MODEL MERGING OVERVIEW

Xuankun Yang Shanghai Jiao Tong University
Shanghai, China
kk-dao@sjtu.edu.cn

ABSTRACT

Model merging has emerged as a technique that merges the parameters or predictions of multiple pre-trained models into a single one. It is a promising approach for unifying independently fine-tuned models into an integrated framework, significantly enhancing computational efficiency in multi-task learning. However, model merging on large-scale deep learning models (e.g., LLMs and foundation models) faces several challenges, including high computational cost, high-dimensional parameter space, interference between different heterogeneous models, etc. In this overview, we foucus on model merging methods on large models, involving fine-tune strategies like FFT(fully fine-tune) and PEFT(parameter efficient fine-tune). Specifically, we We divide existing model merging methods into four categories: (1)"Weighted average". which uses different methods to weight each model and averages them; (2)"Sparsification strategy" sparsely configure the weights of each model for subsequent processing; (3)"Subspace-based methods" perform subspace decomposition on model weights and resolve potential issues to achieve better performance; (4)"MoE-based methods", a novel model merging strategy which imitates the MoE architecture in the large models and provides a new idea for model merging. Our overview is helpful in deeply understaning the correlation between different model merging methods, which can enlighten the research in the field of model merging.

1 INTRODUCTION

In the rapidly evolving landscape of deep learning, pre-trained models have emerged as a central driving force in the field of machine learning, demonstrating remarkable performance across a diverse range of complex tasks. However, with the continuous growth in model scale and the increasing specialization of application scenarios, the number of fine-tuned models for specific tasks has also grown exponentially. This trend presents a series of practical challenges. For instance, deploying separate models for each particular task leads to high computational costs, enormous storage requirements, and inefficiency in multi-task processing. Traditional solutions, such as multi-task learning(Caruana, 1997), typically necessitate simultaneous access to raw training data for all tasks, which is often infeasible in real-world scenarios due to data privacy concerns or difficulties in data integration.

To address these challenges, model merging techniques have emerged and rapidly become a promising research direction in the field of machine learning. The core philosophy of model merging lies in its ability to effectively integrate the parameters or predictions of multiple independently fine-tuned models into a single, unified model, without the need for time-consuming additional training processes or access to raw training data. This approach not only significantly enhances the computational efficiency of multi-task learning but also demonstrates substantial potential in areas such as continual learning and few-shot learning, with its advantages becoming particularly pronounced when dealing with large language models (LLMs) and foundation models. Compared to traditional multi-task learning, model merging does not require simultaneous access to all task training data, and compared to model ensembling (Sagi & Rokach, 2018), the merged model is cheaper to run at inference time.

However, model merging is not without its inherent complexities. In high-dimensional parameter spaces, potential conflicts and interference among different models often lead to a decline in the performance of the merged model. Consequently, effectively balancing parameter competition among tasks and mitigating such interference has become a focal point of research in this domain.

This overview aims to delve into the various model merging methods that have emerged recently in the context of large models, covering fine-tuning strategies such as fully fine-tuning (FFT) and parameter-efficient fine-tuning (PEFT). Specially, we primarily focuses on model merging methods for **fully fine-tuned (FFT)** large models, while also discussing the applicability of **parameter-efficient fine-tuning (PEFT)** strategies. We categorize existing model merging methods into the following four main categories: (1) **Weighted Average Strategies**, which fuse knowledge by applying different weighting methods to each model's parameters, including Simple Average(Wortsman et al., 2022; Ilharco et al., 2022; Choshen et al., 2022), Fisher Average (Matena & Raffel, 2022), RegMean(Jin et al., 2025), Task Arithmetic (Ilharco et al., 2023), AdaMerging Yang et al. (2024b), etc. Notably, the "Model soups" method (Wortsman et al., 2022) falls into this category, demonstrating that averaging weights of multiple fine-tuned models can improve accuracy and robustness without increasing inference time. (2) **Sparsification Strategies**, designed to reduce task interference and optimize model structure by sparsely configuring the weights of each model. This includes DELLA-Merging (Deep et al., 2024) ,TIES-Merging(Yadav et al., 2023), DARE(Yu et al., 2024), Localize-and-Stitch (He et al., 2025), FREE Merging (Zheng & Wang, 2025), TALL masks and Consensus method(Wang et al., 2024). (3) **Subspace-based Methods**, which approach model merging from a deeper mathematical perspective by performing subspace decomposition on model weights to resolve potential issues and achieve better performance. Examples include TSV-Merging(Gargiulo et al., 2025), Isotropic Merging(Marczak et al., 2025), Knots(especially with PEFT)(Stoica et al., 2024), AdaRank (Lee et al., 2025a), which reduces the performance gap between merged models and fine-tuned models to nearly 1%; and STAR(Lee et al., 2025b). Furthermore, PCB Merging (Du et al., 2024) outperforms the strongest baseline for T5-base and T5-Large models by 1.9% and 2.1% respectively. Revisiting Weight Averaging for Model Merging(CART) also shows that the performance gap with traditional multi-task learning can be narrowed to within 1-3% (Choi et al., 2025). (4) **MoE-based Methods**, a novel class of model merging strategies inspired by the Mixture-of-Experts (MoE) architecture in large models, offering new insights for model merging. This category includes SMILE (Tang et al., 2024a), Twin-Merging (Lu et al., 2024), which demonstrates an average improvement of 28.34% in absolute normalized score for discriminative tasks, and WEMoE (Tang et al., 2024b).

