Task-Aware Model Merging via Fisher-Weighted Median

Anonymous ACL submission

Abstract

Fine-tuning large language models provides strong in-domain performance but limits generalization and requires storage of many specialized models. Retraining a unified multitask model is often infeasible due to data unavailability or high computational cost. The majority of model merging approaches rely on performing arithmetic operations directly on model parameters. Although research in model merging has expanded significantly in recent years, two distinct approaches have become dominant: 1) techniques that mitigate interference from redundant parameters and sign conflicts, and 2) techniques that account for the varying sensitivity of individual parameters. However, these two approaches operate independently without considering each other's strengths and remain disconnected from each other. In this work, we aim to unify these two well-established yet currently disconnected approaches by integrating insights from both the approaches. We propose `DRIFT-MEDIAN`, a Fisher-aware model merging method that assigns sensitivity-weighted task vectors through a closed-form Fisher-weighted median, ensuring that task-relevant parameters dominate the merged model. Comprehensive experiments on several LLMs and CLIP models demonstrate that `DRIFT-MEDIAN` outperforms existing model merging methods.

1 Introduction

Large Language Models (LLMs) (Radford et al., 2019; Grattafiori et al., 2024; Touvron et al., 2023) usually require fine-tuning on domain-specific datasets to achieve optimal performance in specialized tasks. Although this approach yields strong in-domain performance, it introduces significant practical challenges in terms of substantial storage, computational costs, and limited data availability or data privacy constraints. Model merging has emerged as a compelling solution to these challenges by combining parameters from independently fine-tuned models of identical architecture into a single unified model (Ilharco et al., 2022a; Hinton et al., 2006; Yadav et al., 2023; Yu et al., 2024; Yang et al., 2023), eliminating the need for costly retraining. Existing approaches operate either in the parameter space (PS), where merging directly manipulates model weights (Jin et al., 2023; Shoemake, 1985; Akiba et al., 2025; Yang et al., 2023), or in data-flow space (DFS), where individual model parameters remain intact while optimization focuses on inference pathways (Kim et al., 2024). Hybrid approaches such as Evolutionary Model Merging (Akiba et al., 2025) incorporate elements of both paradigms. Despite significant progress in this field, there is still considerable scope for improving the effectiveness of current parameter-space merging methods.

Challenges and Motivation: A comprehensive review of current parameter-space merging techniques highlights two distinct approaches. While each addresses complementary facets of the merging challenge, they fall short of exploiting the synergistic benefits that their integration could offer - (1) approaches that resolve parameter interference during merging but do not consider parameter sensitivity to task performance (Yadav et al., 2023; Yu et al., 2024), and (2) methods that account for individual parameter sensitivity but do not address parameter interference aspects during the merging process (Matena and Raffel, 2022).

An example of the first type is TIES merging (Yadav et al., 2023) that tackles two key sources of interference namely the sign disagreement, where task vectors exhibit opposing directional updates that cancel each other during averaging, and redundant parameters, where uninformative updates dilute the contributions of more significant changes. However, it does not consider individual parameter sensitivity to task performance, potentially allowing less critical parameters to overshadow more

influential ones during the merging process.

An alternative approach, represented by Fisher merging (Matena and Raffel, 2022), incorporates parameter sensitivity through Fisher information weighting. It tackles the problem as maximizing the joint likelihood of the models' posterior distribution over parameters, demonstrating that parameter averaging is equivalent to employing an isotropic Gaussian distribution as an approximation for the posterior in each model. The merged parameter is estimated via weighted averaging over parameter's Fisher information. However, it does not address parameter interference issues, resulting in redundant parameters contributing to task conflicts and computational overhead during inference.

This separation suggests a promising approach, as we believe effective model merging requires both: (a) accounting for parameter sensitivity across individual models to preserve critical task-specific knowledge, and (b) appropriately managing parameter interference to eliminate redundancy.

Overview and Contributions: Motivated by aforementioned observations, we propose DRIFT-MEDIAN, a parameter-space merging framework that unifies insights from both interference reduction and sensitivity-based merging techniques.

DRIFT-MEDIAN operates through a carefully chosen sequence of operations, as illustrated in Figure 1. We first compute task vectors (Ilharco et al., 2023), then perform sign resolution to eliminate directionally conflicting updates by establishing consensus directions at each coordinate. To quantify parameter importance, we compute empirical Fisher information matrices that capture the sensitivity of each parameter to task performance. Unlike prior methods that perform pruning within individual models, we introduce coordinate-wise Top-K selection that operates across models, retaining only the most informative task vectors at each parameter position to prevent both parameter crowding and scarcity issues.

The core idea lies in our Fisher-weighted median aggregation, which we formulate as an L_1-minimization problem with a closed-form solution based on the Fisher-weighted median (Gurwitz, 1990). This approach ensures that parameter contributions reflect both sensitivity and relevance while maintaining robustness to outliers which gives a critical advantage over traditional Fisher-weighted averaging approaches that can be dominated by extreme values. Through extensive ablation studies, we confirm the importance of each component in our framework, establishing DRIFT-MEDIAN as a robust and principled approach to parameter-space model merging. Key contributions of our proposed framework are as follows:

- **Fisher-weighted median aggregation:** We propose DRIFT-MEDIAN, a parameter- space merging method where we introduce a closed-form solution, based on the Fisher-weighted median, ensuring that parameter contributions reflect both sensitivity and relevance, leading to robust and balanced merging.
- **Coordinate-wise Top-K selection:** Unlike prior methods that prune updates within each model independently, we perform cross-model coordinate-wise filtering to retain only the most informative task vectors, reducing noise and ensuring balanced aggregation.
- **Comprehensive evaluation:** Experiments across mathematics, coding, multilingual reasoning, safety, vision tasks and instruction-following demonstrate that DRIFT-MEDIAN consistently achieves higher performance retain rate (PRR) compared to prior methods, while ablation studies confirm the importance of each design choice.

2 Related Work

Background: Given a base model with parameters $\theta^{(0)}$ and a collection of fine-tuned models $\{\theta^{(m)}\}_{m=1}^{M}$ specialized for tasks $\{t_1, t_2, \ldots, t_M\}$, the objective is to consolidate these specialized weights into a unified multitask model that maintains strong performance across all constituent domains. The central concept underlying parameter-space merging is the task vector, formally defined as $\tau^{(m)} = \theta^{(m)} - \theta^{(0)}$, where $\theta^{(0)}$ represents the base model parameters and $\theta^{(m)}$ denotes the parameters of a model fine-tuned for task m. Task Arithmetic (Ilharco et al., 2023) demonstrated that these task vectors effectively encode task-specific knowledge and that their addition to the base model successfully transfers the corresponding task capabilities. However, naive averaging of task vectors often leads to destructive interference with conflicting parameter directions, prompting the development of sophisticated merging techniques.

Parameter Interpolation and Weight Averaging Methods: Early approaches to model merging focused on linear interpolation techniques, leveraging the observation that despite the inherent

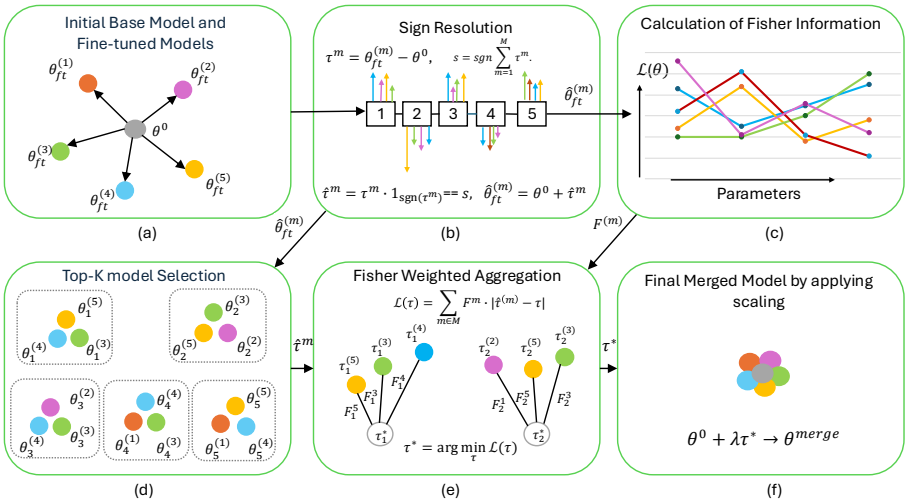


Figure 1: Overview of the steps involved in the proposed model merging approach. (a) θ^0 and $\theta_{ft}^{(m)}$ (with specific color) represent base model and different fine-tuned models respectively. (b) In the sign resolution step, task vectors that agree in sign are retained to compute the fine-tuned parameters $\hat{\theta}_{ft}^{(m)}$. 1, 2, 3, 4, 5 inside small square boxes represent parameter indices. (c) Diagonal Fisher matrix is estimated from the fine-tuned parameters. (d) Top-K models are selected at each coordinate based on distance from the base model, using a specified threshold value. Superscript and subscript of θ represent respective fine-tuned model and parameter indices. (e) Finally, Fisher-weighted aggregation yields τ^*, followed by scaling (f) to obtain the merged parameters θ^{merge}.

non-linearity of neural networks, linear combinations of their weights can preserve high accuracy when the constituent models share common optimization trajectories (Choshen et al., 2022; Ilharco et al., 2022b; Izmailov et al., 2019; Wang et al., 2024; Choi et al., 2024; Daheim et al., 2024). Choshen et al. (2022) proposed a straightforward weight averaging approach for fusing fine-tuned models, demonstrating superior performance compared to using pretrained models alone. Building on this foundation, Wortsman et al. (2022) introduced "model soups", where multiple models fine-tuned with different hyperparameters are combined through weight averaging rather than selecting the single best-performing model based on validation metrics. This approach consistently improves performance over individual model selection. Similar improvements through weight averaging have been reported by Ilharco et al. (2022b); Matena and Raffel (2022); Li et al. (2022). More sophisticated interpolation methods have emerged to address specific challenges in parameter combination. Jin et al. (2023) developed a technique for determining merged model parameters through closed-form solutions, formulating the problem as local linear regression for individual layers within the model. These methods call for advanced merging strategies but typically do not account for parameter importance or interference effects.

Task Vector Arithmetic and Interference Resolution: Ilharco et al. (2023) formalized the concept of task vectors and demonstrated their effectiveness in model editing through arithmetic operations. However, this approach revealed fundamental challenges when task vectors exhibit conflicting update directions, leading to the development of interference-aware methods. TIES merging (Yadav et al., 2023) addresses these interference issues through two key innovations. First, a trim step retains only the largest parameter deviations at each coordinate, suppressing redundant or weak updates that dilute informative changes. Second step ensures directional consistency by choosing the majority sign across models at each coordinate and zeroing out conflicting updates. While effective at reducing destructive interference, TIES merging does not incorporate parameter sensitivity considerations, potentially allowing less critical parameters to overshadow more influential ones during aggregation. DARE (Yu et al., 2024) employs a complementary strategy of randomly dropping delta parameters with probability p and rescaling the remaining parameters to maintain overall magnitude. This stochastic approach provides regularization benefits but lacks principled parameter importance weighting. Recent work such as SCE-merging

(Wan et al., 2025) uses variance and magnitude-based criteria together with sign-consistency rules to identify stable parameters across models, while PCB-merging (DU et al., 2024) focuses on balancing inter-model and intra-model competition among task vectors.

