	32.06	0.2947
	0.8	26.22	32.6	54.64	14.0	64.1	31.74	0.2941
	1.0	25.00	33.2	54.58	13.5	64.2	31.44	0.2879

C Sensitivity-Based Formulation Comparison

In this section, we present the results of running post-merging relaxation using the gradient-based sensitivity-based formulation discussed in the main body. We run the relaxation scheme based on gradients with a $\omega = 0.4$, which we use in AIM. Table 4 shows how performance changes across benchmarks with gradient-based relaxation. We observe that the resulting performance boost remains very close to activation-based relaxation, with HV gain largely unchanged, with both gradient-based and pure activation-based boost trading blows evenly (see Table 1). Noting this and given that activations alone do not come with significant memory and computational cost, activation-based relaxation, which does not require computing gradients, is a more memory-efficient and computationally inexpensive process.

Table 4: Model performance comparison across different benchmarks, after relaxation applied using the gradients of the models with respect to the calibration data.

Method	Model(s)	Relaxation	HumanEval	MBPP	MMLU	MATH	GSM8K	IFEval	HV Gain
Base Models									
-	Base	-	17.07	27.80	52.18	0.70	4.20	25.10	-
-	Code	-	17.07	31.60	52.91	6.00	24.10	26.25	-
-	Instruction Tuned	-	26.83	34.80	53.41	7.50	43.40	35.67	-
-	Math	-	15.24	27.60	51.89	13.10	59.10	21.58	-
Merged Models									
DARE Average	Code + Instruction Tuned	No	26.83	34.40	53.53	8.40	45.80	33.42	0.27
		Yes	29.27 (+9.09%)	36.00 (+4.65%)	54.18 (+1.21%)	8.30 (-1.19%)	46.20 (+0.87%)	32.00 (-4.25%)	0.28 (+2.49%)
	Code + Math	No	16.46	28.60	51.96	15.10	64.70	22.02	0.23
		Yes	15.85 (-3.71%)	29.60 (+3.50%)	52.50 (+1.04%)	14.80 (-1.99%)	64.10 (-0.93%)	21.91 (-0.50%)	0.23 (-1.65%)
	Instruction Tuned + Math	No	5.49	19.00	51.08	9.80	54.30	32.35	0.18
		Yes	12.20 (+122.22%)	28.20 (+48.42%)	52.72 (+3.21%)	12.90 (+31.63%)	62.20 (+14.55%)	31.96 (-1.21%)	0.26 (+40.71%)
	Code + Instruction Tuned + Math	No	11.59	19.60	50.89	9.10	49.70	33.20	0.16
		Yes	15.85 (+36.76%)	27.00 (+37.76%)	52.59 (+3.34%)	12.20 (+34.07%)	60.70 (+22.13%)	33.59 (+1.17%)	0.23 (+40.59%)
DARE Ties	Instruction Tuned + Math	No	8.54	23.80	51.39	9.20	54.10	33.89	0.20
		Yes	15.85 (+85.60%)	30.20 (+26.89%)	52.89 (+2.92%)	11.60 (+26.09%)	57.80 (+6.84%)	35.63 (+5.13%)	0.26 (+31.22%)
	Code + Instruction Tuned	No	30.49	35.20	53.40	8.60	46.20	33.28	0.28
		Yes	30.49	36.80 (+4.55%)	54.02 (+1.16%)	8.60	47.20 (+2.16%)	33.16 (-0.36%)	0.29 (+1.63%)
	Code + Math	No	17.07	27.40	51.92	14.90	63.60	22.53	0.23
		Yes	17.68 (+3.57%)	29.00 (+5.84%)	52.61 (+1.33%)	15.20 (+2.01%)	63.90 (+0.47%)	21.10 (-6.35%)	0.24 (+4.00%)
	Code + Instruction Tuned + Math	No	13.41	21.20	51.15	8.70	51.50	35.75	0.17
		Yes	19.51 (+45.49%)	28.60 (+34.91%)	52.63 (+2.89%)	11.60 (+33.33%)	57.00 (+10.68%)	36.20 (+1.26%)	0.24 (+41.28%)
Task Arithmetic	Code + Instruction Tuned	No	29.27	33.80	53.44	8.60	47.10	31.60	0.28
		Yes	29.88 (+2.08%)	35.80 (+5.92%)	54.12 (+1.27%)	7.80 (-9.30%)	46.60 (-1.06%)	32.01 (+1.30%)	0.28 (+0.61%)
	Instruction Tuned + Math	No	4.27	20.20	51.50	10.00	54.20	31.31	0.18
		Yes	8.54 (+100.00%)	26.40 (+30.69%)	52.83 (+2.58%)	12.80 (+28.00%)	61.30 (+13.10%)	32.62 (+4.18%)	0.24 (+34.52%)
	Code + Math	No	18.29	28.60	52.10	15.00	64.70	21.92	0.24
		Yes	17.68 (-3.34%)	29.20 (+2.10%)	52.52 (+0.81%)	14.60 (-2.67%)	64.50 (-0.31%)	21.54 (-1.73%)	0.24 (-2.65%)
	Code + Instruction Tuned + Math	No	11.59	19.60	51.20	9.00	52.70	32.87	0.16
		Yes	15.24 (+31.49%)	27.40 (+39.80%)	52.63 (+2.79%)	12.00 (+33.33%)	58.10 (+10.25%)	33.91 (+3.16%)	0.22 (+31.97%)
Ties Merging	Code + Math	No	15.85	26.80	51.86	14.30	62.60	21.63	0.20
		Yes	15.85	28.60 (+6.72%)	52.29 (+0.83%)	15.30 (+6.99%)	63.80 (+1.92%)	22.64 (+4.67%)	0.23 (+13.55%)
	Instruction Tuned + Math	No	28.05	34.60	54.45	8.70	44.70	34.04	0.23
		Yes	27.44 (-2.17%)	35.00 (+1.16%)	54.74 (+0.53%)	9.30 (+6.90%)	46.10 (+3.13%)	34.51 (+1.38%)	0.25 (+6.38%)
	Code + Instruction Tuned	No	16.46	23.60	52.70	2.70	5.40	24.48	0.00
		Yes	15.24 (-7.41%)	24.20 (+2.54%)	53.15 (+0.85%)	2.60 (-3.70%)	5.20 (-3.70%)	22.87 (-6.58%)	0.05 (+inf%)
	Code + Instruction Tuned + Math	No	21.34	29.20	53.97	6.30	29.20	26.95	0.11
		Yes	20.73 (-2.86%)	29.20	54.46 (+0.91%)	5.70 (-9.52%)	23.70 (-18.84%)	25.98 (-3.60%)	0.11 (+4.33%)
WIDEN	Instruction Tuned + Math	No	24.39	30.40	54.20	14.60	66.00	30.82	0.30
		Yes	23.78 (-2.50%)	32.00 (+5.26%)	54.69 (+0.90%)	15.10 (+3.42%)	68.20 (+3.33%)	31.23 (+1.33%)	0.31 (+2.54%)
	Code + Math	No	17.07	29.40	53.35	14.20	64.40	24.02	0.24
		Yes	17.07	29.60 (+0.68%)	53.36 (+0.02%)	14.30 (+0.70%)	62.20 (-3.42%)	23.95 (-0.29%)	0.24 (-1.22%)
	Code + Instruction Tuned	No	26.22	35.60	54.90	8.30	45.00	30.42	0.27
		Yes	25.61 (-2.33%)	34.60 (-2.81%)	54.97 (+0.13%)	8.20 (-1.20%)	44.10 (-2.00%)	31.60 (+3.88%)	0.26 (-0.93%)
	Code + Instruction Tuned + Math	No	25.00	33.20	54.58	13.50	64.20	31.44	0.29
		Yes	26.83 (+7.32%)	32.80 (-1.20%)	54.98 (+0.73%)	14.40 (+6.67%)	64.00 (-0.31%)	32.82 (+4.39%)	0.30 (+4.70%)