By systematically organizing and analyzing these diverse methodologies, we aim to gain a deeper understanding of the intrinsic connections and distinctions among different model merging approaches, uncover their underlying theoretical foundations, and explore how they effectively address the challenges faced by large models in multi-task applications. We believe this overview will provide researchers in the field of model merging with a comprehensive and insightful framework, inspiring more in-depth future explorations in this direction.

2 PRELIMINARY

Model merging integrates multiple deep neural network models, each individually trained (i.e. fine-tuned) on distinct tasks starting from the same pre-trained model, into a single merged model. Let $\theta_0 \in \mathbb{R}^d$ denote the weights of the pre-trained network, and $\theta_t \in \mathbb{R}^d$ denote the fine-tuned weights for task t, with $t = 1, \ldots, T$, where d is the dimension of the parameter space and T is the total number of tasks. We will use the notation $\theta_t^{(l)}$ to identify the weights of layer l for task t and L to denote the total number of layers in a network. The objective of model merging is to find a merging function f, such that the merged model:

$$\theta_M^{(l)} = f\left(\theta_0^{(l)}, \{\theta_t^{(l)}\}\right), \quad \forall l = 1, \ldots, L \tag{1}$$

is able to perform all tasks on which the individual models θ_t are trained.

Following `Task Arithmetic`(Ilharco et al., 2023) which defines task vectors capturing the differences in model weights for individual tasks, we denote the task vectors for task t as

$$\tau_t = \theta_t - \theta_0. \tag{2}$$

Similarly, we define the layer-wise `task matrix`(Marczak et al., 2025; Gargiulo et al., 2025) $\Delta_t^{(t)}$ as the difference between the weights of model θ_t and the pre-trained model θ_0 for layer l:

$$\Delta_t^{(l)} = \theta_t^{(l)} - \theta_0^{(l)} \tag{3}$$

In the rest of the article, the l superscript is omitted when not relevant to the discussion, and all definitions refer to an arbitrary layer.

3 WEIGHTED AVERAGE METHODS

Weighted average strategies form a foundational category in model merging, aiming to combine the knowledge embedded in multiple independently fine-tuned models by linearly interpolating their parameters. Given a pre-trained model θ_0 and T task-specific fine-tuned models θ_t for $t = 1, \ldots, T$, the general objective is to find a merged model θ_M such that it retains the capabilities of all individual tasks. This is typically achieved by calculating a weighted sum of the fine-tuned parameters, or more commonly, their task vectors (defined as $\tau_t = \theta_t - \theta_0$). The core challenge within this family of methods lies in determining the optimal set of weights or coefficients to apply to each model or task vector, ensuring minimal performance degradation and effective knowledge integration.