Sensitivity-Aware, Domain-Specific and Sparse Model Fusion Methods : Fisher merging (Matena and Raffel, 2022) formulates merging as maximizing the joint likelihood of models' posterior distributions over parameters, demonstrating that parameter averaging is equivalent to using isotropic Gaussian approximations for each model's posterior. This ensures that parameters with higher estimated importance exert stronger influence during merging. However, Fisher merging does not address parameter interference issues, allowing redundant parameters to contribute to task conflicts and increasing computational overhead as the number of models grows. Recent work has developed specialized merging techniques for specific application domains. Zhou et al. (2024) introduced model exclusive task arithmetic for billion-scale models, while Djuhera et al. (2025); Hammoud et al. (2024) focus on maintaining safety alignment during merging procedures. These domain-specific approaches highlight the importance of preserving critical model properties beyond task performance. LoRA merging methods (Shah et al., 2024; Shenaj et al., 2024; Stoica et al., 2025; Yin et al., 2025) are designed to handle the unique properties of low-rank parameter updates, which present different challenges compared to full parameter fine-tuning. Similarly, vision-specific merging techniques (Zhu et al., 2025) have been developed to address the particular characteristics of computer vision models.

3 Proposed Method

Suppose $\theta^{(0)} \in \mathbb{R}^N$, and $\{\theta^{(m)}\}_{m=1}^M$ denote the parameter vectors for the base model, and a collection of fine-tuned models derived from the same base model, respectively. We denote the corresponding task vectors as $\tau^{(m)} = \theta^{(m)} - \theta^{(0)}$, that capture the parameter displacements induced by fine-tuning. Our objective is to combine these task vectors into a single stable representation that preserves salient task information (Algorithm 1) while mitigating destructive interference. In our method, we accomplish this via the following steps - (i) performing *sign resolution* to eliminate directionally conflicting updates, (ii) computation of *Fisher information* to quantify the sensitivity of each parameter, (iii) applying *coordinate-wise Top-K filtering* to retain only the strongest displacements, (iv) *computing a merged task-vector* by applying Fisher-weighted coordinate-wise median across the filtered task vectors, and (v) *scaling* on the aggregated neurons to compensate for the lost neurons during merging.

3.1 Sign resolution

Consider a collection of task vectors $\{\tau^{(1)}, \ldots, \tau^{(M)}\}$, each defined over coordinates $i \in \{1, \ldots, d\}$. At a given coordinate i, the corresponding set of entries is denoted as $\{\tau_i^{(1)}, \ldots, \tau_i^{(M)}\}$. Since these entries may take both positive and negative values, averaging them directly can result in destructive elimination of directional consistency, thereby discarding potentially stable task-specific knowledge.

To ensure alignment, we inherit the concept of sign consensus from TIES merging (Yadav et al., 2023). This is determined by the sign of the aggregated update across tasks as $s_i = \text{sign}\left(\sum_{m=1}^M \tau_i^{(m)}\right)$. The consensus sign s_i specifies the dominant orientation of update at coordinate i. Contributions inconsistent with this orientation are suppressed via the pruning rule given as follows $\hat{\tau}_i^{(m)} = \begin{cases} \tau_i^{(m)}, & \text{if } \text{sign}(\tau_i^{(m)}) = s_i \\ 0, & \text{otherwise} \end{cases}$. The resulting collection $\{\hat{\tau}^{(1)}, \ldots, \hat{\tau}^{(M)}\}$ is therefore sign-consistent by construction. At each coordinate i, only those updates that are aligned with the consensus direction are preserved, while conflicting contributions are eliminated. This procedure removes destructive interference and results in a representation where all preserved task information is aligned.

3.2 Sensitivity Analysis via Diagonal Fisher Information Matrix

While directional alignment ensures that task updates no longer interfere destructively, it does not incorporate the relative importance of different parameters. Not all coordinates contribute equally to model behavior in that some parameters are highly sensitive and strongly influence the predictive distribution, whereas others are less critical. To account for this, we introduce an importance weighting scheme based on empirical Fisher information (Matena and Raffel, 2022). Formally, consider coordinate i in a model m. We define its empirical

Fisher information as

$$F_i^{(m)} = \mathbb{E}_{(x,y)\sim D_m}\left[\left(\frac{\partial}{\partial\hat{\theta}_i}\log p\Big(y \mid x;\, \hat{\theta}_i^{(m)}\Big)\right)^2\right], \tag{1}$$

where D_m denotes the data distribution associated with task m, $\theta^{(0)}$ is the reference initialization point (e.g., the pre-trained parameters), and $\hat{\theta}_i^{(m)}$ is the sign-aligned model weight and $\log p\Big(y \mid x;\, \hat{\theta}_i^{(m)}\Big)$ represent corresponding model posterior. Intuitively, $F_i^{(m)}$ quantifies the sensitivity of the model's predictive likelihood with respect to perturbations in parameter θ_i. A large $F_i^{(m)}$ value means that even small changes in θ_i significantly impact the likelihood, indicating this parameter is crucial for task m.

Conversely, a small Fisher value suggests that θ_i is relatively insensitive and thus less critical. Accordingly, when merging task vectors, the update contribution $\hat{\tau}_i^{(m)}$ should be weighted proportionally to its Fisher information. This ensures that parameters with high task-specific sensitivity exert stronger influence on the merged representation, while less important directions are naturally down-weighted. In combination with directional alignment, Fisher weighting therefore produces a merged update that is both sign-consistent and importance-aware, preserving critical task knowledge while suppressing noise from less informative coordinates. Following common practice, we use the diagonal of the Fisher matrix, which requires $\mathcal{O}|\theta|$ memory for storage. In contrast, storing the full Fisher matrix would require $\mathcal{O}|\theta|^2$ memory, making it impractical for large models.

3.3 Coordinate-wise Top-K Selection

After applying directional alignment and importance weighting, many parameters still exhibit small residual updates. These weak displacements are typically uninformative and can introduce noise into the merged representation. To mitigate this issue, we retain only the strongest task contributions on a per-parameter basis using an *inter-model* Top-K selection strategy. Our motivation for this coordinate-wise approach follows the perspective of (Qu and Horváth, 2025), which argues that model merging fundamentally decomposes into a set of independent one-dimensional estimation problems, one for each parameter. Consequently, interference, variance, and estimator instability arise *per coordinate*. In contrast, *intra-model* (model-wise) Top-K performs sparsification independently within each model, ignoring how other models distribute their update mass. This often forces multiple task vectors to concentrate disproportionately large updates on a small subset of parameters, thereby increasing cross-task interference. Our inter-model Top-K procedure solves this by limiting how many models may influence any given coordinate, ensuring balanced competition across tasks where the merged model must ultimately produce a single value.

For task m at coordinate i, define the update magnitude $d_i^{(m)} = \big|\hat{\tau}_i^{(m)}\big|$. Among the set $\{d_i^{(1)}, \ldots, d_i^{(M)}\}$, we select the K largest values, with $K = \lfloor \kappa \cdot M \rfloor, \quad \kappa \in (0, 1]$, where κ denotes the 'keep-ratio'. Suppose $\delta_i = \min\Big(\text{Top-}K\{d_i^{(1)}, \ldots, d_i^{(M)}\}\Big)$, represent the cut-off magnitude for retention for coordinate i. Then the set of retained task indices is given by, $M_i = \{m \in \{1, \ldots, M\} : d_i^{(m)} \geq \delta_i\}$. Thus, at each coordinate, only those task updates with sufficiently large magnitude - specifically, the top fraction κ - are preserved. This ensures that the merged representation emphasizes the most informative displacements while discarding weak or noisy contributions. Consequently, our approach leads to a more consistent and conflict-free parameter aggregation. We illustrate this phenomenon in Appendix E, where TIES selects the Top-K in an intra-model manner, whereas our method selects the Top-K in an inter-model manner.

3.4 Fisher-weighted Aggregation

After directional alignment, importance weighting, and Top-K filtering, we obtain a refined set of task-specific displacements at each coordinate. The final step is to merge these retained updates into a single consensus displacement. This can be formulated directly in parameter space as the solution of a *Fisher-weighted absolute-deviation minimization* problem. The objective can be expressed as:

$$L_i(\theta_i) = \sum_{m\in M_i} F_i^{(m)} \cdot \big|\hat{\theta}_i^{(m)} - \theta_i\big| \tag{2}$$

where M_i is the set of retained task indices from the Top-K filtering, and $F_i^{(m)}$ denotes the empirical Fisher information of parameter i for task m. For coordinate i, let τ_i denote the candidate merged displacement. Since each task-specific parameter vector is expressed as $\hat{\theta}^{(m)} = \theta^{(0)} + \hat{\tau}^{(m)}$, the aggregation objective Equation 2 can be expressed

Table 1: Results on GPT-2. The reported values correspond to absolute scores obtained on the validation set, since the test set is not publicly accessible.

Method	COLA	MNLI	MRPC	QNLI	QQP	RTE	SST2	Mean	$\overline{\text{PRR}}$
Fine-tuned Models	76.8	82.1	80.4	88.3	89.6	65.3	91.2	82.0	-
Model Averaging	55.0	55.1	51.0	57.6	76.7	44.8	52.5	56.1	68.5
Task Arithmetic	68.7	68.6	69.6	70.5	81.8	47.3	**83.6**	70.0	85.0
TIES	68.4	71.4	68.4	69.6	82.4	47.7	81.8	70.0	84.9
Fisher Merging	54.8	58.0	39.5	63.3	81.5	49.1	64.7	58.7	71.4
Localize & Stitch	64.1	**76.1**	48.0	65.5	**83.1**	**53.1**	55.7	63.7	77.9
Ours	**69.1**	71.3	**70.1**	**83.0**	79.3	50.2	77.3	**71.5**	**86.9**

in task-vector space as:

$$L_i(\tau_i) = \sum_{m \in M_i} F_i^{(m)} \cdot \left|\hat{\tau}_i^{(m)} - \tau_i\right| \quad (3)$$

This formulation enforces proximity between the merged displacement τ_i and the filtered task-specific values $\hat{\tau}_i^{(m)}$, while weighting each contribution according to its sensitivity. Parameters with larger Fisher values exert stronger influence, reflecting their higher functional importance. We adopt an L_1 (absolute-deviation) objective instead of the standard L_2 loss because the L_1 metric is more robust to outliers.

