Task-Aware Model Merging via Fisher-Weighted Median

Anonymous authors
Paper under double-blind review

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 sensitivity-aware model merging approach that incorporates Fisher information and a coordinate-wise importance measure within a weighted median aggregation framework. Comprehensive experiments on several LLMs and CLIP models demonstrate that task-vector interference mitigation and parameter sensitivity are complementary factors in model merging. `DRIFT-MEDIAN` integrates both principles within a unified framework. Across the evaluated settings, this integration improves mean performance retention (PRR), although performance gains may vary across individual tasks. We make the code publicly available at `https://anonymous.4open.science/r/drift-median`.

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 promising solution to these challenges. It combines parameters from independently fine-tuned models that share an 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). This approach eliminates the need for expensive retraining procedures. 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. In this work we focus primarily on merging LLMs, where multiple task-specific fine-tuned models often exist and retraining a unified multitask model may be computationally prohibitive.

Challenges and Motivation: A comprehensive review of current parameter-space merging techniques highlights two distinct approaches. While these approaches address complementary aspects of the model merging challenge, they do not fully exploit the potential benefits of integrating both perspectives. Some methods focus on resolving parameter interference during merging but do not account for parameter sensitivity (Yadav et al., 2023; Yu et al., 2024) to task performance . In contrast, other approaches consider individual

parameter sensitivity while overlooking parameter interference during the merging process (Matena & Raffel, 2022).

An example of the first category is TIES merging (Yadav et al., 2023). TIES addresses two major sources of interference during model merging: sign disagreement and redundant parameters. Sign disagreement occurs when task vectors contain opposing directional updates that cancel each other during averaging, while redundant parameters correspond to uninformative updates that dilute the impact of more significant parameter 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 & Raffel, 2022), incorporates parameter sensitivity through Fisher information weighting. It formulates the problem as maximizing the joint likelihood of the models' posterior distributions over parameters. The method further shows that parameter averaging is equivalent to approximating each model's posterior using an isotropic Gaussian distribution. 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 may benefit from jointly considering : (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 explained in Figure 2b 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 analyze the contribution of each component in the proposed framework and observe that the combination of interference reduction and sensitivity-aware aggregation improves overall performance retention. Key contributions of our proposed framework are as follows:

- **Fisher-weighted median aggregation:** We propose `DRIFT-MEDIAN`, a parameter- space merging method where we formulate aggregation as a Fisher-weighted L_1 optimization and employ a weighted median estimator for combining task vectors 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 in mathematics, coding, multilingual reasoning, safety, vision tasks, and instruction show that `DRIFT-MEDIAN` achieves an overall improved mean performance compared to previous methods in the evaluated settings.

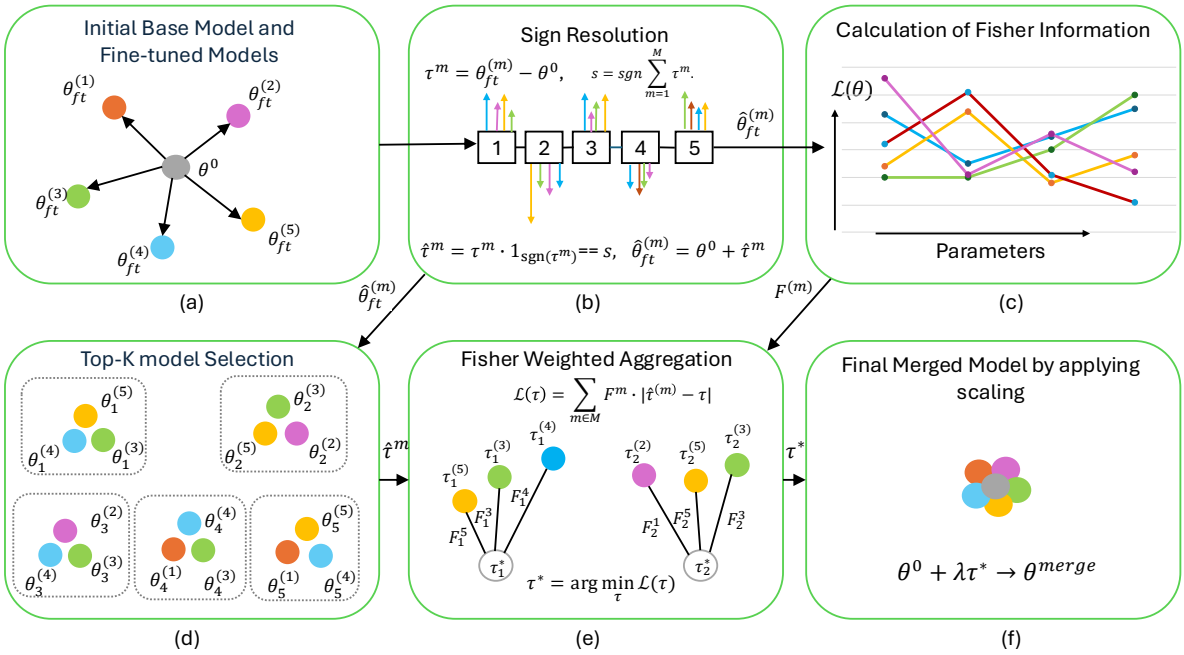
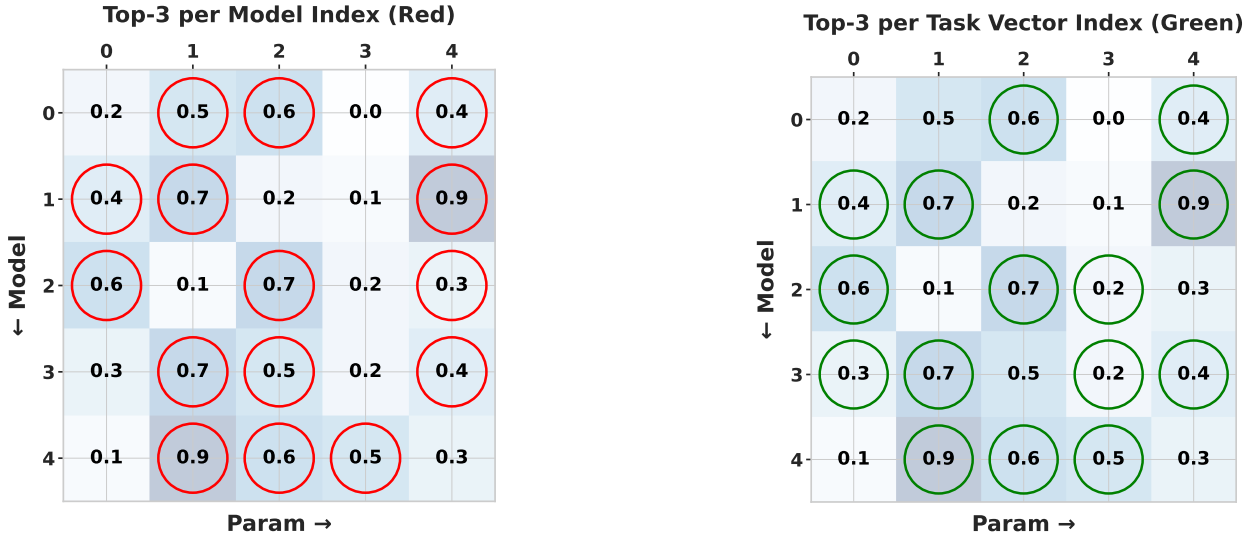


Figure 1: Overview of the steps involved in the proposed model merging approach. (a) θ^0 and $\theta_{ft}^{(m)}$ (with specific color) represent base model and different fine-tuned models respectively. (b) In the sign resolution step, task vectors that agree in sign are retained to compute the fine-tuned 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. Note that the color signifies the top-K finetuned or expert models that can vary based on the co-ordinate. (e) Finally, Fisher-weighted aggregation yields τ^*, followed by scaling (f) to obtain the merged parameters θ^{merge}.

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 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). 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



(a) Top-K Selection (TIES)

(b) Top-K Selection (Ours)

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

performance over individual model selection. Similar improvements through weight averaging have been reported by Ilharco et al. (2022b); Matena & Raffel (2022); Li et al. (2022). More sophisticated interpolation methods have emerged to address specific challenges in parameter combination. AdaMerging (Yang et al., 2024b) formulates model merging as an unsupervised optimization problem and learns task-wise or layer-wise coefficients by minimizing prediction entropy on unlabeled multi-task data, allowing automatic balancing of task contributions. Regmean++ Jin et al. (2023); Nguyen et al. (2025) compute optimal layer-wise merging weights by minimizing prediction discrepancies between the merged model and candidate models, while explicitly modeling intra- and cross-layer dependencies to better capture interactions across model representations.

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. T-Switch (Qi et al., 2024) explores a complementary direction by binarizing task vectors using activation and polarity switches, enabling efficient storage and reducing parameter conflicts while preserving merging performance. 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. EMR-Merging (Huang et al., 2024) adopts a different perspective by first electing a unified model among candidate models and then introducing lightweight task-specific modulators consisting of masks

and rescalers to align parameter directions and magnitudes. This approach avoids additional training while enabling flexible alignment across multiple fine-tuned models.

Sensitivity-Aware, Domain-Specific and Sparse Model Fusion Methods : Fisher merging (Matena & 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. Uncertainty-based gradient matching (Daheim et al., 2024) investigated the role of gradient alignment in merging and the limitations of parameter averaging through gradient mismatch and proposes an uncertainty-weighted scheme that improves merging robustness by reducing gradient inconsistencies across models. 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. Representation Surgery (Yang et al., 2024a) addresses representation bias introduced during merging by learning lightweight task-specific modules that correct discrepancies between the merged model's representations and those of individual 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 $s_i = \text{sign}\Big(\sum_{m=1}^M \tau_i^{(m)}\Big)$. 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 use an importance weighting scheme based on empirical Fisher information (Matena & 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, $\theta^{(0)}$ denotes the reference initialization point (e.g., the pre-trained parameters), and $\log p\Big(y \mid x; \hat{\theta}^{(m)}\Big)$ denotes the log-likelihood of the model parameterized by the sign-aligned parameters $\hat{\theta}^{(m)}$. Intuitively, $F_i^{(m)}$ measures the sensitivity of the model's predictive likelihood to perturbations in parameter θ_i. A larger value of $F_i^{(m)}$ indicates that even small changes in θ_i significantly affect the likelihood, suggesting that the parameter is particularly important 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 & 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. We illustrate this in Figure 2.