3.1 SIMPLE AVERAGE

Simple Average(Wortsman et al., 2022) is the most straightforward approach, often referred to as uniform averaging. In this method, the parameters of the fine-tuned models are directly averaged. If we consider the parameters of T fine-tuned models as $\theta_1, \ldots, \theta_T$, the merged model θ_M is simply given by

$$\theta_M = \frac{1}{T} \sum_{t=1}^{T} \theta_t. \tag{4}$$

This method implicitly assigns an equal weight of $1/T$ to each model, which is equal to

$$\theta_M = \theta_0 + \frac{1}{T} \sum_{i=1}^{T} \tau_i, \tag{5}$$

where $\tau_i = \theta_i - \theta_0$, which is a special case of Task Arithmetic(Ilharco et al., 2023).

While conceptually simple and computationally inexpensive, it assumes that all fine-tuned models contribute equally and are aligned in parameter space, which is often not the case, leading to potential performance drops due to task interference. The "Model soups" method (Wortsman et al., 2022) explores a refined version of weight averaging, demonstrating that averaging weights of multiple models fine-tuned with different hyperparameter configurations can significantly improve accuracy and robustness without incurring additional inference costs. For example, the ViT-G model achieved 90.94% top-1 accuracy on ImageNet using this approach.

3.2 FISHER AVERAGE

Fisher Average (Matena & Raffel, 2022), or Fisher-Weighted Averaging, addresses some limitations of simple averaging by incorporating information about the "importance" of each parameter. This method is rooted in Bayesian principles, i.e.

$$\theta_M = \arg\max_{\theta} \sum_{i=1}^{T} \lambda_i \log p\left(\theta \mid \theta_i, F_i\right), \tag{6}$$

where each model's posterior distribution over its parameters is approximated as a Gaussian distribution and the precision matrix of this Gaussian corresponds to the Fisher Information matrix which is a $\mid \theta \mid \times \mid \theta \mid$ positive semidefinite matrix given by the formula

$$F_\theta = \mathbb{E}_x \left[\mathop{\mathbb{E}}_{y \sim p_\theta(y|x)} \nabla_\theta \log p_\theta(y \mid x) \nabla_\theta \log p_\theta(y \mid x)^T \right]. \tag{7}$$

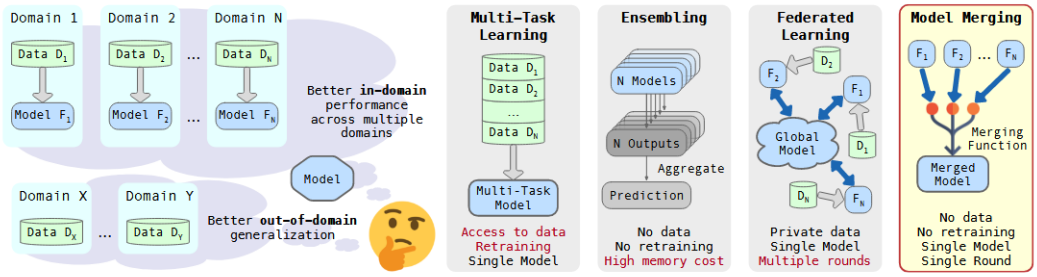

Figure 1: RegMean

Moreover, (Matena & Raffel, 2022) estimated the diagonal of the Fisher matrix via

$$\hat{F}_\theta = \frac{1}{N}\sum_{i=1}^{N} \mathop{\mathbb{E}}_{y\sim p_\theta(y|x)} \left(\nabla_\theta \log p_\theta(y \mid x_i)\right)^2, \tag{8}$$

then Eq. 6 has the close-form solution

$$\theta_M^{(j)} = \frac{\sum_{i=1}^{T} \lambda_i F_i^{(j)} \theta_i^{(j)}}{\sum_{i=1}^{T} \lambda_i F_i^{(j)}}, \tag{9}$$

where $j = 1, \ldots, \mid \theta \mid$.

Intuitively, we can think of Fisher merging as computing a weighted average of the parameter values in each model where the weighting is done according to each parameter's Fisher information. This procedure essentially performs a non-uniform average that gives more emphasis to parameters that are well-determined by the training data. The merging process is formulated to approximately maximize the joint likelihood of the posteriors of the models' parameters.

3.3 RegMean

RegMean (Regularized Mean)(Jin et al., 2025) can be considered a refinement of simple averaging. It starts by inferring the optimal solution of merging two linear regression models trained on different data distributions and analyze its relationship to Simple Average.