Closed-form Fisher-weighted Median The minimizer of the Fisher-weighted absolute-deviation loss admits a closed-form characterization in terms of a weighted median. Specifically, the optimal merged displacement τ_i^* at coordinate i is given by the Fisher-weighted median of the retained updates $\{\hat{\tau}_i^{(m)} : m \in M_i\}$. Formally, τ_i^* is defined as the value satisfying

$$\sum_{\hat{\tau}_i^{(m)} < \tau_i^*} F_i^{(m)} \leq \tfrac{1}{2} \sum_{m \in M_i} F_i^{(m)} \quad \text{and} \quad (4)$$

$$\sum_{\hat{\tau}_i^{(m)} > \tau_i^*} F_i^{(m)} \leq \tfrac{1}{2} \sum_{m \in M_i} F_i^{(m)}. \quad (5)$$

In other words, the Fisher weights of task-specific updates lying to the left and to the right of the solution τ_i^* each account for at most half of the total weight. This solution possesses a crucial robustness property: unlike Fisher-weighted means, which are highly sensitive to outliers due to squaring, the Fisher-weighted median ensures that extreme values cannot dominate the aggregate unless they are supported by sufficiently large Fisher weight. As a result, the merged displacement is both importance-aware and resistant to spurious task updates. We obtain the closed-form expression for the Fisher-weighted median following (Aho and Hopcroft, 1974; Blum et al., 1973; Gurwitz, 1990), and present the derivation in Appendix A.

Scaling Since sign pruning and Top-K filtering reduce the effective magnitudes of task displacements, a rescaling step is applied to restore the overall adaptation strength (Ilharco et al., 2023; Yadav et al., 2023; Yu et al., 2024). For each coordinate, the merged displacement is given by $\tau_i^{\text{merge}} = \lambda \cdot \tau_i^*$, where $\lambda > 0$ is a global scaling factor. After all these steps, the final merged model is then constructed as follows: $\theta^{\text{merge}} = \theta^{(0)} + \tau^{\text{merge}}$. The scaling factor λ serves as a tunable control that balances the contributions of the pre-trained model $\theta^{(0)}$ and the aggregated task updates. A larger value of λ increases the influence of task-specific displacements, causing the merged model to drift further from the base model, while smaller values preserve closer adherence to the pre-trained initialization. Appendix D covers implementation details and design choices.

4 Experimental Results

4.1 Experimental Setup

Baseline Methods We compare `DRIFT-MEDIAN` with seven different base-line methods, namely: Simple Averaging or Model Averaging (Wortsman et al., 2022; Choshen et al., 2022), Task Arithmetic (Ilharco et al., 2023), TIES (Yadav et al., 2023), DARE (Yu et al., 2024), Localize-and-Stitch (He et al., 2025a), Fisher merging (Matena and Raffel, 2022), and PCB-merging (DU et al., 2024). Hyper-parameters detail are given in Appendix I.

Models and Datasets We conduct experiments on three different types of model architectures: 1) Llama family of models with various number of model parameters (Llama-3.1-8B, Llama-3.2-3B, Llama-2-7b), 2) GPT-2, and 3) CLIP-ViT Model. We evaluate `DRIFT-MEDIAN` on various di-

Table 2: Results on Llama-3.1-8B models; The results are reported in relative percentage w.r.t. the fine-tuned models.

Method	Maths	Multilingual	Instruction	Coding	Safety	$\overline{\text{PRR}}$
Model Averaging (excl. embed)	92.70	96.58	44.13	89.57	74.99	78.84
Model Averaging (incl. embed)	93.18	96.83	42.16	89.56	76.13	78.84
Task Arithmetic	93.85	91.80	56.02	90.92	79.63	82.85
TIES	96.44	95.95	51.53	90.40	83.91	83.75
DARE	91.90	89.84	54.31	87.35	77.55	80.60
Fisher Merging	89.17	96.53	61.19	87.37	88.46	83.07
Localize & Stitch	**97.04**	**97.00**	45.26	85.98	61.28	77.32
Ours	85.19	89.44	**71.24**	**94.45**	**101.47**	**87.51**

Table 3: Results on Llama-3.2-3B models; The results are reported in relative percentage w.r.t. the fine-tuned models.

Method	Maths	Multilingual	Instruction	Coding	Safety	$\overline{\text{PRR}}$
Model Averaging (excl. embed)	63.08	101.69	56.81	88.11	57.81	73.50
Model Averaging (incl. embed)	64.83	101.79	51.74	86.36	56.46	72.24
Task Arithmetic	**72.40**	100.78	83.20	94.98	68.19	83.91
TIES	48.44	**102.04**	71.19	88.24	64.07	74.80
DARE	70.34	100.49	84.20	**96.31**	69.11	84.09
Fisher Merging	61.86	101.43	65.56	87.00	62.82	75.73
Localize & Stitch	68.59	101.63	56.81	86.36	49.00	72.48
Ours	65.77	99.96	**100.49**	93.80	**72.39**	**86.48**

verse datasets. Additional details are provided in Appendix H.

Evaluation Metric When each task or domain includes multiple evaluation benchmarks of varying difficulty levels, a direct comparison of raw scores across tasks can be misleading. To obtain a fair comparison, we consider **Performance Retain Rate (PRR)** (He et al., 2025b) as evaluation metric. PRR measures how much of the original performance of the task-specific fine-tuned model is retained by the merged model. Formally, for each task t, the PRR is defined as

$$\text{PRR}(t) = \frac{1}{N_t} \sum_{i=1}^{N_t} \frac{\text{Perf}(\theta^{\text{merge}}, D_{t,i})}{\text{Perf}\big(\theta^{(t)}, D_{t,i}\big)} \times 100,$$

where N_t is the number of evaluation benchmarks datasets for task t, $D_{t,i}$ denotes the i-th benchmark dataset for task t, $\text{Perf}(\theta, D)$ is the performance of model θ on dataset D, θ^{merge} denotes the merged model, and $\theta^{(t)}$ denotes the fine-tuned model on task t. This formulation normalizes the performance of the merged model against the best achievable performance for each benchmark (i.e., the fine-tuned baseline), and then averages across benchmarks within the task. By doing so, it avoids bias introduced by benchmarks of varying difficulty or scale, which would otherwise distort the results if raw scores were averaged directly. Thus, $\text{PRR}(t)$ provides a task-level measure of the degree to which the merged model retains the capabilities of the specialized fine-tuned models. Finally, we compute the mean PRR as $\overline{\text{PRR}} = 1/T \sum_{t=1}^{T} \text{PRR}(t)$ where T is the total number of tasks. This overall score reflects the average retention of task-specific performance by the merged model, providing a single metric for multi-task evaluation.

4.2 Results and Analysis

Merging Fully fine-tuned GPT-2 Based Models: For text classification, we adhere to the experimental setup of (Tang et al., 2024) for data and models. The setting considers a variety of text-classification tasks. We specifically consider 7 text-classification task (CoLA, MNLI, MRPC, QNLI, QQP, RTE, SST2) and report the experiments result in Table 1. Individual cell except the last two columns of this table represents absolute accuracy. Last column represents the evaluation metric mean Performance Retain Rate. We also report mean accuracy of respective merging method to make a fair comparison with (Tang et al., 2024). Notably, `DRIFT-MEDIAN` outperforms all the baseline method and exceeds

Table 4: Ablation study on Llama-3.1-8B models; The results are in relative percentage w.r.t. the fine-tuned models.

Method	Maths	Multilingual	Instruction	Coding	Safety	$\overline{\text{PRR}}$
- With Top-K, Sign Resolution and Median	85.19	89.44	71.24	94.45	101.47	87.51
- w/o Sign Resolution	86.66	91.40	63.09	93.22	101.18	86.16
- Mean instead of Median	88.82	91.88	57.68	93.61	97.19	85.22
- w/o Top-K	88.25	96.10	68.54	88.08	92.40	85.10
- w/o Scaling	87.73	94.01	59.31	93.40	100.67	85.77

the best baseline method by 1.9 margin in mean PRR.

Merging Fully fine-tuned Llama Based Models: For generation tasks we consider Llama-3.1-8B & Llama-3.2-3B, we report experimental results in Table 2 & Table 3. For Llama-3.1-8B and Llama-3.2-3B, we replicate the experiments of (He et al., 2025b) for respective data and models. We consider 5 different task domains (Mathematics, Multilingual, Instruction, Coding, and Safety) with varying level of complexity. Each domain has multiple benchmarks. We first evaluate the fine-tuned model on the test set of corresponding benchmark. Detailed individual benchmark accuracies can be found out in Appendix K. Since each task includes multiple evaluation benchmarks of varying difficulty levels, so we represent the score in corresponding cell by its PRR. Finally, we compare the merging methods by mean PRR. Our model improves the baseline method by 3.76% and 2.39% for Llama-3.1-8B and Llama-3.2-3B, respectively. Note, in some cases multitasking models (here merged model) can exhibit slightly better performance than the individual fine-tuned model. Therefore, this may result in a PRR exceeding 100 in certain cases. We refer the readers to Appendix F for results on LLaMA2-7B models and Appendix C for results on CLIP based models on vision tasks.

Ablation of DRIFT-MEDIAN Components: We systematically conduct ablation studies on each component of `DRIFT-MEDIAN` to evaluate their individual contributions. Beginning with the complete `DRIFT-MEDIAN` approach, we systematically eliminate or replace individual components and measure test set performance for the full model merging process. When we remove sign resolution step, we can clearly see the performance dip from 87.51% to 86.16%. In the Fisher weighted aggregation step of our method, if we consider mean instead of median we lose 2.29% of performance gain. If we eliminate the Top-K selection component, performance drops drastically (2.41%). Further, we conducted an analysis on the first five GLUE tasks using GPT-2 (CoLA, MNLI, MRPC, QNLI, QQP). With a keep ratio of 60%, intra-model Top-K + sign election (as in TIES) leaves 5.89% of parameters with no surviving task update- i.e., no task contributes at those coordinates, forcing a fallback to the base model (parameter scarcity). In contrast, sign election + inter-model Top-K reduces this to 2.06%, meaning far fewer coordinates are left unused. This demonstrates that inter-model Top-K more closely matches the per-coordinate merging objective, reduces update scarcity, and more effectively utilizes the available task information. Finally, if we take out scaling step, we lose 1.74% performance. Table 4 demonstrates that each component of the method is essential for achieving optimal performance. We provide a sensitivity study on λ and top-K on Figure J and observe good results in nearby zones of the optimal hyperparamters. Further, we provide an in-domain vs out-domain performance tradeoff by varying scaling factor λ in Appendix G and observed that in-domain performance increases with increase in scaling while it is the opposite for out-domain tasks.

5 Conclusions

We propose `DRIFT-MEDIAN`, a task-aware model merging framework that combines task-vector sign resolution, coordinate-wise Top-K selection, and Fisher-weighted median aggregation. By explicitly addressing sign disagreements and redundant-parameter interference, and incorporating parameter-sensitivity considerations, our method enables conflict-free parameter fusion while retaining task-specific knowledge. Experiments across mathematics, multilingual reasoning, coding, instruction following, and safety tasks demonstrate that `DRIFT-MEDIAN` consistently outperforms prior parameter-space merging methods such as TIES and Fisher merging. Potential directions for future work include the use of *dynamic* hyperparameters, where λ and κ adapt across models or even layers.