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 cutoff 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.

3.4 Fisher-weighted Aggregation

After sign resolution and coordinate-wise Top-K filtering, we obtain, for each coordinate i, a retained set of sign-consistent task-vector entries. We use $z_i^{(m)} = \hat{\tau}_i^{(m)}$ for the sign-filtered task-vector value, $r_i^{(m)} = \mathbf{1}[m \in \mathcal{M}_i]$ for the Top-K retention indicator, and $w_i^{(m)} = F_i^{(m)} r_i^{(m)}$ for the effective Fisher weight after Top-K filtering. Thus, non-retained entries have zero effective weight.

The coordinate-wise aggregation problem can be formulated as

$$\tau_i^\star = \arg\min_{\tau_i} \sum_{m=1}^{M} w_i^{(m)} \left| z_i^{(m)} - \tau_i \right| = \arg\min_{\tau_i} \sum_{m \in \mathcal{M}_i} F_i^{(m)} \left| \hat{\tau}_i^{(m)} - \tau_i \right|. \tag{2}$$

This aggregation step is implemented in Algorithm 1: lines 1–4 compute task vectors and apply sign resolution, lines 8–11 determine the retained coordinate-wise Top-K set $\mathcal{M}_i$, line 12 forms the Fisher-weighted absolute-deviation objective, line 13 computes its weighted-median solution, and line 14 returns $\theta_i^{\text{merge}} = \theta_i^{(0)} + \lambda \tau_i^\star$. Therefore, the mathematical variable $\tau_i^\star$ in Eq. equation 2 is exactly the coordinate-wise merged displacement used by the implementation.

The use of an L_1 objective is the main difference from Fisher-weighted averaging. A Fisher-weighted L_2 objective gives a Fisher-weighted mean, which can be strongly affected by extreme task-vector values. In contrast, Eq. equation 2 gives a Fisher-weighted median, which is more robust when some retained task updates are large but weakly supported by the other models.

Closed-form Fisher-weighted Median The minimizer of the Fisher-weighted absolute-deviation loss corresponds to a weighted median estimator. Specifically, the optimal merged displacement τ_i^* at coordinate i is given by the Fisher-weighted median of the retained updates $\{\hat{\tau}_i^{(m)} : m \in M_i\}$, where each retained update is weighted by its Fisher value $F_i^{(m)}$. Formally, τ_i^* is any value for which the total Fisher weight of updates smaller than τ_i^* and the total Fisher weight of updates larger than τ_i^* are both at most half of the total retained Fisher weight:

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

In implementation, this condition is realized by sorting the retained values $\hat{\tau}_i^{(m)}$, accumulating their Fisher weights $F_i^{(m)}$ in sorted order, and selecting the first value whose cumulative weight reaches at least $\frac{1}{2} \sum_{m \in M_i} F_i^{(m)}$. Unlike Fisher-weighted mean, which can be strongly affected by extreme task-vector values, the Fisher-weighted median prevents an extreme update from dominating the aggregate unless it is supported by sufficiently large Fisher weight. As a result, the merged displacement is both importance-aware and resistant to spurious task updates. The weighted median solution follows from classical results on L_1 optimization and selection algorithms (Aho & Hopcroft, 1974; Blum et al., 1973; Gurwitz, 1990), and we present the derivation in Appendix A.

Scaling Since sign pruning and Top-K filtering reduce the effective magnitude of the merged task vector, a rescaling step is applied after Fisher-weighted median aggregation 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 constructed as $\theta^{\text{merge}} = \theta^{(0)} + \tau^{\text{merge}}$. 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 C covers implementation details and design choices.

Table 1: 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)	90.98	96.71	30.09	90.87	72.64	76.26
Model Averaging (incl. embed)	91.56	96.67	31.86	87.22	74.87	76.44
Task Arithmetic	83.51	97.70	18.52	82.12	69.65	70.30
TIES	92.63	97.60	29.56	87.46	78.10	77.07
DARE	83.72	97.60	17.56	84.17	69.36	70.48
Fisher Merging	80.28	**98.32**	46.99	92.64	92.28	82.10
PCB	95.36	95.77	27.33	87.29	82.26	77.60
Localize & Stitch	**98.44**	96.51	29.74	85.21	59.58	73.90
Ours	83.69	98.27	**62.96**	**93.44**	**83.79**	**84.43**

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) and Fisher merging (Matena & Raffel, 2022). Hyperparameters detail are given in Appendix I.

Models and Datasets We primarily evaluate our approach on large language models, including Llama family of models with various number of model parameters (Llama-3.1-8B, Llama-3.2-3B) and GPT-2. We additionally include CLIP experiments to illustrate that the method can extend beyond text models. 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 Llama Based Models: For generation tasks we consider Llama-3.1-8B & Llama-3.2-3B, we report experimental results in Table 1 & Table 2. 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

Table 2: 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)	59.53	100.76	27.40	88.23	53.12	65.81
Task Arithmetic	59.79	100.63	24.24	90.95	55.77	66.28
TIES	63.87	100.73	28.46	89.50	55.70	67.65
DARE	60.17	100.56	24.28	87.56	55.71	65.66
Fisher Merging	60.87	100.49	41.24	86.87	64.87	70.87
PCB Merging	64.88	101.12	24.70	89.70	55.99	67.28
Localize & Stitch	**75.76**	**101.13**	38.24	89.79	54.10	71.80
Ours	67.90	100.67	**59.38**	**92.51**	**69.23**	**77.94**

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 E. 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 2.33% and 6.14% 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. This phenomenon typically occurs when complementary knowledge from different task models reinforces each other during merging, enabling the merged model to slightly outperform the individual task-specific model on certain benchmarks. Therefore, this may result in a PRR exceeding 100 in certain cases. Although the proposed method achieves the highest mean PRR, certain individual tasks, particularly Mathematics and Multilingual reasoning in some settings, exhibit lower retention compared to specific baselines. This behavior reflects the trade-off introduced by median-based aggregation, which prioritizes robustness to conflicting updates and balanced multi-task retention over maximizing individual task-specific dominance. In contrast, mean-based aggregation can sometimes favor tasks with dominant parameter updates, but at the cost of increased sensitivity to conflicting or noisy task vectors. Consequently, the median aggregation provides a more stable trade-off across tasks while improving overall performance retention.

We provide a sensitivity study on λ and top-K on Appendix J and observe good results in the nearby zones of the optimal hyperparameters. Further, we provide an in-domain vs out-domain performance tradeoff by varying scaling factor λ in Appendix D and observed that in-domain performance increases with increase in scaling while it is the opposite for out-domain tasks.

4.3 Task-vector geometry and task-wise variation

To better understand the task-wise behavior of DRIFT-MEDIAN, we analyze the task vectors of the Llama-3.2-3B specialist models relative to the base model. For all coordinate-level statistics, we consider a coordinate i is active for task m if $|\tau_i^{(m)}| > \epsilon_{\text{abs}} + \epsilon_{\text{rel}}|\theta_i^{(0)}|$, where we use $\epsilon_{\text{abs}} = 10^{-8}$ and $\epsilon_{\text{rel}} = 10^{-6}$ (we chose some low but arbitary number). The task-vector magnitude is computed as $\|\tau^{(m)}\|_2$, which measures how far task m moves from the base model. The relative drift from the base model is computed as $\text{RelDrift}(m) = \frac{\|\tau^{(m)}\|_2}{\|\theta^{(0)}\|_2} \times 100$. Let $A^{(m)}$ denote the active-coordinate set: $A^{(m)} = \left\{ i : |\tau_i^{(m)}| > \epsilon_{\text{abs}} + \epsilon_{\text{rel}}|\theta_i^{(0)}| \right\}$.

The active-coordinate percentage is computed as $\text{Active}(m) = \frac{|A^{(m)}|}{N} \times 100$, where N is the total number of analyzed coordinates. The average cosine similarity of task m with the remaining task vectors is

$$\text{AvgCos}(m) = \frac{1}{M-1} \sum_{n \neq m} \frac{\langle \tau^{(m)}, \tau^{(n)} \rangle}{\|\tau^{(m)}\|_2 \|\tau^{(n)}\|_2} \times 100 \tag{3}$$

Next, we analyze task-wise directional agreement. We consider the top 0.1% highest-magnitude coordinates of $\tau^{(m)}$, denoted by $T^{(m)}$. For each coordinate $i \in T^{(m)}$, we define the same-direction agreement count as

$$a_i^{(m)} = \sum_{n \neq m} \mathbf{1}\left[i \in A^{(n)} \wedge \operatorname{sign}(\tau_i^{(n)}) = \operatorname{sign}(\tau_i^{(m)}) \right]. \tag{4}$$

The average agreement count is $\mathrm{AvgAgree}(m) = \frac{1}{|T^{(m)}|} \sum_{i \in T^{(m)}} a_i^{(m)}$. The percentage of top coordinates that agree in direction with at least k other tasks is $\mathrm{Agree}_{\geq k}(m) = \frac{1}{|T^{(m)}|} \sum_{i \in T^{(m)}} \mathbf{1}[a_i^{(m)} \geq k] \times 100$. For pairwise analysis, we compute the cosine similarity between task vectors m and n as $\mathrm{Cos}(m,n) = \frac{\langle \tau^{(m)}, \tau^{(n)} \rangle}{\|\tau^{(m)}\|_2 \|\tau^{(n)}\|_2} \times 100$. We also compute directional active-coordinate coverage as $C(m \leftarrow n) = \frac{|A^{(m)} \cap A^{(n)}|}{|A^{(m)}|} \times 100$. Thus, for row task m and column task n, an upper-triangular entry a/b in Table 4 denotes $C(m \leftarrow n)/C(n \leftarrow m)$.

Table 3: Task-level geometry for Llama-3.2-3B specialist models. Δ_{best} denotes DRIFT-MEDIAN PRR minus the best competing merging method for the corresponding task. $\|\tau\|_2$ denotes the task-vector norm. Rel. Drift denotes relative drift. Act. denotes the percentage of active coordinates. $\overline{\cos}$ denotes average cosine similarity with all other task vectors. $\overline{\text{agree}}$ denotes the average same-direction agreement count over top-magnitude coordinates. $\text{agree}_{\geq 1}$ and $\text{agree}_{\geq 2}$ denote the percentage of top-magnitude coordinates that agree in direction with at least one or at least two other task vectors, respectively.