Consider two linear models $f_1(x) = W_1^\top x$ and $f_2(x) = W_2^\top x$, where $x \in \mathbb{R}^m$ and $W_1, W_2 \in \mathbb{R}^{m\times n}$, that are trained on two different annotated datasets $\langle X_1, y_1\rangle, \langle X_2, y_2\rangle$, where $X_1 \in \mathbb{R}^{N_1\times m}$ and $X_2 \in \mathbb{R}^{N_2\times m}$. Each row in X_i corresponds to a training example. Its goal is to obtain a single merged model $f_M(x) = W_M^\top x$ with outputs similar to f_1 on X_1 and f_2 on X_2. With ℓ^2 distance as the metic, the optimization problem can be formulated as

$$\min_W \|W^\top X_1 - W_1^\top X_1\|^2 + \|W^\top X_2 - W_2^\top X_2\|^2. \tag{10}$$

And Eq. 10 has a close-form solution $W_M = \left(X_1^\top X_1 + X_2^\top X_2\right)^{-1}\left(X_1^\top X_1 W_1 + X_2^\top X_2 W_2\right)$. Then we extend this to T tasks, and we have

$$W_M = \left(\sum_{i=1}^{T} X_i^\top X_i\right)^{-1} \sum_{i=1}^{T} \left(X_i^\top X_i W_i\right). \tag{11}$$

(Jin et al., 2025) empirically found that directly applying Eq. 11 for merging yields degenerated models in case of some pre-trained LM architectures. They therefore decrease the non-diagonal items of the inner product matrices by multiplying them with a scalar α (set as 0.9 most of the times). This also corresponds to adding a regularization term in the optimization objective in Eq. 10 that penalizes the Euclidean distance between the merged weights W_M and individual model weights $W_{1,\ldots,T}$. We include a formal derivation and proof in Appendix A and the schematic diagram is in Fig. 1.

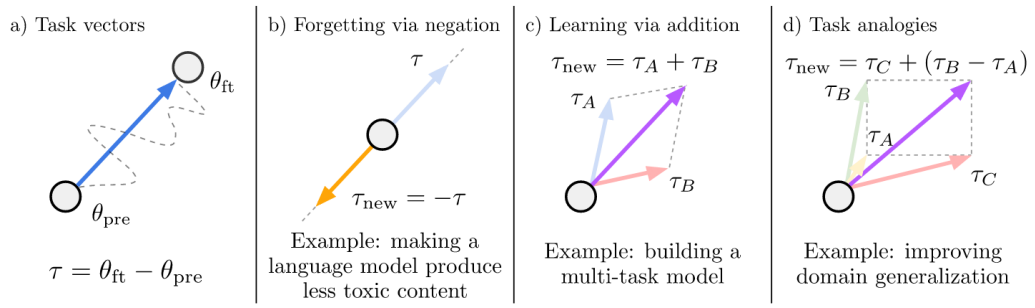

Figure 2: Task Arithmetic

Generally, regularized mean methods aim to improve upon basic averaging by introducing a regularization term that penalizes deviations from a central point or encourages certain properties in the merged model. This regularization could be based on various criteria, such as minimizing the variance among parameters or promoting sparsity. The goal is to find a set of weights or a merged parameter set that is a compromise between the individual models, guided by some notion of stability or generalizability beyond a simple arithmetic mean.

3.4 Task Arithmetic

Task Arithmetic (Ilharco et al., 2023) provides a powerful framework for combining model capabilities by operating on task vectors $\tau_t = \theta_t - \theta_0$. This approach posits that the knowledge learned for a specific task can be represented as a direction in the model's weight space. Different arithmetic operations, such as addition and negation, can then be performed on these task vectors to achieve desired changes in the model's behavior. For instance, adding multiple task vectors, scaled by certain coefficients, creates a new model capable of performing all associated tasks. The merged model can be represented as

$$\theta_M = \theta_0 + \lambda \sum_{t=1}^{T} \tau_t, \tag{12}$$

where λ is the scaling coefficient and is determined using held-out validation sets. The fundamental idea is that movement in the direction of a task vector improves performance on that task, and these vectors can be combined to steer the model towards multi-task proficiency (Ilharco et al., 2023). Task vectors has many more functions in editing models, please refer to Fig. 2