Limitations

DRIFT-MEDIAN relies on empirical Fisher information to estimate parameter importance, which introduces additional computational overhead compared to purely arithmetic merging methods. Although we use only the diagonal Fisher and compute it once per task, this step still requires multiple forward–backward passes over validation data and can be expensive for large models or many tasks. The method also assumes access to reasonably representative validation data for Fisher estimation. While our experiments show robustness to moderate domain mismatch, performance can degrade when the Fisher estimation data is highly dissimilar from the target task distribution. This limits applicability in settings where suitable proxy data is unavailable.

In addition, DRIFT-MEDIAN is restricted to parameter-space merging of models with identical architectures and shared initialization. It does not directly support merging heterogeneous architectures or structurally different parameterizations such as incompatible adapters.

Finally, the method introduces hyperparameters for sparsity and scaling. Although performance is relatively stable near optimal values, improper tuning can affect generalization, and automated selection of these hyperparameters remains an open problem.

Reproducibility Statement

To ensure reproducibility, we provide a comprehensive description of our methodology together with derivation and a clear presentation of the proposed algorithm. Our work relies exclusively on publicly available resources: all datasets, evaluation benchmarks, and pretrained model checkpoints used in this study are openly accessible to the research community. Upon acceptance, we will release the full source code, including training and evaluation scripts as well as detailed documentation, to facilitate independent verification and extension of our results. Furthermore, we specify all experimental details including hyper-parameters in Appendix H and Appendix I, ensuring that every component of our pipeline can be faithfully reproduced.

References

Alfred V Aho and John E Hopcroft. 1974. *The design and analysis of computer algorithms*. Pearson Education India.

Takuya Akiba, Makoto Shing, Yujin Tang, Qi Sun, and David Ha. 2025. Evolutionary optimization of model merging recipes. *Nature Machine Intelligence*, 7(2):195–204.

Jacob Austin, Augustus Odena, Maxwell Nye, Maarten Bosma, Henryk Michalewski, David Dohan, Ellen Jiang, Carrie Cai, Michael Terry, Quoc Le, and Charles Sutton. 2021. Program synthesis with large language models. *Preprint*, arXiv:2108.07732.

Manuel Blum, Robert W. Floyd, Vaughan R. Pratt, Ronald L. Rivest, Robert Endre Tarjan, and 1 others. 1973. Time bounds for selection. *J. Comput. Syst. Sci.*, 7(4):448–461.

Mark Chen, Jerry Tworek, Heewoo Jun, Qiming Yuan, Henrique Ponde de Oliveira Pinto, Jared Kaplan, Harri Edwards, Yuri Burda, Nicholas Joseph, Greg Brockman, Alex Ray, Raul Puri, Gretchen Krueger, Michael Petrov, Heidy Khlaaf, Girish Sastry, Pamela Mishkin, Brooke Chan, Scott Gray, and 39 others. 2021. Evaluating large language models trained on code. *Preprint*, arXiv:2107.03374.

Gong Cheng, Junwei Han, and Xiaoqiang Lu. 2017. Remote sensing image scene classification: Benchmark and state of the art. *Proceedings of the IEEE*, 105(10):1865–1883.

Jiho Choi, Donggyun Kim, Chanhyuk Lee, and Seunghoon Hong. 2024. Revisiting weight averaging for model merging. *CoRR*, abs/2412.12153.

Leshem Choshen, Elad Venezian, Noam Slonim, and Yoav Katz. 2022. Fusing finetuned models for better pretraining. *CoRR*, abs/2204.03044.

Mircea Cimpoi, Subhransu Maji, Iasonas Kokkinos, Sammy Mohamed, and Andrea Vedaldi. 2014. Describing textures in the wild. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3606–3613.

Tarin Clanuwat, Mikel Bober-Irizar, Asanobu Kitamoto, Alex Lamb, Kazuaki Yamamoto, and David Ha. 2018. Deep learning for classical japanese literature. *Preprint*, arXiv:cs.CV/1812.01718.

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. *Preprint*, arXiv:2110.14168.

Nico Daheim, Thomas Möllenhoff, Edoardo Ponti, Iryna Gurevych, and Mohammad Emtiyaz Khan. 2024. Model merging by uncertainty-based gradient matching. In *The Twelfth International Conference on Learning Representations*.

Aladin Djuhera, Swanand Kadhe, Farhan Ahmed, Syed Zawad, and Holger Boche. 2025. SafeMERGE: Preserving safety alignment in fine-tuned large language models via selective layer-wise model merging. In

ICLR 2025 Workshop on Foundation Models in the Wild.

Guodong DU, Junlin Lee, Jing Li, Runhua Jiang, Yifei Guo, Shuyang Yu, Hanting Liu, Sim Kuan Goh, Ho-Kin Tang, Daojing He, and Min Zhang. 2024. Parameter competition balancing for model merging. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*.

Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Alex Vaughan, Amy Yang, Angela Fan, Anirudh Goyal, Anthony Hartshorn, Aobo Yang, Archi Mitra, Archie Sravankumar, Artem Korenev, , and 6 others. 2024. The llama 3 herd of models. *Preprint*, arXiv:2407.21783.

Chaya Gurwitz. 1990. Weighted median algorithms for l1 approximation. *BIT*, 30(2):301–310.

Hasan Abed Al Kader Hammoud, Umberto Michieli, Fabio Pizzati, Philip Torr, Adel Bibi, Bernard Ghanem, and Mete Ozay. 2024. Model merging and safety alignment: One bad model spoils the bunch. In *Findings of the Association for Computational Linguistics: EMNLP 2024*, pages 13033–13046, Miami, Florida, USA. Association for Computational Linguistics.

Seungju Han, Kavel Rao, Allyson Ettinger, Liwei Jiang, Bill Yuchen Lin, Nathan Lambert, Yejin Choi, and Nouha Dziri. 2024. Wildguard: Open one-stop moderation tools for safety risks, jailbreaks, and refusals of llms. In *Advances in Neural Information Processing Systems*, volume 37, pages 8093–8131. Curran Associates, Inc.

Yifei He, Yuzheng Hu, Yong Lin, Tong Zhang, and Han Zhao. 2025a. Localize-and-stitch: Efficient model merging via sparse task arithmetic. *Transactions on Machine Learning Research*.

Yifei He, Siqi Zeng, Yuzheng Hu, Rui Yang, Tong Zhang, and Han Zhao. 2025b. Mergebench: A benchmark for merging domain-specialized llms. *Preprint*, arXiv:2505.10833.

Patrick Helber, Benjamin Bischke, Andreas Dengel, and Damian Borth. 2019. Eurosat: A novel dataset and deep learning benchmark for land use and land cover classification. *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, 12(7):2217–2226.

Dan Hendrycks, Collin Burns, Saurav Kadavath, Akul Arora, Steven Basart, Eric Tang, Dawn Song, and Jacob Steinhardt. 2021. Measuring mathematical problem solving with the MATH dataset. In *Thirty-fifth Conference on Neural Information Processing Systems Datasets and Benchmarks Track (Round 2)*.

Geoffrey E. Hinton, Simon Osindero, and Yee Whye Teh. 2006. A fast learning algorithm for deep belief nets. *Neural Computation*, 18:1527–1554.

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

Gabriel Ilharco, Marco Túlio Ribeiro, Mitchell Wortsman, Suchin Gururangan, Ludwig Schmidt, Hannaneh Hajishirzi, and Ali Farhadi. 2022a. Editing models with task arithmetic. *CoRR*, abs/2212.04089.

Gabriel Ilharco, Mitchell Wortsman, Samir Yitzhak Gadre, Shuran Song, Hannaneh Hajishirzi, Simon Kornblith, Ali Farhadi, and Ludwig Schmidt. 2022b. Patching open-vocabulary models by interpolating weights. *Advances in Neural Information Processing Systems*, 35:29262–29277.

Pavel Izmailov, Dmitrii Podoprikhin, Timur Garipov, Dmitry Vetrov, and Andrew Gordon Wilson. 2019. Averaging weights leads to wider optima and better generalization. *Preprint*, arXiv:1803.05407.

Qiao Jin, Bhuwan Dhingra, Zhengping Liu, William Cohen, and Xinghua Lu. 2019. Pubmedqa: A dataset for biomedical research question answering. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 2567–2577.

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

Mandar Joshi, Eunsol Choi, Daniel Weld, and Luke Zettlemoyer. 2017. TriviaQA: A large scale distantly supervised challenge dataset for reading comprehension. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1601–1611, Vancouver, Canada. Association for Computational Linguistics.

Sanghoon Kim, Dahyun Kim, Chanjun Park, Wonsung Lee, Wonho Song, Yunsu Kim, Hyeonwoo Kim, Yungi Kim, Hyeonju Lee, Jihoo Kim, Changbae Ahn, Seonghoon Yang, Sukyung Lee, Hyunbyung Park, Gyoungjin Gim, Mikyoung Cha, Hwalsuk Lee, and Sunghun Kim. 2024. SOLAR 10.7B: Scaling large language models with simple yet effective depth upscaling. In *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 6: Industry Track)*, pages 23–35, Mexico City, Mexico. Association for Computational Linguistics.

Jonathan Krause, Michael Stark, Jia Deng, and Li Fei-Fei. 2013. 3d object representations for fine-grained categorization. In *Proceedings of the IEEE international conference on computer vision workshops*, pages 554–561.

Viet Lai, Chien Nguyen, Nghia Ngo, Thuat Nguyen, Franck Dernoncourt, Ryan Rossi, and Thien Nguyen. 2023. Okapi: Instruction-tuned large language models in multiple languages with reinforcement learning from human feedback. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 318–327, Singapore. Association for Computational Linguistics.

Yann LeCun. 1998. The mnist database of handwritten digits. *http://yann. lecun. com/exdb/mnist/*.

Sanwoo Lee, Jiahao Liu, Qifan Wang, Jingang Wang, Xunliang Cai, and Yunfang Wu. 2025. Dynamic fisher-weighted model merging via Bayesian optimization. In *Proceedings of the 2025 Conference of the Nations of the Americas Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pages 4923–4935, Albuquerque, New Mexico. Association for Computational Linguistics.

Aitor Lewkowycz, Anders Johan Andreassen, David Dohan, Ethan Dyer, Henryk Michalewski, Vinay Venkatesh Ramasesh, Ambrose Slone, Cem Anil, Imanol Schlag, Theo Gutman-Solo, Yuhuai Wu, Behnam Neyshabur, Guy Gur-Ari, and Vedant Misra. 2022. Solving quantitative reasoning problems with language models. In *Advances in Neural Information Processing Systems*.