Task	PRR	Δ_{best}	$\|\tau\|_2$	Rel. Drift (%)	Act. (%)	$\overline{\cos}$	$\overline{\text{agree}}$	$\text{agree}_{\geq 1}$	$\text{agree}_{\geq 2}$
Mathematics	67.90	-7.86	16.25	1.54	88.56	1.02	1.20	72.63	35.99
Multilingual	100.67	-0.46	6.38	0.60	18.74	0.41	1.76	78.34	58.50
Instruction	59.38	+18.14	10.24	0.97	82.61	3.98	1.41	81.90	44.74
Coding	92.51	+1.56	6.18	0.58	72.08	2.85	1.20	73.16	36.33
Safety	69.23	+4.36	9.40	0.89	80.51	4.21	1.30	77.02	40.61

Table 4: Pairwise task-vector cosine and active-coordinate coverage for Llama-3.2-3B. Lower triangular entries report cosine similarity between task vectors in percent. Upper triangular entries report $C(m \leftarrow n)/C(n \leftarrow m)$ in percent, where $C(m \leftarrow n)$ is the percentage of active coordinates of task m that are also active in task n.

Task	Mathematics	Multilingual	Instruction	Coding	Safety
Mathematics	–	20.81 / 98.33	84.13 / 90.19	74.32 / 91.32	82.21 / 90.43
Multilingual	0.10	–	97.45 / 22.11	95.67 / 24.88	97.08 / 22.60
Instruction	0.63	0.70	–	75.61 / 86.66	83.23 / 85.40
Coding	2.51	0.44	3.73	–	85.07 / 76.15
Safety	0.86	0.40	**10.88**	4.71	–

The results explain why Mathematics behaves differently from Instruction and Safety. Mathematics has the largest task-vector magnitude, the highest relative drift, and the highest active-coordinate percentage. This indicates that the math specialist moves farthest from the base model and changes the largest portion of the parameter space. However, this large update shows weak directional agreement with other task vectors. Its average cosine similarity is only 1.02%, and Table 4 shows that, although Mathematics has high active-coordinate coverage with Instruction, Coding, and Safety, the corresponding cosine similarities remain small. In other words, other tasks often modify the same coordinates as Mathematics, but not in a consistently aligned direction.

In contrast, Instruction and Safety task-vector magnitudes, although relatively large, show stronger directional agreement with other tasks. Instruction has the highest $\text{Agree}_{\geq 1}$ among the high-active tasks, while Safety has the highest average cosine similarity. The pairwise table further shows that the strongest cosine similarity is between Instruction and Safety, indicating that these two task vectors are mutually directionally aligned. Consequently, DRIFT-MEDIAN preserves Instruction and Safety updates more effectively. Overall, the analysis suggests that DRIFT-MEDIAN is most effective when important task updates are directionally aligned across task vectors, rather than merely being large in magnitude.

Table 5: Model Performance on Vision Tasks; Results are reported as raw accuracy values, except for the PRR columns.

Method	SUN397	CARS	RESISC45	Eurosat	SVHN	GTSRB	MNIST	DTD	Average	Min. PRR	$\overline{\text{PRR}}$
Baseline	63.14	59.78	60.67	45.93	31.63	32.53	48.26	43.88	48.23	32.88	54.95
Skyline(s)	74.97	78.31	95.19	99.04	97.27	98.91	99.58	79.73	90.38	-	-
Task Arithmetic	64.39	61.51	70.52	80.44	73.89	62.77	93.02	51.60	69.77	63.46	77.16
TIES	65.02	62.84	72.57	79.00	82.19	**73.90**	96.33	52.71	73.07	66.22	80.63
PCB	63.94	62.21	72.05	**83.26**	84.86	76.13	97.32	52.34	74.01	65.64	81.50
Adamerging	59.08	57.13	71.14	81.56	71.42	57.21	97.29	54.95	68.60	57.84	75.71
Ours	**65.43**	**65.58**	**73.54**	80.48	**86.91**	67.73	**97.34**	**57.08**	**74.28**	**68.48**	**82.10**

4.4 Performance on Image Classification Tasks

Merging Fully fine-tuned CLIP-ViT-B/32 Models Although `DRIFT-MEDIAN` is primarily designed for LLMs, we conduct this analysis on CLIP-based vision models for clearer and more controlled evaluation. Unlike LLM benchmarks, which involve heterogeneous metrics such as accuracy, pass@1, and refusal rate, vision tasks provide a unified evaluation metric across datasets. This allows for a more direct comparison of performance across tasks. Additionally, CLIP models are relatively smaller and a larger number of fully fine-tuned checkpoints are readily available, making them well-suited for systematic analysis of merging behavior. For image classification, we evaluate multi-task model merging across eight datasets. We use the CLIP model (Radford et al., 2021) with ViT-B/32 as the visual encoder, and adopt the same experimental setup as Tang et al. (2024). Results are shown in Table 5.

While `DRIFT-MEDIAN` achieves improved mean PRR across evaluated tasks, we observe that certain tasks experience larger degradation after merging than others. Understanding the conditions under which such imbalance occurs provides useful insight into the limitations of parameter-space model merging.

Performance Imbalance and Task-Specialized Updates Although `DRIFT-MEDIAN` improves the overall PRR on CLIP vision tasks, the gains are not uniform across all datasets. To better understand this task-wise variation, we analyze whether degradation after merging is related to how specialized each fine-tuned model is for its own dataset. Let $S(i, j)$ denote the performance of the model fine-tuned on task i when evaluated on dataset j. The diagonal entry $S(j, j)$ corresponds to the performance of the task-specific model on its own dataset, while the off-diagonal entries capture cross-task transfer from models fine-tuned on other datasets. We define the *Cross-Model Performance Gap* for dataset j as

$$\text{Gap}(j) = S(j, j) - \frac{1}{M-1} \sum_{i \neq j} S(i, j), \tag{5}$$

where the second term is the average performance on dataset j obtained by the other fine-tuned models. This gap measures how isolated a task is with respect to transfer from the remaining models. A larger value indicates that the task-specific model performs much better than the other fine-tuned models on the same dataset, suggesting that the corresponding task vector contains more specialized updates that are less consistently shared across tasks.

Table 6 reports the Cross-Model Performance Gap together with the degradation of `DRIFT-MEDIAN` relative to the corresponding fine-tuned model. We also include the baseline accuracy and normalize both the baseline score and the gap by the fine-tuned model accuracy. This relative view is useful because the same absolute gap can have different implications depending on the strength of the task-specific model.

The results indicate that degradation is more strongly associated with task specialization than with baseline performance alone. The correlation between relative gap and degradation is positive (Pearson $r \approx 0.46$), showing that tasks whose fine-tuned behavior is more isolated tend to suffer larger degradation after merging. In contrast, the correlation between relative baseline performance and degradation is weakly negative (Pearson $r \approx -0.14$), suggesting that baseline accuracy by itself does not explain the observed imbalance. When the relative baseline score and the relative gap are considered jointly in a two-variable linear regres-

Table 6: Relationship between Cross-Model Performance Gap and performance degradation after merging.

Dataset	Baseline	Fine-tuned Model	Other Mean	Gap	Baseline Rel. (%)	Gap Rel. (%)	Degradation (%)
SUN397	63.14	74.97	48.52	26.45	84.22	35.28	12.73
CARS	59.78	78.31	42.10	36.21	76.34	46.24	16.26
RESISC45	60.67	95.19	35.20	60.00	63.74	63.03	22.74
EuroSAT	45.93	99.04	30.80	68.24	46.38	68.90	18.74
SVHN	31.63	97.27	30.62	66.64	32.52	68.51	10.51
GTSRB	32.53	98.91	24.66	74.25	32.89	75.07	31.52
MNIST	48.26	99.58	46.79	52.79	48.46	53.01	2.25
DTD	43.88	79.73	33.24	46.50	55.04	58.32	28.42

Table 7: Ablation study for CLIP `DRIFT-MEDIAN` on vision tasks. Results are reported as raw accuracy values, except for the PRR columns.

Method	SUN397	CARS	RESISC45	EuroSAT	SVHN	GTSRB	MNIST	DTD	Average	Min. PRR	$\overline{\text{PRR}}$
Default	65.43	65.58	73.54	80.48	87.04	67.74	97.34	57.07	74.28	68.48	82.10
Mean instead of median	63.92	62.64	71.70	80.19	87.43	70.29	97.47	53.24	73.36	66.78	80.89
w/o scaling	64.13	62.89	72.10	82.85	83.54	68.63	96.62	53.62	73.05	67.24	80.60
w/o sign election	64.94	66.01	71.17	81.44	88.37	63.90	97.49	57.71	73.88	64.60	81.71
Fisher Alternatives											
w/o Fisher	59.96	59.01	68.06	80.85	79.03	69.60	96.26	49.68	70.31	62.31	77.38
Gradient magnitude	61.10	61.04	69.84	82.30	80.80	70.06	96.85	52.02	71.75	65.24	79.04
Task vector magnitude	53.36	51.19	62.54	73.70	85.47	76.12	97.44	41.86	67.71	52.50	73.98

sion, they explain a larger fraction of the degradation pattern ($R^2 \approx 0.52$). This suggests that performance degradation depends both on how much the fine-tuned model improves over the base model and on how much of that improvement is shared with other fine-tuned models.

This trend is consistent with the aggregation mechanism of `DRIFT-MEDIAN`. The Fisher-weighted L_1 objective favors robust aggregation under conflicting updates. As a result, highly task-specific updates can be attenuated when they are not sufficiently corroborated by other task vectors. For example, GTSRB and DTD show large degradation despite strong fine-tuned performance, which is consistent with their relatively large specialization gaps. In contrast, MNIST has a large relative gap but very small degradation, suggesting that high task specialization alone is not sufficient to explain degradation; the final outcome also depends on whether the task-specific updates are compatible with the aggregation and scaling process.

Table 7 further clarifies which components are responsible for the final performance. Replacing the median with the mean reduces the average accuracy from 74.28 to 73.36 and lowers the mean PRR from 82.10 to 80.89. This indicates that the median aggregation contributes to robustness, especially when some task vectors contain large but weakly supported updates. Removing the scaling step also reduces mean PRR to 80.60, showing that scaling is useful for recovering adaptation strength after sign filtering and coordinate-wise selection. Removing sign election produces a smaller drop in mean PRR, from 82.10 to 81.71, but it reduces the minimum PRR more substantially, from 68.48 to 64.60. This suggests that sign election is especially important for avoiding poor worst-task behavior, even when the average remains relatively stable.