3.5 AdaMerging

AdaMerging (Yang et al., 2024b) extends the concept of task arithmetic by adaptively learning the merging coefficients for different tasks or layers (shown in Fig. 3), corresponding to **Task-wise AdaMerging**:

$$\theta_M = \theta_0 + \sum_{i=1}^{T} \lambda_i \tau_i \tag{13}$$

and **Layer-wise AdaMerging**:

$$\theta_M^{(l)} = \theta_0^{(l)} + \sum_{i=1}^{T} \lambda_i^{(l)} \Phi(\tau_i^{(l)}). \tag{14}$$

Naturally, it can also adaptively merge the task vector $\Phi(\tau)$ after removing parameter redundant values and sign conflicts in Ties-Merging(Yadav et al., 2023), which is named **AdaMerging++**.

Unlike traditional task arithmetic, which might use predefined or manually tuned coefficients, AdaMerging aims to autonomously learn these weights without relying on the original training data.

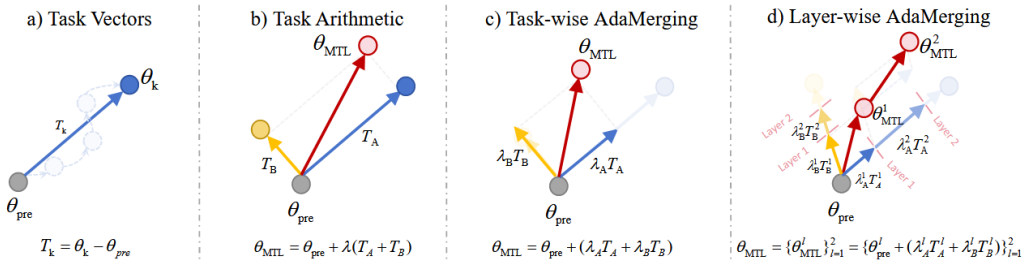

Figure 3: AdaMerging

It leverages an unsupervised objective, i.e. entropy minimization on unlabeled test samples, to iteratively refine the merging coefficients, which is denoted as follows:

$$\min_{\lambda_1,\lambda_2,\ldots,\lambda_T} \sum_{i=1}^{T} \sum_{x\in\mathcal{B}_i} H(f_{\theta_M}(x)), \tag{15}$$

where $\mathcal{B}_i$ represents a batch of unlabeled test samples sampled in task i.

This adaptive learning mechanism helps in mitigating potential conflicts and complex correlations among multiple tasks which often cause performance deterioration in direct addition methods. AdaMerging has shown remarkable performance improvements, such as an 11% gain over state-of-the-art task arithmetic merging schemes (Yang et al., 2024b).

3.6 Uncertainty-Based Gradient Matching

Uncertainty-Based Gradient Matching (Daheim et al., 2024) improves upon existing merging schemes by connecting the inaccuracy of weighted averaging to mismatches in the gradients between individual models and a conceptual "target" model, then reducing this mismatch using uncertainty estimates. This method is rooted in Bayesian principles and derives a merging scheme that approximates the maximum a posteriori (MAP) estimate of the joint posterior over all datasets, i.e.

$$\hat{\theta}_{1:T} = \arg\max_{\theta} q_\alpha(\theta \mid D_{1:T}), \tag{16}$$

where $q_\alpha(\theta \mid D_{1:T})$ is a Gaussian approximation to the weighted posterior $p_\alpha(\theta \mid D_{1:T}) \propto p(\theta) \prod_{t=1}^{T} e^{-\alpha_t \bar{\ell}_t(\theta)}$ obtained via Laplace's method. Here, $p(\theta) = \mathcal{N}(\theta \mid \theta_{\text{LLM}}, H_0^{-1})$ is a Gaussian prior centered at the pretrained model θ_{LLM}, and the likelihood terms are approximated using second-order Taylor expansions around each fine-tuned model θ_t:

$$\bar{\ell}_t(\theta) \approx \bar{\ell}_t(\theta_t) + \frac{1}{2}\|\theta - \theta_t\|^2_{H_t}, \tag{17}$$

with $H_t = \nabla^2 \bar{\ell}_t(\theta_t)$ being the Hessian matrix of the loss $\bar{\ell}_t$ at θ_t. In practice, (Daheim et al., 2024) approximate the Hessians using the diagonal of the Fisher information matrix, estimated as

$$\hat{F}_t = \frac{1}{N} \sum_{i=1}^{N} \mathop{\mathbb{E}}_{y\sim p_{\theta_t}(y|x_i)} \left(\nabla_{\theta_t} \log p_{\theta_t}(y \mid x_i)\right)^2. \tag{18}$$