Haonan Li, Yixuan Zhang, Fajri Koto, Yifei Yang, Hai Zhao, Yeyun Gong, Nan Duan, and Timothy Baldwin. 2024. CMMLU: Measuring massive multitask language understanding in Chinese. In *Findings of the Association for Computational Linguistics: ACL 2024*, pages 11260–11285, Bangkok, Thailand. Association for Computational Linguistics.

Margaret Li, Suchin Gururangan, Tim Dettmers, Mike Lewis, Tim Althoff, Noah A. Smith, and Luke Zettlemoyer. 2022. Branch-train-merge: Embarrassingly parallel training of expert language models. *Preprint*, arXiv:2208.03306.

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

Mantas Mazeika, Long Phan, Xuwang Yin, Andy Zou, Zifan Wang, Norman Mu, Elham Sakhaee, Nathaniel Li, Steven Basart, Bo Li, David Forsyth, and Dan Hendrycks. 2024. Harmbench: A standardized evaluation framework for automated red teaming and robust refusal. *Preprint*, arXiv:2402.04249.

Yuval Netzer, Tao Wang, Adam Coates, Alessandro Bissacco, Baolin Wu, Andrew Y Ng, and 1 others. 2011. Reading digits in natural images with unsupervised feature learning. In *NIPS workshop on deep learning and unsupervised feature learning*, volume 2011, page 7. Granada.

OpenAI, Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, Red Avila, and 1 others. 2024. Gpt-4 technical report. *Preprint*, arXiv:2303.08774.

Xingyu Qu and Samuel Horváth. 2025. Vanishing feature: Diagnosing model merging and beyond. In *Conference on Parsimony and Learning*, volume 280 of *Proceedings of Machine Learning Research*, pages 1051–1086. PMLR.

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, 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.

Pranav Rajpurkar, Robin Jia, and Percy Liang. 2018. Know what you don't know: Unanswerable questions for SQuAD. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 784–789, Melbourne, Australia. Association for Computational Linguistics.

Siva Reddy, Danqi Chen, and Christopher D. Manning. 2019. CoQA: A conversational question answering challenge. *Transactions of the Association for Computational Linguistics*, 7:249–266.

Paul Röttger, Hannah Kirk, Bertie Vidgen, Giuseppe Attanasio, Federico Bianchi, and Dirk Hovy. 2024. XSTest: A test suite for identifying exaggerated safety behaviours in large language models. In *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pages 5377–5400, Mexico City, Mexico. Association for Computational Linguistics.

Viraj Shah, Nataniel Ruiz, Forrester Cole, Erika Lu, Svetlana Lazebnik, Yuanzhen Li, and Varun Jampani. 2024. Ziplora: Any subject in any style by effectively merging loras. In *ECCV (1)*, pages 422–438.

Xinyue Shen, Zeyuan Chen, Michael Backes, Yun Shen, and Yang Zhang. 2024. "do anything now": Characterizing and evaluating in-the-wild jailbreak prompts on large language models. In *Proceedings of the 2024 on ACM SIGSAC Conference on Computer and Communications Security*, CCS '24, page 1671–1685, New York, NY, USA. Association for Computing Machinery.

Donald Shenaj, Ondrej Bohdal, Mete Ozay, Pietro Zanuttigh, and Umberto Michieli. 2024. Lora.rar: Learning to merge loras via hypernetworks for subject-style conditioned image generation. *CoRR*, abs/2412.05148.

Ken Shoemake. 1985. Animating rotation with quaternion curves. In *Proceedings of the 12th annual conference on Computer graphics and interactive techniques*, pages 245–254.

Johannes Stallkamp, Marc Schlipsing, Jan Salmen, and Christian Igel. 2011. The german traffic sign recognition benchmark: a multi-class classification competition. In *The 2011 international joint conference on neural networks*, pages 1453–1460. IEEE.

George Stoica, Pratik Ramesh, Boglarka Ecsedi, Leshem Choshen, and Judy Hoffman. 2025. Model merging with SVD to tie the knots. In *The Thirteenth International Conference on Learning Representations*.

Anke Tang, Li Shen, Yong Luo, Han Hu, Bo Du, and Dacheng Tao. 2024. Fusionbench: A comprehensive benchmark of deep model fusion. *Preprint*, arXiv:2406.03280.

Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, and 1 others. 2023. Llama 2: Open foundation and fine-tuned chat models. *Preprint*, arXiv:2307.09288.

Fanqi Wan, Longguang Zhong, Ziyi Yang, Ruijun Chen, and Xiaojun Quan. 2025. FuseChat: Knowledge fusion of chat models. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pages 21629–21653, Suzhou, China. Association for Computational Linguistics.

Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel Bowman. 2018. GLUE: A multi-task benchmark and analysis platform for natural language understanding. In *Proceedings of the 2018 EMNLP Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*, pages 353–355, Brussels, Belgium. Association for Computational Linguistics.

Hu Wang, Congbo Ma, Ibrahim Almakky, Ian Reid, Gustavo Carneiro, and Mohammad Yaqub. 2024. Rethinking weight-averaged model-merging. *CoRR*, abs/2411.09263.

Zijing Wang, Xingle Xu, Yongkang Liu, Yiqun Zhang, Peiqin Lin, Shi Feng, Xiaocui Yang, Daling Wang, and Hinrich Schütze. 2025. Why do more experts fail? a theoretical analysis of model merging. *Preprint*, arXiv:2505.21226.

Alex Warstadt, Amanpreet Singh, and Samuel R. Bowman. 2019. Neural network acceptability judgments. *Transactions of the Association for Computational Linguistics*, 7:625–641.

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.

Jianxiong Xiao, Krista A Ehinger, James Hays, Antonio Torralba, and Aude Oliva. 2016. Sun database: Exploring a large collection of scene categories. *International Journal of Computer Vision*, 119(1):3–22.

Prateek Yadav, Derek Tam, Leshem Choshen, Colin A Raffel, and Mohit Bansal. 2023. Ties-merging: Resolving interference when merging models. *Advances in Neural Information Processing Systems*, 36:7093–7115.

Enneng Yang, Zhenyi Wang, Li Shen, Shiwei Liu, Guibing Guo, Xingwei Wang, and Dacheng Tao. 2023. Adamerging: Adaptive model merging for multi-task learning. *CoRR*, abs/2310.02575.

Ming Yin, Jingyang Zhang, Jingwei Sun, Minghong Fang, Hai Helen Li, and Yiran Chen. 2025. LoBAM: LoRA-based backdoor attack on model merging. In *ICLR 2025 Workshop on Navigating and Addressing Data Problems for Foundation Models*.

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

Jeffrey Zhou, Tianjian Lu, Swaroop Mishra, Siddhartha Brahma, Sujoy Basu, Yi Luan, Denny Zhou, and Le Hou. 2023. Instruction-following evaluation for large language models. *Preprint*, arXiv:2311.07911.

Yuyan Zhou, Liang Song, Bingning Wang, and Weipeng Chen. 2024. Metagpt: Merging large language models using model exclusive task arithmetic. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing (EMNLP)*.

Didi Zhu, Yibing Song, Tao Shen, Ziyu Zhao, Jinluan Yang, Min Zhang, and Chao Wu. 2025. REMEDY: Recipe merging dynamics in large vision-language models. In *The Thirteenth International Conference on Learning Representations*.

A Closed-form Solution for Weighted L_1 Loss: Weighted Median

We are given the objective function:

$$\mathcal{L}(\tau_i) = \sum_{m \in \mathcal{M}_i} F_i^{(m)} \cdot \left| \hat{\tau}_i^{(m)} - \tau_i \right|,$$

where $\hat{\tau}_i^{(m)}$ are fixed scalar values and $F_i^{(m)} \geq 0$ are associated weights. Our goal is to find a value of τ_i that minimizes $\mathcal{L}(\tau_i)$.

Piecewise Linearity and Subgradient

Each term $\left| \hat{\tau}_i^{(m)} - \tau_i \right|$ is convex and piecewise linear in τ_i, with a non-differentiable point at $\tau_i = \hat{\tau}_i^{(m)}$. Thus, $\mathcal{L}(\tau_i)$ is convex and piecewise linear overall.

The subgradient with respect to τ_i is given by

$$\frac{\partial}{\partial \tau_i} F_i^{(m)} \cdot \left| \hat{\tau}_i^{(m)} - \tau_i \right| = \begin{cases} -F_i^{(m)} & \text{if } \tau_i < \hat{\tau}_i^{(m)}, \\ [-F_i^{(m)},\, F_i^{(m)}] & \text{if } \tau_i = \hat{\tau}_i^{(m)}, \\ +F_i^{(m)} & \text{if } \tau_i > \hat{\tau}_i^{(m)}. \end{cases}$$

When τ_i does not coincide with any $\hat{\tau}_i^{(m)}$, the function is differentiable and its derivative is given by:

$$\frac{d\mathcal{L}}{d\tau_i} = -\sum_{\hat{\tau}_i^{(m)} < \tau_i} F_i^{(m)} + \sum_{\hat{\tau}_i^{(m)} > \tau_i} F_i^{(m)}.$$

Setting the derivative to zero yields:

$$-\sum_{\hat{\tau}_i^{(m)} < \tau_i} F_i^{(m)} + \sum_{\hat{\tau}_i^{(m)} > \tau_i} F_i^{(m)} = 0,$$

$$\Rightarrow \sum_{\hat{\tau}_i^{(m)} < \tau_i} F_i^{(m)} = \sum_{\hat{\tau}_i^{(m)} > \tau_i} F_i^{(m)} = \frac{1}{2} \sum_m F_i^{(m)}.$$

This condition defines the **weighted median**.

When $\tau^* = \hat{\tau}^m$

For any τ (not necessarily median), let

$$A = \sum_{\hat{\tau}^{(m)} < \tau} F^{(m)}, \quad B = \sum_{\hat{\tau}^{(m)} = \tau} F^{(m)},$$

$$C = \sum_{\hat{\tau}^{(m)} > \tau} F^{(m)},$$

so $A + B + C = T$.

Each term is convex, so

$$\partial \mathcal{L}(\tau) = \sum_{\hat{\tau}^{(m)} < \tau} (-F^{(m)}) + \sum_{\hat{\tau}^{(m)} = \tau} [-F^{(m)}, F^{(m)}] + \sum_{\hat{\tau}^{(m)} > \tau} F^{(m)}$$

$$= [-A + C - B,\ -A + C + B].$$

Hence $0 \in \partial \mathcal{L}(\tau)$ iff

$$-A + C - B \leq 0 \leq -A + C + B \quad \Longleftrightarrow \quad |A - C| \leq B.$$

Showing weighted median satisfies the criteria

A point τ^* is called a *weighted median* if

$$A \leq \frac{T}{2} \quad \text{and} \quad C \leq \frac{T}{2}.$$

We show this implies $|A - C| \leq B$:

1. $A \leq \frac{T}{2}$ implies:

$$A \leq \frac{A + B + C}{2}$$

$$\Rightarrow 2A \leq A + B + C$$

$$\Rightarrow A \leq B + C$$

$$\Rightarrow A - C \leq B.$$

2. $C \leq \frac{T}{2}$ implies:

$$C \leq \frac{A + B + C}{2}$$

$$\Rightarrow 2C \leq A + B + C$$

$$\Rightarrow C \leq A + B$$

$$\Rightarrow C - A \leq B.$$

3. Combining the two, we get:

$$|A - C| \leq B.$$

Since $|A - C| \leq B$ is exactly the condition for optimality, the weighted median minimizes $\mathcal{L}(\tau)$.