The largest degradation occurs when Fisher weighting is removed or replaced by simpler importance signals. Without Fisher (we take median of the weights with uniform weighting), mean PRR drops from 82.10 to 77.38, and the average accuracy drops from 74.28 to 70.31. Replacing Fisher with gradient magnitude improves over the no-Fisher variant but still remains below the default setting, with mean PRR 79.04. Using task-vector magnitude as the importance signal performs worst among the tested variants, with mean PRR 73.98 and minimum PRR 52.50. This shows that simply prioritizing large parameter updates is not sufficient and can even amplify task-specific directions that do not transfer well across datasets. Fisher weighting therefore provides a more reliable sensitivity signal for balancing task-specific preservation and cross-task robustness. We include a discussion on the reason for choosing Fisher as a sensitivity metric in Appendix O.

Overall, the gap analysis and the ablation study provide complementary perspective. The gap analysis explains when `DRIFT-MEDIAN` is more vulnerable, namely when a task depends on highly specialized updates

that are weakly supported by other fine-tuned models. The ablation study shows why the full method remains effective overall: Fisher weighting, median aggregation, sign consistency, and scaling each contribute to balancing robustness and retention. Among these components, Fisher weighting has the strongest effect on mean performance, while sign election is particularly useful for improving worst-task stability.

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 show that `DRIFT-MEDIAN` improves mean PRR compared with several existing parameter-space merging approaches under the evaluated settings. However, tasks with highly specialized or weakly shared parameter updates may exhibit lower retention after merging, highlighting an inherent trade-off between robustness to interference and preservation of dominant task-specific behaviors. Potential directions for future work include the use of *dynamic* hyperparameters, where λ and κ adapt across models or even layers.

Limitations

DRIFT-MEDIAN introduces additional computational overhead due to Fisher estimation. In our implementation, diagonal Fisher estimation requires approximately 17 minutes per task for Llama-3.2-3B and approximately 40 minutes per task for Llama-3.1-8B on a single GPU. Although these statistics can be cached and reused, the preprocessing cost remains higher than simpler methods such as TIES or DARE. Model merging also inherently reduces performance relative to the original task-specific fine-tuned models, including safety-specialized models and guardrails. Consequently, the merged models (using any merging method) may become more susceptible to harmful prompts compared to the original standalone safety model. We therefore caution readers that merged safety capabilities should not be treated as equivalent to dedicated safety fine-tuning. Finally, the tasks considered in this work exhibit relatively high levels of task conflict and directional disagreement. DRIFT-MEDIAN is intentionally conservative and avoids being dominated by extreme or weakly supported updates. While this improves robustness in conflicting multitask settings, it may be suboptimal when task vectors are already well aligned. In such cases, methods such as TIES, DARE, or PCB may be preferable, and mean-based aggregation may outperform median aggregation since the mean better preserves cosine similarity with aligned task vectors.

Broader Impact

This work focuses on improving robustness and effectiveness in parameter-space model merging. Model merging provides several practical advantages, including reduced retraining cost, improved parameter efficiency, and the ability to combine independently fine-tuned capabilities into a unified model without requiring access to the original training data. Such approaches can make the deployment and adaptation of large language models more computationally accessible and environmentally efficient.

At the same time, parameter-space merging may introduce broader concerns related to privacy, memorization, safety, and model security. Since merged models integrate information from multiple fine-tuned models, there exists the possibility that sensitive or memorized information encoded in individual models could persist after merging. Similarly, merged models may inherit vulnerabilities related to model extraction or unintended memorization behaviors already present in constituent models. These concerns are not unique to the proposed method and broadly apply to parameter-based model merging approaches. Since `DRIFT-MEDIAN` operates solely on model parameters and does not require access to the original training data during merging, it does not explicitly increase exposure to private data. Nevertheless, systematic evaluation of privacy and memorization behavior in merged models remains an important direction for future work.

Another important consideration is the potential dilution of safety-aligned behaviors during model merging. As discussed in Section 4.3, DRIFT-MEDIAN employs support-aware median aggregation, where parameter updates that receive weak support across models may be attenuated during aggregation. While this mechanism improves robustness against conflicting parameter updates, it may also affect highly specialized updates, including safety- or alignment-related parameter changes that are not broadly shared across tasks. In deployment settings, this could potentially weaken certain safety-specific behaviors when merged alongside multiple non-safety expert models.

We emphasize that this challenge is not specific to DRIFT-MEDIAN and is likely relevant to model merging methods more broadly. However, the Fisher-weighted aggregation employed in our framework partially mitigates this issue by assigning greater importance to parameters that exhibit higher sensitivity to task performance. Furthermore, our empirical evaluation demonstrates competitive performance on safety-related benchmarks after merging. Future work could further strengthen safety preservation through mechanisms such as safety-aware weighting schemes, protected alignment subspaces, constrained aggregation strategies, or explicit preservation of safety-critical parameters during merging.

References

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

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

Jacob Austin, Augustus Odena, Maxwell Nye, Maarten Bosma, Henryk Michalewski, David Dohan, Ellen Jiang, Carrie Cai, Michael Terry, Quoc Le, and Charles Sutton. Program synthesis with large language models, 2021. URL https://arxiv.org/abs/2108.07732.

Loubna Ben Allal, Niklas Muennighoff, Logesh Kumar Umapathi, Ben Lipkin, and Leandro von Werra. A framework for the evaluation of code generation models. https://github.com/bigcode-project/bigcode-evaluation-harness, 2022.

Stella Biderman, Hailey Schoelkopf, Lintang Sutawika, Leo Gao, Jonathan Tow, Baber Abbasi, Alham Fikri Aji, Pawan Sasanka Ammanamanchi, Sidney Black, Jordan Clive, Anthony DiPofi, Julen Etxaniz, Benjamin Fattori, Jessica Zosa Forde, Charles Foster, Jeffrey Hsu, Mimansa Jaiswal, Wilson Y. Lee, Haonan Li, Charles Lovering, Niklas Muennighoff, Ellie Pavlick, Jason Phang, Aviya Skowron, Samson Tan, Xiangru Tang, Kevin A. Wang, Genta Indra Winata, François Yvon, and Andy Zou. Lessons from the trenches on reproducible evaluation of language models, 2024. URL https://arxiv.org/abs/2405.14782.

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

Mark Chen, Jerry Tworek, Heewoo Jun, Qiming Yuan, Henrique Ponde de Oliveira Pinto, Jared Kaplan, Harri Edwards, Yuri Burda, Nicholas Joseph, Greg Brockman, Alex Ray, Raul Puri, Gretchen Krueger, Michael Petrov, Heidy Khlaaf, Girish Sastry, Pamela Mishkin, Brooke Chan, Scott Gray, Nick Ryder, Mikhail Pavlov, Alethea Power, Lukasz Kaiser, Mohammad Bavarian, Clemens Winter, Philippe Tillet, Felipe Petroski Such, Dave Cummings, Matthias Plappert, Fotios Chantzis, Elizabeth Barnes, Ariel Herbert-Voss, William Hebgen Guss, Alex Nichol, Alex Paino, Nikolas Tezak, Jie Tang, Igor Babuschkin, Suchir Balaji, Shantanu Jain, William Saunders, Christopher Hesse, Andrew N. Carr, Jan Leike, Josh Achiam, Vedant Misra, Evan Morikawa, Alec Radford, Matthew Knight, Miles Brundage, Mira Murati, Katie Mayer, Peter Welinder, Bob McGrew, Dario Amodei, Sam McCandlish, Ilya Sutskever, and Wojciech Zaremba. Evaluating large language models trained on code, 2021. URL https://arxiv.org/abs/2107.03374.

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

Jiho Choi, Donggyun Kim, Chanhyuk Lee, and Seunghoon Hong. Revisiting weight averaging for model merging. *CoRR*, abs/2412.12153, 2024. URL https://doi.org/10.48550/arXiv.2412.12153.

Leshem Choshen, Elad Venezian, Noam Slonim, and Yoav Katz. Fusing finetuned models for better pretraining. *CoRR*, abs/2204.03044, 2022. URL https://doi.org/10.48550/arXiv.2204.03044.

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

Tarin Clanuwat, Mikel Bober-Irizar, Asanobu Kitamoto, Alex Lamb, Kazuaki Yamamoto, and David Ha. Deep learning for classical japanese literature, 2018.

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. Training verifiers to solve math word problems, 2021. URL https://arxiv.org/abs/2110.14168.

Nico Daheim, Thomas Möllenhoff, Edoardo Ponti, Iryna Gurevych, and Mohammad Emtiyaz Khan. Model merging by uncertainty-based gradient matching. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=D7KJmfEDQP.

Aladin Djuhera, Swanand Kadhe, Farhan Ahmed, Syed Zawad, and Holger Boche. 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*, 2025. URL https://openreview.net/forum?id=FJUkCeFoMB.

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. Parameter competition balancing for model merging. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*, 2024. URL https://openreview.net/forum?id=l5SbrtvSRS.

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, , ry DeVito, Zef Rosnbrick, Zhaoduo Wen, Zhenyu Yang, Zhiwei Zhao, et al. The llama 3 herd of models, 2024. URL https://arxiv.org/abs/2407.21783.

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

Hasan Abed Al Kader Hammoud, Umberto Michieli, Fabio Pizzati, Philip Torr, Adel Bibi, Bernard Ghanem, and Mete Ozay. Model merging and safety alignment: One bad model spoils the bunch. In Yaser Al-Onaizan, Mohit Bansal, and Yun-Nung Chen (eds.), *Findings of the Association for Computational Linguistics: EMNLP 2024*, pp. 13033–13046, Miami, Florida, USA, November 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.findings-emnlp.762. URL https://aclanthology.org/2024.findings-emnlp.762/.

Seungju Han, Kavel Rao, Allyson Ettinger, Liwei Jiang, Bill Yuchen Lin, Nathan Lambert, Yejin Choi, and Nouha Dziri. Wildguard: Open one-stop moderation tools for safety risks, jailbreaks, and refusals of llms. In A. Globerson, L. Mackey, D. Belgrave, A. Fan, U. Paquet, J. Tomczak, and C. Zhang (eds.), *Advances in Neural Information Processing Systems*, volume 37, pp. 8093–8131. Curran Associates, Inc., 2024. URL https://proceedings.neurips.cc/paper_files/paper/2024/file/0f69b4b96a46f284b726fbd70f74fb3b-Paper-Datasets_and_Benchmarks_Track.pdf.