Eq. 16 then has the closed-form solution

$$\hat{\theta}_M = \theta_{\text{LLM}} + \sum_{i=1}^{T} \alpha_i \left(\bar{H}^{-1} H_{0+i}\right)(\theta_i - \theta_0), \tag{19}$$

where $\bar{H} = H_0 + \sum_{i=1}^{T} \alpha_i H_i$ and $H_{0+i} = H_0 + H_i$, and the detailed derivation is shown in Appendix B. Intuitively, this method can be seen as preconditioning task arithmetic with Hessian-based scaling matrices that account for parameter uncertainty, reducing gradient mismatches and emphasizing parameters with lower uncertainty (higher Hessian values). The merging process approximates the MAP of a Laplace-approximated posterior, providing a principled way to fuse models while revealing implicit assumptions in prior methods like Task Arithmetic and Fisher Averaging.

3.7 NAN

NAN (Si et al., 2025) offers a training-free solution for estimating model merging coefficients. Similar to RegMean, it starts from a least square problem, and the close-form solution is $W_M = \left(\sum_{i=1}^{T} X_i^\top X_i\right)^{-1} \sum_{i=1}^{T} \left(X_i^\top X_i W_i\right)$. Considering the case where the input features are normalized, $X_i^\top X_i$ becomes approximately proportional to the sample size n_i. And we can assume the features are approximately isotropic: $X_i^\top X_i \approx n_i \mathbf{I}_d$, where $\mathbf{I}_d$ is the d-dimensional identity matrix. Recent findings suggest that the variance of the learned weights is inversely correlated with the training data volume, i.e. $n \propto \frac{1}{\mathrm{Var}(\mathbf{W})}$. Intuitively, models trained on larger datasets exhibit lower variance in parameter updates, as the optimization process averages out stochastic fluctuations over more samples. Assuming zero-mean updates, we have $\mathrm{Var}(\mathbf{W}) \propto \|\mathbf{W}\|_F^2$, where the Frobenius norm serves as a direct measure of magnitude. Finally, we have the coefficient for the weight $\theta_i^{(l)}$ in task i at layer l as:

$$\alpha_i = \frac{1/\|\mathbf{W}_i\|_F}{\sum_{j=1}^{T} 1/\|\mathbf{W}_j\|_F}. \tag{20}$$

When merging a large number of models, the softmax-normalized coefficients can become excessively small. To mitigate this issue, we apply a global scaling factor $T/2$ to the merged weights. Then we get the merged model:

$$\theta_M^{(l)} = \theta_0 + \frac{T}{2} \sum_{i=1}^{T} \alpha_i \tau_i. \tag{21}$$

As a training-free and plug-and-play method, NAN is highly scalable and broadly applicable across various merging strategies, consistently improving the performance of baseline methods. This makes it particularly attractive for practical applications where retraining or extensive hyperparameter tuning is not feasible.

4 SPARSIFICATION METHODS

Sparsification strategies in model merging focus on selectively activating or retaining only the most critical parameters or components within the merged model, aiming to reduce interference between tasks, improve efficiency, and enhance performance. These methods recognize that not all parameters contribute equally to the performance of a task, and that keeping all parameters from all individual models can lead to conflicts and redundancy. By introducing sparsity, these techniques seek to distill the essential knowledge from each fine-tuned model while minimizing detrimental interactions.

4.1 TIES-MERGING

TIES-Merging (TRIM, ELECT SIGN & MERGE)(Yadav et al., 2023) addresses interference in model merging by explicitly handling redundancy and sign conflicts in task-specific parameter updates. It builds on the concept of task vectors, introduced in prior work like Task Arithmetic(Ilharco et al., 2023), to mitigate performance degradation when combining multiple fine-tuned models.