B Algorithm

We summarize the method in algorithm form in Algorithm 1.

Algorithm 1 Fisher-Weighted Model Merging with Sign Resolution and Coordinate-wise Top-K Selection

Require: Base parameters $\boldsymbol{\theta}^{(0)} \in \mathbb{R}^N$, candidate models $\{\boldsymbol{\theta}^{(m)}\}_{m=1}^M$, Data $\{\mathcal{D}_m\}_{m=1}^M$, scaling factor λ, Keep ratio - κ

Ensure: Merged model $\boldsymbol{\theta}^{\text{merge}}$

1: Compute task vectors $\boldsymbol{\tau}^{(m)} \leftarrow \boldsymbol{\theta}^{(m)} - \boldsymbol{\theta}^{(0)}$

2: Compute directional sign $s_i \leftarrow \text{sign}\left(\sum_{m=1}^M \tau_i^{(m)}\right)$ for all i ▷ Resolve sign step From TIES

3: **for** each $m = 1$ to M **do**

4: $$\hat{\tau}_i^{(m)} = \begin{cases} \tau_i^{(m)}, & \text{if } \text{sign}(\tau_i^{(m)}) = s_i, \\ 0, & \text{otherwise.} \end{cases}$$

5: $\hat{\boldsymbol{\theta}}^{(m)} \leftarrow \boldsymbol{\theta}^{(0)} + \hat{\tau}^{(m)}$

6: Compute Fisher $\boldsymbol{F}^{(m)}$ via empirical Fisher:

$$F^{(m)} = \mathbb{E}_{x \sim \mathcal{D}_m}\left[\left(\frac{\partial}{\partial \hat{\theta}^{(m)}} \log p(y \mid x; \hat{\boldsymbol{\theta}}^{(m)})\right)^2\right]$$

7: **end for**

8: $K = \lfloor \kappa \cdot M \rfloor$ ▷ Number of models to keep at each coordinate

9: **for** each coordinate i **do**

10: $\delta_i = \min\left(\text{Top-}K\left(\left\{\left|\hat{\tau}_i^{(m)}\right|\right\}_{m=1}^M\right)\right)$ ▷ Minimum deviation for consideration in merging

11: $\mathcal{M}_i := \left\{m \in \{1, \ldots, M\} : \left|\hat{\tau}_i^{(m)}\right| \geq \delta_i\right\}$

12: $\mathcal{L}(\tau_i) = \sum_{m \in \mathcal{M}_i} F_i^{(m)} \cdot \left|\hat{\tau}_i^{(m)} - \tau_i\right|$

13: $\tau_i^* = \arg\min_{\tau_i} \mathcal{L}(\tau_i)$ ▷ Closed form solution in Equation 4

14: $\boldsymbol{\theta}_i^{\text{merge}} \leftarrow \boldsymbol{\theta}_i^{(0)} + \lambda \cdot \tau_i^*$

15: **end for**

16: Return final expert: $\boldsymbol{\theta}^{\text{merge}}$

Table 5: Model Performance on Vision Tasks

Method	SUN397	CARS	RESISC45	Eurosat	SVHN	GTSRB	MNIST	DTD	Average	$\overline{\text{PRR}}$
Skyline(s)	74.86	78.52	95.14	99.07	97.27	98.91	99.58	79.68	90.38	
Task Arithmetic	64.30	61.30	70.56	78.26	73.89	62.77	93.02	51.91	69.50	76.88
TIES	64.97	62.87	**72.29**	76.19	82.19	73.90	96.33	52.61	72.67	80.21
Adamerging	59.63	57.92	71.30	**80.15**	68.98	53.53	97.29	55.21	68.00	75.16
PCB	62.00	61.31	71.79	75.41	85.92	**79.76**	**97.74**	51.38	73.16	80.51
Ours	**65.01**	**66.17**	71.38	76.19	**88.05**	64.33	97.55	**58.35**	**73.38**	**81.22**

C Results on Vision Tasks

Merging Fully fine-tuned CLIP-ViT-B/32 Models For image classification, we evaluate multi-task model merging across eight image classification datasets. Following (DU et al., 2024), we use the CLIP model (Radford et al., 2021) with ViT-B/32 as visual encoders. We have adopted same experimental configurations as Tang et al. (2024). We display our result in Table 5. While certain baseline methods occasionally outperform our approach on individual datasets, our method achieves superior overall performance compared with the leading baselines.

D Implementation Details

A central design consideration in `DRIFT-MEDIAN` is the placement of different components in the merging pipeline. The calculation of the Fisher information matrix constitutes the primary computational bottleneck of our approach. To make the method practical, we decouple operations that require repeated hyperparameter tuning from those that do not. Specifically, the *Sign Resolution* step is performed prior to *Fisher Information Estimation*, since it does not involve any tunable parameters and can be fixed once for all runs. In contrast, the two hyperparameters of our method – keep ratio κ for top-K selection and scaling factor λ —directly

Table 6: Results on Llama-2-7b models; The reported results for CMMLU, GSM8K and HumanEval are absolute values obtained from corresponding evaluation set.

Method	CMMLU	GSM8K	HumanEval	$\overline{\text{PRR}}$
Fine-tuned Models (DU et al., 2024)	38.6	65.6	17.1	-
Model Averaging	35.6	47.8	8.5	71.60
Task Arithmetic	35.5	46.1	10.4	74.35
TIES	36.4	53.4	14	85.86
PCB-Merging	36.4	53.8	16.5	90.93
Fine-tuned models (Ours)	35.2	61.9	16.5	-
Ours	35.8	42.2	19.5	96.02

affect the aggregation and scaling stages. We, therefore, design the method such that these choices come *after* Fisher information estimation. This ensures that once the Fisher matrix is computed, it can be re-used efficiently for any combination of κ and λ without additional estimation overhead.

This design differs from prior work Lee et al. (2025), who perform scaling before Fisher merging and search over λ across different models. While their approach is effective, it was applied primarily to much smaller models, where repeated Fisher estimation is less of a burden. In contrast, our design explicitly targets large-scale LLMs, where recomputing the Fisher matrix even a few times would be prohibitively expensive. By fixing Fisher estimation early and allowing hyperparameter flexibility afterward, DRIFT-MEDIAN achieves both scalability and adaptability.

E Illustration of Coordinate-wise top-K

Illustration of coordinate-wise top-K selection has been provided in Figure 2.

F Results on LLaMA2-7B

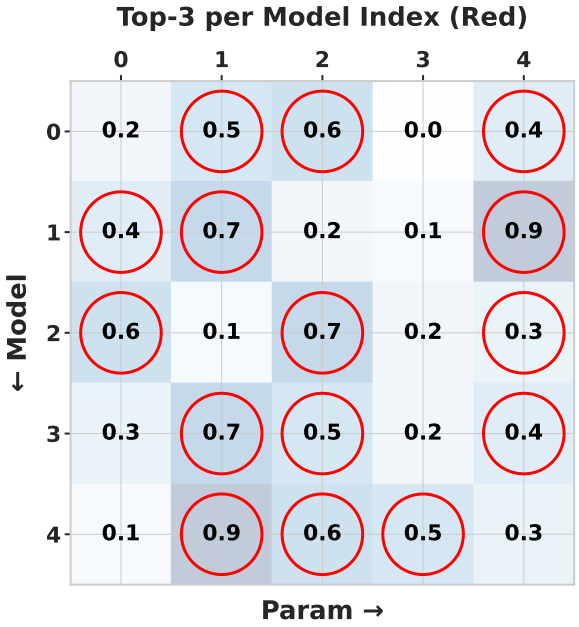
In Table 6, we compare our method with the baseline method PCB-merging (DU et al., 2024). When we evaluate the fine-tuned models in respective task (sixth row), we get slightly different numbers than reported (first row) in the paper. So, for a fair comparison, first five rows in this table are copied from the paper (DU et al., 2024) and then we evaluate the merged methods in terms of mean PRR. In particular, our method outperforms PCB-merging by 5.09%.

G Ablation on Hyperparameter λ

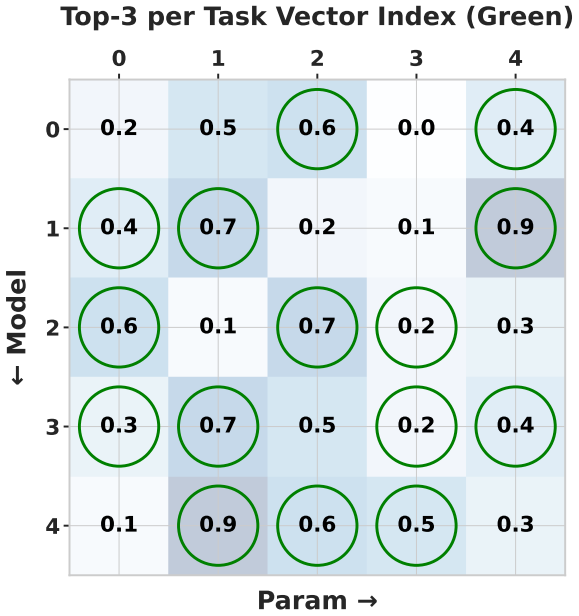
For this ablation, we consider the same experimental setup and tasks given in the Table 2 for merging. We use a fixed keep ratio of 60% in this experiment. In Figure 3, we plot our model performance with respect to DRIFT-MEDIAN hyperparameter λ. It highlights the trade-off between in-domain specialization and out-domain generalization performance of our merged model with respect to λ. Here, $\lambda = 0$ signifies base model. Initially, with increase in λ value, in-domain performance increases, reaching a saturation point then starts to decrease. The out-domain performance decrease with increase in λ. After certain value of λ, merging becomes unstable and performance of merged model drops significantly for in-domain and as well as out-domain. Our intended goal is to maximize performance on the known, in-domain tasks, and the out-domain results are reported primarily for completeness. If out-domain robustness were also an objective, one possible direction would be to adjust the trade-off parameter λ. We provide a sensitivity study on λ and top-K on Figure J and observe good results in nearby zones of the optimal hyperparamters. Detailed descriptions of the specific task configurations for out-domain is given as General Domain paragraph in Appendix H.