Yifei He, Yuzheng Hu, Yong Lin, Tong Zhang, and Han Zhao. Localize-and-stitch: Efficient model merging via sparse task arithmetic. *Transactions on Machine Learning Research*, 2025a. ISSN 2835-8856. URL https://openreview.net/forum?id=9CWU8Oi86d.

Yifei He, Siqi Zeng, Yuzheng Hu, Rui Yang, Tong Zhang, and Han Zhao. Mergebench: A benchmark for merging domain-specialized llms, 2025b. URL https://arxiv.org/abs/2505.10833.

Patrick Helber, Benjamin Bischke, Andreas Dengel, and Damian Borth. 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, 2019.

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. In *Thirty-fifth Conference on Neural Information Processing Systems Datasets and Benchmarks Track (Round 2)*, 2021. URL https://openreview.net/forum?id=7Bywt2mQsCe.

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

Chenyu Huang, Peng Ye, Tao Chen, Tong He, Xiangyu Yue, and Wanli Ouyang. Emr-merging: tuning-free high-performance model merging. In *Proceedings of the 38th International Conference on Neural Information Processing Systems*, NIPS '24, Red Hook, NY, USA, 2024. Curran Associates Inc. ISBN 9798331314385.

Gabriel Ilharco, Marco Túlio Ribeiro, Mitchell Wortsman, Suchin Gururangan, Ludwig Schmidt, Hannaneh Hajishirzi, and Ali Farhadi. Editing models with task arithmetic. *CoRR*, abs/2212.04089, 2022a. URL https://doi.org/10.48550/arXiv.2212.04089.

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

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. URL https://openreview.net/forum?id=6t0Kwf8-jrj.

Pavel Izmailov, Dmitrii Podoprikhin, Timur Garipov, Dmitry Vetrov, and Andrew Gordon Wilson. Averaging weights leads to wider optima and better generalization, 2019. URL https://arxiv.org/abs/1803.05407.

Qiao Jin, Bhuwan Dhingra, Zhengping Liu, William Cohen, and Xinghua Lu. 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)*, pp. 2567–2577, 2019.

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. URL https://openreview.net/forum?id=FCnohuR6AnM.

Mandar Joshi, Eunsol Choi, Daniel Weld, and Luke Zettlemoyer. TriviaQA: A large scale distantly supervised challenge dataset for reading comprehension. In Regina Barzilay and Min-Yen Kan (eds.), *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 1601–1611, Vancouver, Canada, July 2017. Association for Computational Linguistics. doi: 10.18653/v1/P17-1147. URL https://aclanthology.org/P17-1147/.

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. SOLAR 10.7B: Scaling large language models with simple yet effective depth up-scaling. In Yi Yang, Aida Davani, Avi Sil, and Anoop Kumar (eds.), *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 6: Industry Track)*, pp. 23–35, Mexico City, Mexico, June 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.naacl-industry.3. URL https://aclanthology.org/2024.naacl-industry.3/.

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

Viet Lai, Chien Nguyen, Nghia Ngo, Thuat Nguyen, Franck Dernoncourt, Ryan Rossi, and Thien Nguyen. Okapi: Instruction-tuned large language models in multiple languages with reinforcement learning from human feedback. In Yansong Feng and Els Lefever (eds.), *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pp. 318–327, Singapore, December 2023. Association for Computational Linguistics. doi: 10.18653/v1/2023.emnlp-demo.28. URL https://aclanthology.org/2023.emnlp-demo.28/.

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

Sanwoo Lee, Jiahao Liu, Qifan Wang, Jingang Wang, Xunliang Cai, and Yunfang Wu. Dynamic fisher-weighted model merging via Bayesian optimization. In Luis Chiruzzo, Alan Ritter, and Lu Wang (eds.), *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)*, pp. 4923–4935, Albuquerque, New Mexico, April 2025. Association for Computational Linguistics. ISBN 979-8-89176-189-6. doi: 10.18653/v1/2025.naacl-long.254. URL https://aclanthology.org/2025.naacl-long.254/.

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. Solving quantitative reasoning problems with language models. In Alice H. Oh, Alekh Agarwal, Danielle Belgrave, and Kyunghyun Cho (eds.), *Advances in Neural Information Processing Systems*, 2022. URL https://openreview.net/forum?id=IFXTZERXdM7.

Margaret Li, Suchin Gururangan, Tim Dettmers, Mike Lewis, Tim Althoff, Noah A. Smith, and Luke Zettlemoyer. Branch-train-merge: Embarrassingly parallel training of expert language models, 2022. URL https://arxiv.org/abs/2208.03306.

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

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. Harmbench: A standardized evaluation framework for automated red teaming and robust refusal, 2024. URL https://arxiv.org/abs/2402.04249.

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

The-Hai Nguyen, Dang Huu-Tien, Takeshi Suzuki, and Le-Minh Nguyen. Regmean++: Enhancing effectiveness and generalization of regression mean for model merging, 2025. URL https://arxiv.org/abs/2508.03121.

OpenAI, Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, Red Avila, et al. Gpt-4 technical report, 2024. URL https://arxiv.org/abs/2303.08774.

Biqing Qi, Fangyuan Li, Zhen Wang, Junqi Gao, Dong Li, Peng Ye, and Bowen Zhou. Less is more: Efficient model merging with binary task switch, 2024. URL https://arxiv.org/abs/2412.00054.

Xingyu Qu and Samuel Horváth. Vanishing feature: Diagnosing model merging and beyond. In Beidi Chen, Shijia Liu, Mert Pilanci, Weijie Su, Jeremias Sulam, Yuxiang Wang, and Zhihui Zhu (eds.), *Conference on Parsimony and Learning*, volume 280 of *Proceedings of Machine Learning Research*, pp. 1051–1086. PMLR, 24–27 Mar 2025. URL https://proceedings.mlr.press/v280/qu25a.html.

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

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

Pranav Rajpurkar, Robin Jia, and Percy Liang. Know what you don't know: Unanswerable questions for SQuAD. In Iryna Gurevych and Yusuke Miyao (eds.), *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pp. 784–789, Melbourne, Australia, July 2018. Association for Computational Linguistics. doi: 10.18653/v1/P18-2124. URL https://aclanthology.org/P18-2124/.

Siva Reddy, Danqi Chen, and Christopher D. Manning. CoQA: A conversational question answering challenge. *Transactions of the Association for Computational Linguistics*, 7:249–266, 2019. doi: 10.1162/tacl_a_00266. URL https://aclanthology.org/Q19-1016/.

Paul Röttger, Hannah Kirk, Bertie Vidgen, Giuseppe Attanasio, Federico Bianchi, and Dirk Hovy. XSTest: A test suite for identifying exaggerated safety behaviours in large language models. In Kevin Duh, Helena Gomez, and Steven Bethard (eds.), *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pp. 5377–5400, Mexico City, Mexico, June 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.naacl-long.301. URL https://aclanthology.org/2024.naacl-long.301/.

Viraj Shah, Nataniel Ruiz, Forrester Cole, Erika Lu, Svetlana Lazebnik, Yuanzhen Li, and Varun Jampani. Ziplora: Any subject in any style by effectively merging loras. In *ECCV (1)*, pp. 422–438, 2024. URL https://doi.org/10.1007/978-3-031-73232-4_24.

Xinyue Shen, Zeyuan Chen, Michael Backes, Yun Shen, and Yang Zhang. "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, pp. 1671–1685, New York, NY, USA, 2024. Association for Computing Machinery. ISBN 9798400706363. doi: 10.1145/3658644.3670388. URL https://doi.org/10.1145/3658644.3670388.

Donald Shenaj, Ondrej Bohdal, Mete Ozay, Pietro Zanuttigh, and Umberto Michieli. Lora.rar: Learning to merge loras via hypernetworks for subject-style conditioned image generation. *CoRR*, abs/2412.05148, 2024. URL https://doi.org/10.48550/arXiv.2412.05148.

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

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

George Stoica, Pratik Ramesh, Boglarka Ecsedi, Leshem Choshen, and Judy Hoffman. Model merging with SVD to tie the knots. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=67X93aZHII.

Anke Tang, Li Shen, Yong Luo, Han Hu, Bo Du, and Dacheng Tao. Fusionbench: A comprehensive benchmark of deep model fusion, 2024. URL https://arxiv.org/abs/2406.03280.

Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, et al. Llama 2: Open foundation and fine-tuned chat models, 2023. URL https://arxiv.org/abs/2307.09288.

Fanqi Wan, Longguang Zhong, Ziyi Yang, Ruijun Chen, and Xiaojun Quan. FuseChat: Knowledge fusion of chat models. In Christos Christodoulopoulos, Tanmoy Chakraborty, Carolyn Rose, and Violet Peng (eds.), *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pp. 21629–21653, Suzhou, China, November 2025. Association for Computational Linguistics. ISBN 979-8-89176-332-6. doi: 10.18653/v1/2025.emnlp-main.1096. URL https://aclanthology.org/2025.emnlp-main.1096/.

Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel Bowman. GLUE: A multi-task benchmark and analysis platform for natural language understanding. In Tal Linzen, Grzegorz Chrupała, and Afra Alishahi (eds.), *Proceedings of the 2018 EMNLP Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*, pp. 353–355, Brussels, Belgium, November 2018. Association for Computational Linguistics. doi: 10.18653/v1/W18-5446. URL https://aclanthology.org/W18-5446/.

Hu Wang, Congbo Ma, Ibrahim Almakky, Ian Reid, Gustavo Carneiro, and Mohammad Yaqub. Rethinking weight-averaged model-merging. *CoRR*, abs/2411.09263, 2024. URL https://doi.org/10.48550/arXiv.2411.09263.

Zijing Wang, Xingle Xu, Yongkang Liu, Yiqun Zhang, Peiqin Lin, Shi Feng, Xiaocui Yang, Daling Wang, and Hinrich Schütze. Why do more experts fail? a theoretical analysis of model merging, 2025. URL https://arxiv.org/abs/2505.21226.

Alex Warstadt, Amanpreet Singh, and Samuel R. Bowman. Neural network acceptability judgments. *Transactions of the Association for Computational Linguistics*, 7:625–641, 2019. doi: 10.1162/tacl_a_00290. URL https://aclanthology.org/Q19-1040/.

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*, pp. 23965–23998. PMLR, 2022.