Consider a pre-trained model with parameters θ_0, fine-tuned on n tasks to yield task-specific models $\{\theta_t\}_{t=1}^{T}$ and the task vector for each task t: $\tau_t = \theta_t - \theta_0 \in \mathbb{R}^d$, capturing the parameter updates during fine-tuning. Each τ_t can be decomposed into a sign vector $\gamma_t = \mathrm{sgn}(\tau_t) \in \{-1, 0, +1\}^d$ and a magnitude vector $\mu_t = |\tau_t| \in \mathbb{R}^d_{\geqslant 0}$, such that

$$\tau_t = \gamma_t \odot \mu_t, \tag{22}$$

where $\odot$ denotes element-wise multiplication.

The goal is to merge these task vectors into a single merged task vector τ_M, producing a multitask model $\theta_M = \theta_0 + \lambda \tau_M$, where λ is a scaling hyperparameter. TIES-Merging identifies two primary sources of interference: (1) redundant parameters with small magnitudes that obscure influential updates, and (2) sign conflicts where parameters across models point in opposing directions, leading to destructive averaging.

To resolve these, TIES-Merging proceeds in three steps:

(1) **Trim**: For each task vector τ_t, retain only the top-$k\%$ parameters by magnitude (e.g., $k = 20$), resetting the rest to zero to eliminate redundant updates: $\hat{\tau}_t = \text{keep\_topk\_reset\_rest\_to\_zero}(\tau_t, k)$. This yields trimmed vectors $\hat{\tau}_t = \hat{\gamma}_t \odot \hat{\mu}_t$.

(2) **Elect Sign**: Resolve sign conflicts by electing an aggregate sign vector

$$\gamma_m = \text{sgn}\left(\sum_{t=1}^{n} \hat{\tau}_t\right), \tag{23}$$

which selects the direction with the highest total magnitude across models for each parameter.

(3) **Disjoint Merge**: For each parameter $p \in \{1, \ldots, d\}$, average only the values from models aligned with the elected sign: Let $A_p = \{t \in [n] \mid \hat{\gamma}_t^p = \gamma_m^p\}$, then

$$\tau_m^p = \frac{1}{|A_p|} \sum_{t \in A_p} \hat{\tau}_t^p, \tag{24}$$

where zeros are ignored.

(Yadav et al., 2023) empirically demonstrate that this process preserves influential updates and avoids shrinkage in merged parameter magnitudes. Ablations show all steps are crucial, with sign resolution being particularly vital, as flipping signs of high-magnitude parameters causes catastrophic drops.

Generally, interference-resolving merging methods aim to enhance multitask performance by disentangling conflicting parameter updates, often through pruning redundancies and aligning directions. This promotes robustness in merged models, enabling better in-domain and out-of-domain generalization without additional training, and bridges the gap toward multitask-trained baselines.

4.2 EMR-MERGING

EMR-Merging (ELECT, MASK & RESCALE-MERGING)(Huang et al.) introduces a tuning-free paradigm for model merging that mitigates performance gaps by combining a unified model with lightweight task-specific modulators, rather than forcing a single merged model to handle all tasks. It extends task vector-based approaches like Task Arithmetic(Ilharco et al., 2023) and Ties-Merging(Yadav et al., 2023), addressing limitations where a single set of weights inadequately approximates diverse fine-tuned models, especially with increasing task counts or complexity.

Given a pre-trained model with parameters θ_0, fine-tuned on T tasks to produce task-specific models $\{\theta_i\}_{i=1}^T$, the task vector for task i is $\tau_i = \theta_i - \theta_0 \in \mathbb{R}^d$. EMR-Merging aims to derive a unified task vector τ_{uni} shared across tasks, augmented by task-specific masks and rescalers that align direction and magnitude without additional training or data. This yields a multitask model via $\theta_M = \theta_0 + \hat{\tau}_t$, where $\hat{\tau}_t$ is the modulated unified vector for task t.

The method proceeds in three steps:

(1) **Elect Unified Task Vector**: Compute an aggregate sign vector

$$\gamma_{\text{uni}} = \text{sgn}\left(\sum_{i=1}^{T} \tau_i\right), \tag{25}$$

selecting the dominant direction per parameter. Then, form the magnitude vector ϵ_{uni} by taking the maximum absolute value across task vectors that align with γ_{uni} for each parameter. The unified vector is

$$\tau_{\text{uni}} = \gamma_{\text{uni}} \odot \epsilon_{\text{uni}}, \tag{26}$$

preserving maximal amplitude while minimizing interference.