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: claims made by the paper are quantitative and justified by experiments.

   Guidelines:

   - The answer NA means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A No or NA answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: experiments clearly state outcomes and performance, and future work needed is discussed.

   Guidelines:

   - The answer NA means that the paper has no limitation while the answer No means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

   Answer: [NA]

   Justification: [NA]

Guidelines:

- The answer NA means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes]

   Justification: details on training procedures and experiments are made clear in the appendix and main body. Code is also provided for exact replication.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - If the paper includes experiments, a No answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
   - If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
   - Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
   - While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
     (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
     (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
     (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
     (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

   Answer: [Yes]

Justification: code is provided to replicate the results, and the public checkpoints used are clearly mentioned.

Guidelines:

- The answer NA means that paper does not include experiments requiring code.
- Please see the NeurIPS code and data submission guidelines (https://nips.cc/public/guides/CodeSubmissionPolicy) for more details.
- While we encourage the release of code and data, we understand that this might not be possible, so "No" is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
- The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (https://nips.cc/public/guides/CodeSubmissionPolicy) for more details.
- The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
- The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
- At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
- Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer, etc.) necessary to understand the results?

   Answer: [Yes]

   Justification: Experiments and what each benchmark measures and how it is measured are made clear.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [No]

   Justification: Experiments in these settings are not accompanied by error bars, since a single model is used to run benchmarks and not a stochastic process that would involve differing results across runs.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The authors should answer "Yes" if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
   - The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
   - The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
   - The assumptions made should be given (e.g., Normally distributed errors).

- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g. negative error rates).
- If error bars are reported in tables or plots, The authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: We indicate the type of hardware used in benchmark experiments. No training is needed so that is not mentioned.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics https://neurips.cc/public/EthicsGuidelines?

   Answer: [Yes]

   Justification: we have verified adherence to the code of conduct.

   Guidelines:

   - The answer NA means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer No, they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [NA]

    Justification: No relevant societal impacts unique to this paper (based on existing public LLMs).

    Guidelines:

    - The answer NA means that there is no societal impact of the work performed.
    - If the authors answer NA or No, they should explain why their work has no societal impact or why the paper does not address societal impact.
    - Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.

- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pretrained language models, image generators, or scraped datasets)?

    Answer: [NA]

    Justification: based on already public models so not applicable.

    Guidelines: This paper does not contain such risks.

    - The answer NA means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
    - Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
    - We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [Yes]

    Justification: All sources of external data and model are mentioned.

    Guidelines:

    - The answer NA means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.
    - The name of the license (e.g., CC-BY 4.0) should be included for each asset.
    - For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
    - If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
    - For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
    - If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [Yes]

    Justification: Link to anonymized code is made available.

    Guidelines:

    - The answer NA means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [NA]

    Justification: no crowd-sourcing is involved in this work.

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
    - According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [NA]

    Justification: no crowdsourcing or human subjects involved.

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
    - Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
    - We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
    - For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

    Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does not impact the core methodology, scientific rigorousness, or originality of the research, declaration is not required.

Answer: [NA]

Justification: [NA]

Guidelines:

- The answer NA means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
- Please refer to our LLM policy (https://neurips.cc/Conferences/2025/LLM) for what should or should not be described.