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

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

Prateek Yadav, Tu Vu, Jonathan Lai, Alexandra Chronopoulou, Manaal Faruqui, Mohit Bansal, and Tsendsuren Munkhdalai. What matters for model merging at scale? *Transactions on Machine Learning Research*, 2025. ISSN 2835-8856. URL https://openreview.net/forum?id=9sbetmvNpW.

Enneng Yang, Zhenyi Wang, Li Shen, Shiwei Liu, Guibing Guo, Xingwei Wang, and Dacheng Tao. Adamerging: Adaptive model merging for multi-task learning. *CoRR*, abs/2310.02575, 2023. URL https://doi.org/10.48550/arXiv.2310.02575.

Enneng Yang, Li Shen, Zhenyi Wang, Guibing Guo, Xiaojun Chen, Xingwei Wang, and Dacheng Tao. Representation surgery for multi-task model merging, 2024a. URL https://arxiv.org/abs/2402.02705.

Enneng Yang, Zhenyi Wang, Li Shen, Shiwei Liu, Guibing Guo, Xingwei Wang, and Dacheng Tao. Adamerging: Adaptive model merging for multi-task learning. In *The Twelfth International Conference on Learning Representations*, 2024b. URL https://openreview.net/forum?id=nZP6NgD3QY.

Ming Yin, Jingyang Zhang, Jingwei Sun, Minghong Fang, Hai Helen Li, and Yiran Chen. LoBAM: LoRA-based backdoor attack on model merging. In *ICLR 2025 Workshop on Navigating and Addressing Data Problems for Foundation Models*, 2025. URL https://openreview.net/forum?id=NHOEz72fip.

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 *Forty-first International Conference on Machine Learning*, 2024.

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. URL https://arxiv.org/abs/2311.07911.

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

Didi Zhu, Yibing Song, Tao Shen, Ziyu Zhao, Jinluan Yang, Min Zhang, and Chao Wu. REMEDY: Recipe merging dynamics in large vision-language models. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=iX7eHHE5Tx.

A Weighted Median Solution for the Fisher-weighted L_1 Objective

We prove the weighted-median solution used in Section 3.4. Fix a coordinate i, and simplify notation by writing $z_m = \hat{\tau}_i^{(m)}$ and $w_m = F_i^{(m)}$ for each retained model $m \in \mathcal{M}_i$. The coordinate-wise Fisher-weighted absolute-deviation objective is

$$\mathcal{L}(\tau_i) = \sum_{m \in \mathcal{M}_i} w_m |z_m - \tau_i|, \qquad w_m \geq 0.$$

Here, the values z_m are fixed scalar task-vector entries and the weights w_m are their corresponding Fisher importance values. Since each absolute-value term is convex, $\mathcal{L}(\tau_i)$ is also convex. Therefore, τ_i^* minimizes $\mathcal{L}$ if and only if $0 \in \partial \mathcal{L}(\tau_i^*)$.

For any candidate value τ_i, define $A(\tau_i) = \sum_{z_m < \tau_i} w_m$, $B(\tau_i) = \sum_{z_m = \tau_i} w_m$, and $C(\tau_i) = \sum_{z_m > \tau_i} w_m$. These are the total Fisher weights strictly to the left of τ_i, exactly at τ_i, and strictly to the right of τ_i, respectively. The subgradient of the objective is

$$\partial \mathcal{L}(\tau_i) = [A(\tau_i) - C(\tau_i) - B(\tau_i),\ A(\tau_i) - C(\tau_i) + B(\tau_i)].$$

Thus, the optimality condition $0 \in \partial \mathcal{L}(\tau_i)$ is equivalent to

$$A(\tau_i) - C(\tau_i) - B(\tau_i) \leq 0 \leq A(\tau_i) - C(\tau_i) + B(\tau_i),$$

or, equivalently, $|A(\tau_i) - C(\tau_i)| \leq B(\tau_i)$.

Now let $T = A(\tau_i) + B(\tau_i) + C(\tau_i) = \sum_{m \in \mathcal{M}_i} w_m$ be the total retained Fisher weight. A weighted median is any value τ_i^* satisfying

$$A(\tau_i^*) \leq \frac{T}{2} \qquad \text{and} \qquad C(\tau_i^*) \leq \frac{T}{2}.$$

These two inequalities imply $A(\tau_i^*) - C(\tau_i^*) \leq B(\tau_i^*)$ and $C(\tau_i^*) - A(\tau_i^*) \leq B(\tau_i^*)$. Hence, $|A(\tau_i^*) - C(\tau_i^*)| \leq B(\tau_i^*)$, which is exactly the subgradient optimality condition above. Therefore, any weighted median minimizes the Fisher-weighted L_1 objective.

In implementation, this means that for each coordinate i, we sort the retained task-vector values $z_m = \hat{\tau}_i^{(m)}$, accumulate their Fisher weights $w_m = F_i^{(m)}$ in sorted order, and select the first value whose cumulative weight reaches at least $T/2$. We summarize the method in algorithm form in Algorithm 1.

B Results on GPT-2 based GLUE text Classification Tasks

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 8. 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` achieves better mean PRR compared with existing merging baselines while maintaining overall balance across the tasks in terms of minimum PRR.

We further 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.

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

Input: 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

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

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

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

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

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

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

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

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

end for

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

for each coordinate i **do**

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

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

$\tau_i^* \leftarrow \text{WeightedMedian}\left(\{\hat{\tau}_i^{(m)}\}_{m \in M_i}, \{F_i^{(m)}\}_{m \in M_i}\right)$ ▷ Closed-form solution in Equation 2

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

end for

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

Table 8: Results on GPT-2. The reported values correspond to absolute scores (except for min. PRR and $\overline{\text{PRR}}$) obtained on the validation set, since the test set is not publicly accessible. The results are mostly deterministic, fisher merging and the proposed method can vary depending upon the examples, we report average of three runs in such cases.

Method	COLA	MNLI	MRPC	QNLI	QQP	RTE	SST2	Mean	min. PRR	$\overline{\text{PRR}}$
Fine-tuned Models	76.80	82.08	80.39	88.27	89.64	65.34	91.17	81.96	-	-
Averaging	55.03	55.08	50.98	57.61	76.70	44.77	52.52	56.10	57.61	68.45
Task Arithmetic	68.74	68.56	69.61	70.49	81.82	47.29	83.60	70.02	72.38	84.98
TIES	68.55	70.45	69.36	69.69	82.54	46.93	81.08	69.80	71.82	84.74
Localize & Stitch	67.50	76.53	51.47	63.74	83.42	48.38	50.80	63.12	55.72	77.17
Fisher Merging	50.66 (± 9.06)	53.74 (± 3.18)	38.07 (± 2.26)	62.21 (± 2.67)	79.37 (± 2.21)	50.54 (± 0.63)	75.92 (± 21.67)	58.64 (± 0.56)	47.36 (± 2.82)	71.21 (± 0.60)
Ours	67.88 (± 0.14)	71.23 (± 0.01)	63.60 (± 0.17)	72.87 (± 0.16)	82.39 (± 0.00)	49.82 (± 0.00)	85.72 (± 0.08)	70.50 (± 0.02)	76.24 (± 0.00)	85.58 (± 0.02)

C 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 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.

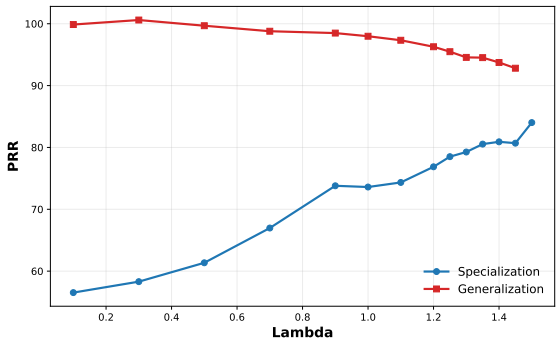


Figure 3: Sensitivity of λ on in-domain and out-domain performance; Initially in-domain performance increases with increase in λ, reaching a saturation point then starts to decrease. The out-domain performance decrease with increase in λ.

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.

D 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 Appendix 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. Note that this graph has been generated by evaluating on test set, on denser sweep than the merging experiments. Since we consider best selected checkpoint on the validation set for all the merging experiments, the good performance here may not reflect in the tables as that may not have been selected as the best checkpoint on validation.

E Detailed Results

We report the results for individual domains in Table 9, Table 10, and Table 11 for the Llama-3.1-8B based models, and in Table 12, Table 13, and Table 14 for the Llama-3.2-3B based models.

F Fisher Estimation Details

For the LLM-based experiments, we estimate the diagonal empirical Fisher information matrix using the corresponding task-specific validation datasets provided by MergeBench.[1] Specifically, we use `math_val`,

[1] `https://huggingface.co/MergeBench/datasets`

Table 9: Scores for mathematics and multilingual ARC tasks on Llama-3.1-8B models.

Method	Mathematics		Multilingual			
	GSM8K	Minerva	ARC-de	ARC-es	ARC-fr	ARC-ru
Metric	Exact Match	Math Verify	Acc. Norm.	Acc. Norm.	Acc. Norm.	Acc. Norm.
Skyline	68.84	38.88	46.19	50.43	49.96	45.77
Averaging (incl. embed.)	73.16	29.88	42.94	44.96	46.62	41.15
Averaging (excl. embed.)	72.86	29.60	42.26	44.96	46.45	41.92
Task Arithmetic	67.63	26.74	43.71	46.41	46.45	43.28
TIES	73.46	30.54	43.80	45.73	46.62	42.43
DARE	67.25	27.12	43.54	46.67	46.54	42.94
Fisher Merging	65.73	25.30	44.57	47.09	47.82	43.46
PCB	76.12	31.16	42.09	43.85	45.42	41.83
Localize & Stitch	76.65	33.26	42.86	45.56	45.68	42.34
Ours	68.16	26.58	44.31	46.75	48.08	43.71

Table 10: Scores for multilingual HellaSwag and MMLU tasks on Llama-3.1-8B models.