H Evaluation Benchmarks

We evaluate large language models (LLMs) across multiple domains using a diverse suite of benchmarks, each with carefully designed test sets. We evaluate DRIFT-MEDIAN on the following benchmark datasets: Minerva (Hendrycks et al., 2021), GSM8K (Cobbe et al., 2021), Harmbench (Mazeika et al., 2024), DAN (Shen et al., 2024), XSTest (Röttger et al., 2024), WildguardTest (Han et al., 2024), IFEval (Zhou et al., 2023), CMMLU (Li et al., 2024), 3 Multilingual Understanding tasks (Lai et al., 2023) (M_ARC, M_MMLU and M_HellaSwag), MBPP+ (Austin et al., 2021), HUMANEVAL+ (Chen et al., 2021), and 7 GLUE



(a) Top-K Selection (TIES)



(b) Top-K Selection (Ours)

Figure 2: (a) In TIES Top-K selection, only a single task vector at index 3, while four are concentrated at indices 1, 2, 4. (b) In contrast, our inter-model selection distributes task vectors more evenly, preventing crowding at certain parameter indices and scarcity at others, thereby ensuring consistent influence per index and avoiding dilution during aggregation.

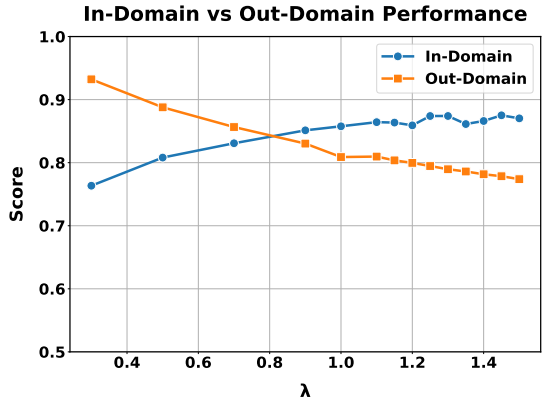


Figure 3: Sensitivity of λ on in-domain and out-domain performance; Initially in-domain performance increases with increase in λ, reaching a saturation point then starts to decrease. The out-domain performance decrease with increase in λ.

(Wang et al., 2018; Warstadt et al., 2019) tasks (QQP, QNLI, RTE, CoLA, MRPC, MNLI and SST-2).

Mathematics. We consider two variants of the GSM8K (Cobbe et al., 2021) test set from the `lm-eval-harness`, namely GSM8K (5-shot) and GSM8K-CoT (8-shot). Since the test items (and gold answers) are identical, but model performance can vary depending on whether direct or chain-of-thought prompting is used, we report the best score across the two settings for each model. The GSM8K test set contains approximately 1.3k grade-school math word problems requiring multi-step reasoning and exact numeric answers. In addition, we include the Minerva Math (Lewkowycz et al., 2022) test set in a 4-shot setting, which consists of STEM-focused quantitative problems curated from the MATH benchmark (Hendrycks et al., 2021).

Multilingual Understanding. For cross-lingual evaluation, we employ translated test sets from three widely used benchmarks: M_ARC, M_MMLU, and M_HellaSwag (Lai et al., 2023). These test sets are direct multilingual extensions of the original English benchmarks, created via high-quality machine translation and covering multiple languages. We restrict evaluation to four representative languages: French (fr), German (de), Russian (ru), and Spanish (es) to assess reasoning and commonsense understanding across diverse linguistic settings. The test sets retain the multiple-choice structure of their English counterparts: M_ARC for science question answering, M_MMLU for

multi-domain knowledge across 57 subjects, and M_HellaSwag for adversarial commonsense reasoning. To evaluate performance on Chinese on Llama2-7b models, we use the CMMLU (Li et al., 2024) benchmark .

Instruction Following. We evaluate using the IFEval (Zhou et al., 2023) test set, which contains 541 prompts covering 25 categories of verifiable instructions. Each prompt specifies explicit and automatically checkable constraints (e.g., output length, language, or formatting). In line with the original protocol, we report both *prompt-level strict accuracy*, which requires that all constraints be satisfied exactly, and *prompt-level loose accuracy*, which allows multiple post-processing transformations of the model output and considers a response correct if any transformed version meets all specified criteria.

Code Generation. We adopt the HumanEval+ (Chen et al., 2021) and MBPP+ (Austin et al., 2021) test sets from the EvalPlus framework, which augment the original HumanEval and MBPP problems with substantially more hidden test cases (approximately 80× more for HumanEval and 35× more for MBPP). We report the *Pass@1* metric across these test sets. For consistency with prior work on PCB merging, we evaluate **Llama-2-7b** using the original HumanEval test set of 164 handwritten programming tasks, ensuring comparability with published results.

Safety and Robustness. To assess safety, we employ several adversarial and red-teaming test sets. The WildGuardTest (Han et al., 2024) set contains ∼5k human-annotated examples from WildGuardMix, labeled across 13 harm categories and evaluated for prompt harmfulness, response harmfulness, and refusal detection. The HarmBench (Mazeika et al., 2024) test suite provides a standardized set of adversarial prompts for automated red-teaming, enabling direct measurement of attack success rates and robust refusal behavior. In addition, we include adversarial jailbreak prompts from the DAN (Do Anything Now) (Shen et al., 2024) family, which are widely used to probe model vulnerabilities in controlled settings. Finally, we use the XSTest (Röttger et al., 2024) benchmark, which comprises 250 safe prompts and 200 unsafe prompts designed to evaluate both over-refusal (failing to answer benign queries) and under-refusal (incorrectly answering harmful queries).

Natural Language Understanding (GLUE). For GPT2, we evaluate models on the GLUE benchmark (Wang et al., 2018). Specifically, we include the following tasks: CoLA (linguistic acceptability), MNLI (multi-genre natural language inference), MRPC (paraphrase detection), QNLI (Question Natural Language Inference), QQP (Quora question pairs), RTE (textual entailment), and SST-2 (Stanford Sentiment Treebank). We use the fine-tuned checkpoints from Fusion-Bench (Tang et al., 2024) library.

Vision Datasets Following PCB merging (DU et al., 2024), we consider multi-task model merging across eight image classification datasets. SUN397 (Xiao et al., 2016) comprises of 397 classes of scene images. Stanford Cars (Krause et al., 2013) is car classification dataset consisting of 196 car classes. RESISC45 (Cheng et al., 2017) consist of 45 classes of remote sensing image scenes. EuroSAT (Helber et al., 2019) includes 10 classes of geo-referenced satellite images. SVHN (Netzer et al., 2011) contains 10 classes of real-world digital classification images. GTSRB (Stallkamp et al., 2011) features 43 classes of traffic signs. MNIST (LeCun, 1998) consists of grayscale handwritten digits across 10 classes. Finally, DTD (Cimpoi et al., 2014)is a texture classification dataset with 47 classes.

General Domain To assess broader reasoning and domain generalization, we include several widely used benchmarks: CoQA (Reddy et al., 2019), MMLU (Hendrycks et al., 2021), PubMedQA (Jin et al., 2019), SQuADv2 (Rajpurkar et al., 2018), and TriviaQA (Joshi et al., 2017). CoQA measures conversational question answering with context-dependent reasoning, while MMLU evaluates multi-domain expert knowledge across 57 subjects. PubMedQA focuses on biomedical question answering, enabling evaluation in a specialized scientific domain. SQuADv2 extends extractive QA with unanswerable questions, testing robustness in distinguishing relevant from irrelevant contexts. TriviaQA probes open-domain QA with a mix of factoid and reasoning-intensive queries. Together, these benchmarks capture general-purpose reasoning, knowledge retrieval, and robustness across domains.

I Hyperparameters and Computation Requirements

To identify suitable hyperparameter configurations for our proposed DRIFT-MEDIAN framework, we initially conducted exploratory searches on GPT-2, owing to its relatively small size and faster training and inference cycles. In this setting, we varied the Top-K parameter from Top-1 through Top-7, and also evaluated the *keep-above-mean* and *keep-above-median* strategies. The sweep over λ was deliberately non-uniform: we first sampled random values across the full interval $[0.3, 3.0]$ and observed that the strongest performance consistently occurred when λ lay in the narrower band of approximately 1.1–1.5. We then performed a denser search within this region using 0.05 increments.

From these experiments, we found that the best configuration on GPT-2 corresponded not to a fixed Top-K selection, but rather to a thresholding strategy, where all coordinates above the mean are retained, combined with a scaling value of $\lambda = 1.35$. For larger LLMs, we subsequently searched in the neighborhood of these optimal GPT-2 values. For Llama-3.1-8B, the best performance was achieved with Top-3 and $\lambda = 1.45$, while for Llama-3.2-3B, the best performance was obtained with Top-3 and $\lambda = 1.30$. For Llama-2-7b, the best configuration was Top-2 with $\lambda = 1.20$. It is important to note that aggressive hyperparameter search is not required. DRIFT-MEDIAN maintains good performance even when λ and K are set near the optimal values, demonstrating that the method is robust and not overly sensitive to hyperparameter choices. For baseline comparisons, we adopt the hyperparameter settings recommended by MergeBench (He et al., 2025b) for Llama-3.2-3B and Llama-3.1-8B models. Specifically, for TIES we use Top-$K = 0.3$ and a scaling factor of $\lambda = 0.4$, for DARE we set the sparsity to 0.9 with $\lambda = 0.4$, and for L&S we use a sparsity of 0.1. Since we were unable to reproduce the with-data version of L&S due to hardware constraints, we report results for the dataless variant.

The most expensive step in our pipeline is the computation of Fisher information matrices. For GPT-2, we use 256 examples, while for larger LLMs we use 1000 examples, which is the maximum available from the validation data. On a single NVIDIA A100 80GB GPU, Fisher estimation for the 8B model takes approximately one hour per domain, which results in about five hours of computation for five tasks. Crucially, this cost is incurred only once per task since Fisher estimation is independent of hyperparameters. Once the Fisher values are computed, they can be reused for all subsequent searches over λ and K, significantly reducing the overhead of iterative experimentation. Moreover, Fisher computation is naturally parallelizable: different tasks' Fisher information can be computed simultaneously, and multi-GPU setups can further accelerate the process. While we did not explore these parallelization strategies in this work, they represent a clear avenue for further speedup in large-scale deployments. Due to large number of benchmarks, we report the results on one run.

J Hyperparameter Sensitivity

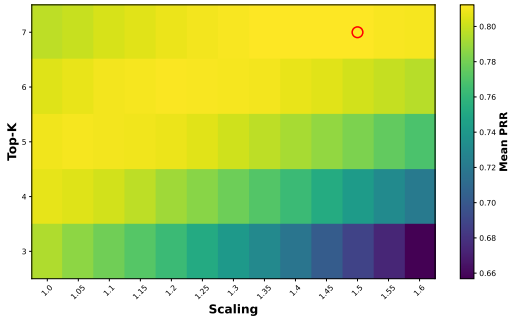


Figure 4: Sensitivity of the hyperparameters K and λ in `DRIFT-MEDIAN` on CLIP based tasks. The best performance is obtained on top-7 models with 1.5 as the scaling factor. The values near to the optimal hyperparameter have similar performance.