Method	Multilingual							
	HellaSwag-de	HellaSwag-es	HellaSwag-fr	HellaSwag-ru	mMMLU-de	mMMLU-es	mMMLU-fr	mMMLU-ru
Metric	Acc. Norm.	Acc. Norm.	Acc. Norm.	Acc. Norm.	Acc.	Acc.	Acc.	Acc.
Skyline	62.82	69.84	67.66	60.57	53.90	56.37	56.07	52.33
Averaging (incl. embed.)	63.55	70.39	68.37	61.56	52.60	54.86	54.70	51.08
Averaging (excl. embed.)	63.58	70.55	68.43	61.59	52.67	54.81	54.54	51.31
Task Arithmetic	62.82	69.59	67.44	60.67	53.51	56.16	55.83	52.44
TIES	63.47	69.84	68.12	61.03	53.34	56.33	55.36	52.11
DARE	62.71	69.44	67.38	60.71	53.68	56.04	55.73	52.27
Fisher Merging	63.78	70.32	68.66	61.45	53.30	55.62	55.21	51.37
PCB	62.65	69.18	67.59	60.61	52.56	55.14	54.48	51.10
Localize & Stitch	62.41	68.96	67.14	60.16	53.14	55.86	55.01	51.57
Ours	63.65	70.36	68.31	61.26	53.08	55.97	55.14	51.55

Table 11: Scores for instruction following, coding, and safety tasks on Llama-3.1-8B based models.

Method	Instruction		Coding		Safety			
	IFEval	IFEval	HumanEval+	MBPP+	DAN	HarmBench	WildGuard	XSTest
Metric	Prompt Loose Acc.	Prompt Strict Acc.	Pass@1	Pass@1	1 − ASR	1 − ASR	1 − harm rate	Acc.
Skyline	53.23	45.47	58.54	54.50	82.00	80.94	78.10	69.56
Averaging (incl. embed.)	16.82	14.60	43.29	54.76	59.00	44.06	60.08	66.89
Averaging (excl. embed.)	16.45	13.31	47.56	54.76	58.00	44.38	58.75	62.44
Task Arithmetic	9.98	8.32	39.02	53.17	60.67	46.88	55.94	52.22
TIES	15.90	13.31	43.29	55.03	68.67	52.19	62.62	58.44
DARE	9.61	7.76	40.85	53.70	60.67	45.62	54.74	53.56
Fisher Merging	25.14	21.26	48.78	55.56	80.67	70.00	71.30	64.67
PCB	14.60	12.38	44.51	53.70	74.00	60.31	63.68	57.56
Localize & Stitch	16.08	13.31	42.07	53.70	43.00	41.56	51.94	47.33
Ours	33.27	28.84	50.00	55.29	69.67	57.19	66.62	65.56

Table 12: Scores for mathematics and multilingual ARC tasks on Llama-3.2-3B models.

Method	Mathematics		Multilingual			
	GSM8K	Minerva	ARC-de	ARC-es	ARC-fr	ARC-ru
Metric	Exact Match	Math Verify	Acc. Norm.	Acc. Norm.	Acc. Norm.	Acc. Norm.
Skyline	60.05	27.76	39.18	42.05	41.57	36.01
Averaging (excl. embed.)	41.47	13.88	37.64	40.77	41.15	38.07
Task Arithmetic	42.30	13.64	37.47	40.60	41.15	37.72
TIES	44.12	15.06	37.64	40.85	41.23	37.47
DARE	42.15	13.92	37.13	40.77	41.15	37.64
Fisher Merging	42.08	14.34	37.04	40.60	41.32	37.47
PCB	44.73	15.34	38.84	41.79	41.57	36.95
Localize & Stitch	50.80	18.58	38.32	42.14	42.17	36.95
Ours	46.93	16.00	37.72	40.68	40.72	37.55

Table 13: Scores for multilingual HellaSwag and MMLU tasks on Llama-3.2-3B models.

Method	Multilingual							
	HellaSwag-de	HellaSwag-es	HellaSwag-fr	HellaSwag-ru	mMMLU-de	mMMLU-es	mMMLU-fr	mMMLU-ru
Metric	Acc. Norm.	Acc. Norm.	Acc. Norm.	Acc. Norm.	Acc.	Acc.	Acc.	Acc.
Skyline	54.60	60.44	59.13	53.10	46.20	48.30	47.29	43.19
Averaging (excl. embed.)	54.97	60.97	59.61	53.36	47.20	48.94	48.25	44.48
Task Arithmetic	54.94	60.93	59.70	53.28	47.37	49.11	48.17	44.44
TIES	55.16	61.12	59.81	53.40	47.27	49.00	48.21	44.41
DARE	54.84	60.88	59.71	53.16	47.30	49.12	48.30	44.50
Fisher Merging	55.00	61.05	59.60	53.26	46.98	49.20	48.09	44.61
PCB	55.53	61.43	60.16	53.42	47.09	48.92	48.10	43.94
Localize & Stitch	55.51	61.62	60.30	53.41	47.03	48.58	47.80	43.99
Ours	55.63	61.60	59.94	53.80	46.69	49.00	47.86	44.30

Table 14: Scores for instruction following, coding, and safety tasks on Llama-3.2B based models.

Method	Instruction		Coding		Safety			
	IFEval	IFEval	HumanEval+	MBPP+	DAN	HarmBench	WildGuard	XSTest
Metric	Prompt Loose Acc.	Prompt Strict Acc.	Pass@1	Pass@1	1 − ASR	1 − ASR	1 − harm rate	Acc.
Skyline	45.88	38.63	39.02	45.77	91.00	89.06	85.85	38.89
Averaging (excl. embed.)	13.12	10.72	32.32	42.86	37.67	34.06	36.05	35.33
Task Arithmetic	11.46	9.61	33.54	43.92	37.67	34.06	37.84	38.67
TIES	13.68	11.09	33.54	42.59	37.00	34.06	39.65	38.00
DARE	11.28	9.80	32.93	41.53	37.33	35.00	36.98	38.67
Fisher Merging	20.52	16.75	31.71	42.33	45.67	39.06	48.33	42.44
PCB	11.46	9.98	34.15	42.06	31.00	34.06	36.98	42.22
Localize & Stitch	18.30	14.97	33.54	42.86	26.33	35.62	34.85	41.56
Ours	27.50	22.00	34.75	43.92	46.33	43.44	53.54	44.67

`multilingual_val`, `instruction_val`, `coding_val`, and `safety_val`.[2] [3] [4] [5] [6] Since these datasets contain approximately 1000 examples each, we use the full available dataset for Fisher estimation. Fisher information for each task-specific model is computed using the corresponding task dataset after the sign-resolution step. All Fisher estimation experiments were performed on a single GPU.

For the GPT-2 experiments, we estimate Fisher information using the training split of the GLUE benchmark.[7] We use the training split to avoid leakage from the validation set, since the official test labels are not publicly available and evaluation is therefore performed on the validation set. Following prior work, we randomly sample 512 training examples for each task during Fisher estimation.

For the CLIP-based experiments, we estimate Fisher information using 256 randomly selected training examples from the corresponding task-specific datasets provided by Fusion-Bench.[8]

G Validation Details

We use a consistent validation pipeline across all merging methods to ensure fair comparison. All methods are evaluated using identical validation examples, inference settings, and evaluation scripts. For LLM-based experiments, evaluation is performed using vLLM with GPU memory utilization fixed to 0.9. We use `lm-evaluation-harness` (Biderman et al., 2024) for language model evaluation, BigCode Evaluation Harness (Ben Allal et al., 2022) for code generation evaluation, and `safety-eval-fork` for safety evaluation. Instruction-following perplexity is evaluated using a custom vLLM-based script.

For validation-time model selection, we compute the **PRR** relative to the corresponding specialist model for each task. For metrics where higher values are better, such as accuracy or BLEU, we compute

[2] `https://huggingface.co/datasets/MergeBench/math_val`
[3] `https://huggingface.co/datasets/MergeBench/multilingual_val`
[4] `https://huggingface.co/datasets/MergeBench/instruction_val`
[5] `https://huggingface.co/datasets/MergeBench/coding_val`
[6] `https://huggingface.co/datasets/MergeBench/safety_val`
[7] `https://huggingface.co/datasets/nyu-mll/glue`
[8] `https://huggingface.co/datasets/tanganke`

$\text{PRR}_t = \frac{M_t}{S_t}$, where M_t denotes the merged model performance on task t, and S_t denotes the performance of the corresponding specialist model. For safety evaluation, we report safety retention instead of attack success rate (ASR). Therefore, we compute $\text{PRR}_{\text{safety}} = \frac{1-\text{ASR}_{\text{merged}}}{1-\text{ASR}_{\text{specialist}}}$. For perplexity-based instruction evaluation, lower values are better. We use log-regret normalization and compute $\text{PRR}_{\text{instruction}} = 1 - (\log P_{\text{merged}} - \log P_{\text{specialist}})$, where P denotes corpus perplexity. For LLM-based validation, we evaluate each merged checkpoint on five validation domains corresponding to the five specialist models:

- **Instruction Validation:** We evaluate perplexity on `MergeBench/instruction_val` using the instruction-output pairs.
- **Mathematics Validation:** We evaluate using `mmlu_stem` through `lm-evaluation-harness`.
- **Coding Validation:** We evaluate using the CoNaLa benchmark through the BigCode Evaluation Harness.
- **Safety Validation:** We evaluate on `wildjailbreak:harmful` using `safety-eval-fork`.
- **Multilingual Validation:** We evaluate using `lambada_openai`.

For Fisher estimation in LLM-based experiments, we use 1000 examples from each MergeBench validation dataset: `instruction_val`, `math_val`, `coding_val`, `multilingual_val`, and `safety_val`.

We use the following validation hyperparameter grids:

- **Task Arithmetic:** scaling coefficient $\in \{0.1, 0.2, 0.3, 0.4, 0.5\}$
- **TIES:** sparsity $K \in \{0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9\}$ and scaling coefficient $\in \{0.1, 0.2, 0.3, 0.4, 0.5\}$
- **DARE:** sparsity $p \in \{0.5, 0.6, 0.7, 0.8, 0.9\}$ and scaling coefficient $\in \{0.1, 0.2, 0.3, 0.4, 0.5\}$
- **Localize and Stitch:** sparsity $\in \{0.01, 0.02, 0.05, 0.10, 0.20\}$
- **DRIFT-MEDIAN:** keep ratio $\in \{0.6, 0.8, 1.0\}$ and scaling factor $\in \{1.0, 1.1, 1.2, 1.3, 1.4, 1.5\}$
- **PCB:** ratio $\in \{0.05, 0.1, 0.2\}$

For GPT-2 based GLUE text classification experiments, we use the training split of each dataset to avoid validation leakage, since public test labels are unavailable and validation performance is used for reporting. We randomly sample 512 training examples per task for Fisher estimation and validation-time hyperparameter selection. The same sampled examples are reused across all merging methods.