To better understand the interaction between the Top-K and the scaling coefficient λ, we conduct a sensitivity study whose results are shown in Figure 4. The heatmap illustrates how different configurations of Top-K and λ jointly affect mean PRR. As Top-K increases, more low-magnitude task deltas are included in the aggregation pool. These small updates, which lie very close to the base model, pull the merged parameter back toward the pretrained initialization. Consequently, configurations with higher Top-K values generally require a higher scaling coefficient λ to counterbalance this pull and ensure that task-relevant updates maintain sufficient influence during aggregation. For this ablation, we consider the same experimental setup and tasks given in the Table 5 for merging.

K Detailed Results

We report the results of individual domains on Table 7. Table 8, Table 9 and Table 10.

L Scalability of DRIFT-MEDIAN

The proposed method should be applicable to larger sized models. Computing the diagonal Fisher information matrix requires only forward–backward passes over a small validation set and does not demand more resources than a brief fine-tuning run. In practice, we did not conduct such experiments due to two limitations: (i) a lack of computational resources to load the models, and (ii) the absence of large, publicly available fine-tuned checkpoints built on the same base model, which makes controlled comparisons difficult. Further, understanding how performance behaves as the number of merged tasks increases is an important question for evaluating the robustness of any parameter-space merging method. However, this behavior does not depend solely on the task count; it is shaped by several interacting factors, including task similarity, parameter-space overlap, sparsity patterns of the task vectors, and the curvature or variance induced by each task. As noted in (Wang et al., 2025), the effective parameter space can saturate as more experts are merged due to Gaussian Width concavity and redundancy constraints. This implies that performance may plateau or even degrade when the additional tasks introduce conflicting or redundant update directions, independent of their absolute number. A faithful study of this phenomenon requires a carefully controlled setting that isolates these effects, which is beyond the scope of the present work.

M Correlation with Task-Vector Magnitude

To better understand whether the degree of parameter deviation in each task influences the final merged performance, we compare the difference between our method and the Skyline(s) baseline with the mean absolute magnitude of the corresponding task vectors. The performance difference is computed as Ours – Skyline, capturing how much accuracy is lost relative to the ideal single-task fine-tuned models. The mean task-vector magnitude reflects the average absolute parameter shift introduced by each task relative to the pretrained backbone. Figure 5 summarizes these two quantities across all eight datasets. The performance differences span a much wider range, from modest degradation (e.g., MNIST) to substantial drops (e.g., GTSRB and RESISC). Importantly, datasets with relatively higher task-vector magnitudes do not consistently exhibit lower deviations from Skyline performance, and tasks with lower magnitudes do not show systematically higher gaps. Overall, this analysis indicates that the magnitude of the task-vector updates is not strongly correlated with the degradation observed when merging models. The variation in performance across datasets is therefore more likely driven by dataset-specific difficulty or inherent conflicts between task objectives rather than by simple differences in the size of the underlying task vectors.

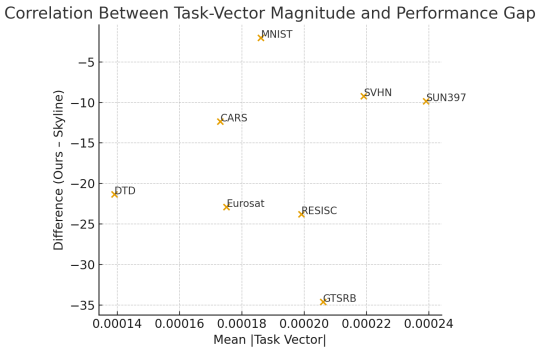


Figure 5: Relationship between task-vector magnitude and performance degradation in the merged model. Despite small variations in the average parameter shift across tasks, no meaningful correlation emerges, indicating that other factors such as dataset difficulty or task conflicts may dominate the observed performance differences.

N Domain Sensitivity of DRIFT-MEDIAN

To further analyze the robustness of DRIFT-MEDIAN, we additionally study the effect of domain mismatch in the Fisher estimation stage. Specifically, we replace the MNIST validation data with KMNIST, a visually distinct digit-recognition dataset where the characters correspond to Japanese cursive hiragana (Clanuwat et al., 2018). Despite the significant visual shift from English numerals, performance across domains remains relatively stable, demonstrating that DRIFT-MEDIAN tolerates moderate domain shifts. We chose KMNIST because both MNIST and KMNIST share the same label space (0 to 9).

In the last row of Table 11, we perform a more extreme modification by replacing SVHN (Street View House Numbers), which contains real-world

Table 7: Model Performance: Mathematics, Multilingual, and Instruction Following Tasks on Llama-3.1-8B

Model Name	GSM8K	Minerva	M_MMLU	M_ARC	M_Hellaswag	IFEval (strict)	IFEval (loose)
Base Model	55.72	18.00	53.28	44.75	63.08	18.70	20.10
Skyline(s)	70.43	33.22	54.02	47.62	65.35	57.70	63.60
Averaging (only layers.)	71.27	25.40	52.68	43.51	65.90	24.80	28.80
Averaging (All)	71.49	25.76	52.70	43.81	65.94	23.80	27.40
Task Arithmetic	75.66	28.02	49.08	41.22	64.03	30.90	37.20
TIES	76.72	28.22	52.26	43.02	65.83	28.80	33.80
DARE	75.06	27.02	47.32	40.86	62.81	30.10	35.90
Fisher merging	67.32	22.60	52.67	43.62	65.66	34.60	39.70
L&S	76.12	28.60	53.49	44.04	65.01	24.20	30.90
Ours	64.67	23.28	47.67	40.22	62.48	40.30	46.20

Table 8: Model Performance: Coding and Safety Tasks on Llama-3.1-8B

Model Name	HumanEval+	MBPP+	WildguardTest	Harmbench	DAN	XSTest
Base Model	31.70	51.30	42.19	24.69	29.67	34.22
Skyline(s)	57.30	54.80	78.37	81.56	71.33	69.11
Averaging (only layers.)	46.30	54.00	58.88	44.06	59.67	60.22
Averaging (All)	44.50	55.60	60.35	41.87	58.00	65.56
Task Arithmetic	49.40	52.40	59.81	48.75	58.33	69.56
TIES	46.30	54.80	61.82	56.25	67.67	64.22
DARE	45.10	52.60	60.35	50.73	51.33	68.44
Fisher merging	44.50	53.20	68.89	57.50	77.00	60.45
L&S	42.70	53.40	51.27	40.31	42.00	49.33
Ours	52.40	53.40	75.83	75.62	78.67	73.33

Table 9: Model Performance: Mathematics, Multilingual, and Instruction Following Tasks on Llama-3.2-3B

Model Name	GSM8K	Minerva	M_MMLU	M_ARC	M_Hellaswag	IFEval (strict)	IFEval (loose)
Base Model	28.66	8.04	45.32	36.75	55.45	20.0	21.6
Skyline(s)	55.50	23.68	44.12	39.19	56.45	37.9	43.8
Averaging (only layers.)	40.86	12.44	45.83	38.94	57.49	21.6	24.8
Averaging (All)	41.77	12.88	45.87	39.00	57.52	19.4	22.9
Task Arithmetic	44.45	15.30	44.27	39.22	57.54	32.0	35.9
TIES	31.46	9.52	45.43	39.66	57.55	27.4	30.7
DARE	43.44	14.78	43.89	39.34	57.36	32.5	36.2
Fisher merging	40.26	12.12	45.98	38.64	57.28	24.6	29.0
L&S	42.76	14.24	45.20	39.73	57.05	21.6	24.8
Ours	40.33	13.94	44.33	38.51	57.10	38.1	44.0

Table 10: Model Performance: Coding and Safety Tasks on Llama-3.2-3B

Model Name	HumanEval+	MBPP+	WildguardTest	Harmbench	DAN	XSTest
Base Model	25.0	39.4	26.70	26.25	29.33	28.67
Skyline(s)	38.4	45.8	85.71	88.75	90.67	38.67
Averaging (only layers.)	31.7	42.9	37.38	35.62	37.33	41.11
Averaging (All)	31.7	41.3	37.92	34.37	37.33	39.33
Task Arithmetic	34.8	45.5	51.67	39.37	34.67	50.22
TIES	32.3	42.3	47.00	40.94	46.00	40.44
DARE	35.4	46.0	49.00	37.19	33.67	54.22
Fisher merging	31.1	42.6	48.06	39.06	48.00	38.00
L&S	31.7	41.3	30.71	27.81	26.67	38.44
Ours	35.4	43.7	56.74	37.81	46.67	50.00

Table 11: Model Performance on Different Domain Data for Fisher Estimation

Validation Data	SUN397	CARS	RESISC45	Eurosat	SVHN	GTSRB	MNIST	DTD	Average	$\overline{\mathbf{PRR}}$
Unchanged	65.01	66.17	71.38	76.19	88.05	64.33	97.55	58.35	73.38	81.22
MNIST → KMNIST	65.17	66.45	71.11	75.96	86.83	64.31	98.06	58.09	73.25	81.09
MNIST → KMNIST & SVHN → MNIST	65.33	66.92	71.98	77.19	77.43	64.65	97.36	58.35	72.40	80.25

RGB street-number images, with MNIST grayscale digits. In this case, we observe a substantial performance drop on SVHN, while the other domains remain consistent. This behavior is expected because SVHN contains cluttered and noisy backgrounds and RGB images whereas MNIST contains grayscale images. Together, these results show that DRIFT-MEDIAN is robust to moderate domain shifts in the Fisher estimation data but can degrade when the substitute domain differs too drastically from the target distribution. Importantly, in Table 2 and Table 3, we use validation data that do not exactly match the downstream evaluation tasks. For example, we use multilingual instruction-following data to compute Fisher information, while the evaluation is performed on tasks such as ARC, HellaSwag, and MMLU. Similarly, for other LLM benchmarks, including MBPP, Humaneval, and GSM8K, there are no official validation sets available. In all the cases, we rely on datasets whose topical focus may be broadly related, but whose style and distributions differ substantially from the downstream tasks. Even though these datasets differ from the evaluation sets, DRIFT-MEDIAN maintains strong performance for all models.

In conclusion, DRIFT-MEDIAN is generally resilient to reasonable domain mismatch and can operate effectively even when the Fisher estimation data and evaluation data come from different distributions. However, extremely mismatched domains such as replacing SVHN with MNIST can negatively impact performance.

O Use of Large Language Models

For transparency, we disclose our use of LLMs during the preparation of this manuscript. ChatGPT (OpenAI et al., 2024) was utilized in a limited capacity as a general-purpose writing assistant for grammatical refinement and sentence paraphrasing. The core research ideas, experimental design, results, and their interpretation were conceived and formulated entirely by the authors. The LLM's role was strictly limited to language refinement and did not contribute to the scientific ideation or analysis presented in this work. The authors have reviewed all content and take full responsibility for the final manuscript.