For CLIP-based image classification experiments, we use 10% of the training split from each task-specific dataset for validation and 256 examples for Fisher estimation. The exact same subset of images is reused across all compared merging methods. For methods without hyperparameters, no validation-time search is performed.

H Evaluation Benchmarks

We evaluate LLM based checkpoints and merged models across multiple domains using a diverse suite of benchmarks. We evaluate 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), 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 (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.

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.

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 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 broad exploratory searches on *GPT-2* during the algorithm development phase, owing to its relatively small size and faster inference cycles. In this stage, we varied the Top-K parameter from Top-1 through Top-7, and additionally 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 1.0–1.5. Based on this observation, we subsequently performed a denser search within this region using 0.1 increments. The broader search over the full interval was used only during the initial development and exploratory phase of the algorithm and only using GPT2-based models. For the final validation and all reported experiments, we restricted the hyperparameter search to the refined range around the empirically identified high-performing region, using the denser grid described above. We perform GLUE based experiments on 2-4 RTX 2080 Ti GPU (based on availability), however we ensure that each process runs independently of other. For CLIP and LLM-based experiments, we use single A100 80GB GPU.

J Hyperparameter Sensitivity

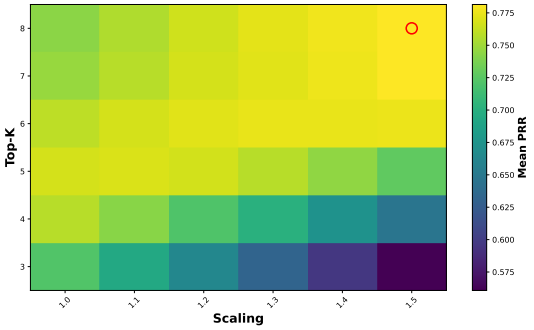


Figure 4: Sensitivity of the hyperparameters K and λ in `DRIFT-MEDIAN` on CLIP based tasks on validation set. The best performance is obtained on top-8 models with 1.5 as the scaling factor. The values near to the optimal hyperparameter have similar performance.

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 Runtime of Algorithms

We provide a rough runtime estimate of the considered algorithms based on experiments using Llama-3.2-3B. We separate validation-time hyperparameter search from the final test-time merge. Let t_m denote the merge runtime per hyperparameter configuration, t_e denote the evaluation runtime per hyperparameter configuration, and G denote the validation grid size. The total runtime is computed as

$$T_{\text{total}} = (t_m + t_e) \times G + t_m^{\text{test}},$$

where t_m^{test} is the merge runtime for the final selected configuration used for test evaluation.

For methods without validation-time hyperparameter search, we set $G = 0$, $t_m = 0$, and $t_e = 0$. Thus, their runtime only includes the final test-time merge, i.e.,

$$T_{\text{total}} = t_m^{\text{test}}.$$

For `DRIFT-MEDIAN`, Fisher statistics are computed in the first run and reused thereafter. Therefore, we separate the first configuration from the remaining cached configurations:

$$T_{\text{Drift}} = (t_m^{\text{first}} + t_e) + (t_m^{\text{cached}} + t_e) \times (G - 1) + t_m^{\text{test}}.$$

The estimated runtimes are shown in Table 15. While Fisher computation becomes more expensive for larger models such as Llama-3.1-8B, the validation evaluation cost also increases for all competing algorithms (including ours). Therefore, the relative overhead should not be interpreted solely from the merge-time computation.

That said, we do not claim that `DRIFT-MEDIAN` is runtime-equivalent. Lower validation data sizes can substantially reduce validation evaluation cost for competing methods, whereas `DRIFT-MEDIAN` incurs fixed Fisher computation overheads. Similarly, although we perform extensive hyperparameter search for fair comparison, significantly smaller search spaces could be sufficient in practice for the baselines and would considerably reduce the runtime of competing methods. Nevertheless, we believe that the runtime of `DRIFT-MEDIAN` remains sufficiently low for practical use cases, particularly when Fisher statistics are cached and reused across multiple runs. However, it should be noted that caching incurs an additional storage cost comparable to the size of the model. We also acknowledge that the implementations used in our experiments may not represent the most runtime-optimized versions of the respective methods. The runtime could potentially be reduced further by performing validation during the merging process and retaining the best checkpoint observed so far, thereby avoiding an additional test-time merging step and its associated model-size storage requirement. We did not adopt this strategy because it would introduce additional implementation complexity, while the test-time runtime remains sufficiently low for all methods considered.

L Scalability of `DRIFT-MEDIAN`

`DRIFT-MEDIAN` is in principle applicable to larger models because its main operations are coordinate-wise, and diagonal Fisher estimation requires only forward and backward passes over a small validation set with memory linear in the number of parameters. We did not extend the study to larger models mainly due to computational limits and the lack of controlled, publicly available fine-tuned checkpoints derived from the same base model.

Scalability, however, is not determined by model size alone. It also depends on task compatibility, parameter-space overlap, sparsity patterns, and the degree of directional support among task vectors. Our analysis in subsection 4.3 shows that highly specialized tasks with weak cross-task support are more likely to be attenuated during median-based aggregation, while mutually supported task updates are preserved more

Table 15: Estimated runtime for validation and test-time merging on Llama-3.2-3B. All runtime values are reported in minutes. Here, t_m denotes merge runtime per hyperparameter configuration, t_e denotes evaluation runtime per hyperparameter configuration, G denotes validation grid size, and T_{total} denotes the total estimated runtime.

Method	t_m	t_e	G	$T_{\textbf{total}}$
Simple Average	0	0	0	0.44
Task Arithmetic	0.56	17	5	88.36
TIES	1.80	17	40	753.80
DARE	3.35	17	25	512.10
PCB	15.00	17	3	111.00
Localize & Stitch	13.90	17	5	168.40
Drift Median, first run	95.98	17	1	∼546
Drift Median, cached runs	8.02	17	17	
Vanilla Fisher	0	0	0	92.54

Table 16: Model Performance on Different Domain Data for Fisher Estimation

Validation Data	SUN397	CARS	RESISC45	Eurosat	SVHN	GTSRB	MNIST	DTD	Average	**PRR**
Unchanged	64.89	64.86	73.16	80.41	87.10	68.08	97.45	56.54	74.06	81.82
MNIST → KMNIST	64.80	65.30	72.97	78.56	85.08	68.58	97.66	57.34	73.79	81.57
MNIST → KMNIST & SVHN → MNIST	64.77	65.12	73.43	81.70	79.56	68.37	97.25	56.65	73.35	81.10

effectively. This also explains why smaller models may exhibit stronger performance imbalance across diverse domains: their generic capabilities are weaker, and their task-specific updates may be less compatible.

This view is consistent with Yadav et al. (2025), who report that larger models are easier to merge and can support merging more expert models more effectively. At the same time, Wang et al. (2025) suggests that the effective parameter space can saturate as additional experts are added due to redundancy and Gaussian-width concavity. Therefore, a faithful scaling study would need to isolate model size, number of experts, task similarity, and support overlap, which we leave for future work.

M 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 16, we perform a more extreme modification by replacing SVHN (Street View House Numbers), which contains real-world 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 1 and Table 2, 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. In such cases, dataless merging approaches such as TIES would be appropriate.

N Component-wise Filtering Statistics

To understand whether inter-model Top-K affects all tasks uniformly, we analyze the fraction of task-vector coordinates removed after sign resolution with a stricter keep ratio of 0.6, corresponding to retaining three out of five task updates per coordinate. We find that sign resolution is the dominant conflict-removal step, dropping 39.68% of active coding updates, 34.91% of safety updates, 33.96% of instruction updates, 32.82% of multilingual updates, and 26.37% of math updates. In contrast, Top-K acts as a secondary magnitude-based filter. It removes 10.95% of sign-consistent coding updates and 10.19% of multilingual updates, but only 3.40% of instruction and 1.98% of math updates. This suggests that Top-K does not simply reduce all task updates equally; rather, it preserves the largest retained updates at each coordinate and filters weaker retained updates that may otherwise affect the median aggregation. Importantly, this behavior is coordinate-dependent: at many parameter indices, coding may have the largest retained magnitude and will therefore be preserved while updates from other tasks are discarded, whereas at other indices the opposite can happen. Thus, the observed task-level removal rates should be interpreted as aggregate statistics over coordinates, not as a global preference against any particular task. Overall, inter-model Top-K introduces coordinate-wise competition among task vectors, retaining whichever tasks provide the strongest sign-consistent updates at a given parameter index.

O Why Fisher Information for Sensitivity Weighting?

Fisher information is used in our framework because it provides non-negative, coordinate-wise sensitivity weights that are directly compatible with our aggregation objective. Our goal is not to claim that Fisher is the only valid importance metric, but to combine task-vector interference handling with a principled notion of parameter sensitivity.

Magnitude-based scores are a natural alternative and are widely used in model merging. However, magnitude mainly captures how far a parameter moves from the base model, not whether that parameter is functionally important for the task. Since our coordinate-wise Top-K step already uses update magnitude, using magnitude again for aggregation would conflate displacement size with sensitivity. We observe lower scores after applying these methods in Table 7.

Fisher information is also closely connected to Hessian-based curvature. Under standard regularity conditions for likelihood models,

$$\mathcal{I}(\theta) = \mathbb{E}\left[\nabla_\theta \log p(y \mid x;\theta) \nabla_\theta \log p(y \mid x;\theta)^\top\right] = \mathbb{E}\left[\nabla^2_\theta\big(-\log p(y \mid x;\theta)\big)\right].$$

Thus, Fisher can be interpreted as a practical curvature-aware sensitivity estimate. Direct Hessian estimation is expensive for large models, whereas the diagonal empirical Fisher is scalable, and non-negative.

Activation-based scores are another possible choice, but they require additional design decisions about layers, tokens, examples, and activation statistics. In contrast, Fisher is tied directly to the likelihood objective and has already been used for model merging (Matena & Raffel, 2022). Therefore, Fisher is a suitable sensitivity signal for DRIFT-MEDIAN because it is likelihood-based, curvature-related, coordinate-wise, and practical under a diagonal approximation.

P 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, sentence paraphrasing and minor code debugger/re-writer for automating the running of the merging

methods using bash scripts. The core research ideas, experimental design, results, and their interpretation were conceived and formulated entirely by the authors.