(2) **Task-Specific Masks**: For each task i, generate a binary mask

$$M_i = (\tau_i \odot \tau_{\text{uni}} > 0), \tag{27}$$

which zeros out parameters with conflicting signs to align directions. Masks are 1-bit per parameter, making them extremely lightweight (e.g., $\sim$1/32 the size of a float32 model).

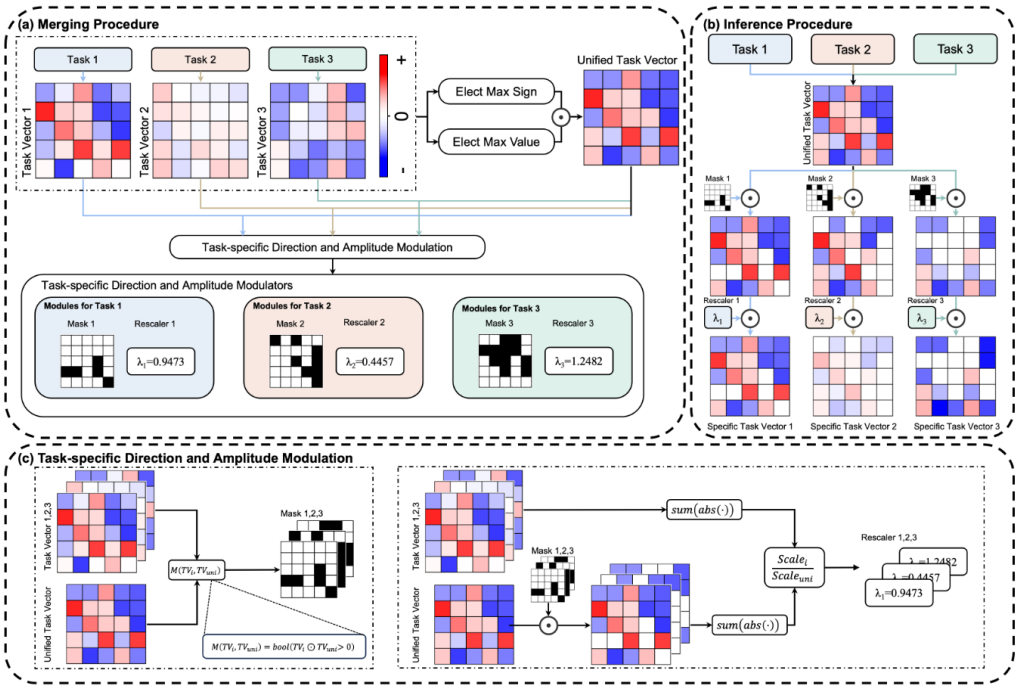

Figure 4: EMR-Merging

(3) **Task-Specific Rescalers**: Compute a scalar

$$\lambda_i = \frac{\sum |\tau_i|}{\sum |M_i \odot \tau_{\text{uni}}|}, \tag{28}$$

which aims to match the average magnitude of the masked unified vector to the original task vector. During inference for task t, modulate as

$$\hat{\tau}_t = \lambda_t \cdot M_t \odot \tau_{\text{uni}}. \tag{29}$$

(Huang et al.) theoretically show that masks and rescalers minimize the ℓ^2 distance between the modulated unified model and individual models (we provide detailed proofs in Appendix G). The framework overview is in Fig. 4.

The method is tuning-free, requiring no validation data (fixed hyperparameters suffice), and modulators add negligible storage ($\sim$3% overhead for 8 tasks). This facilitates scalable multi-task deployment, particularly for large models where full retraining is prohibitive, and supports extensions to diverse modalities and architectures.

4.3 Localize-and-Stitch

Localize-and-Stitch (He et al., 2025) introduces an efficient model merging strategy that addresses task interference by selectively integrating sparse, task-specific regions from finetuned models into a pretrained base model. Unlike global merging methods that operate on all parameters, this approach operates in two key steps:

(1) **Localization**: Identify sparse regions (approximately 1% of total parameters) encapsulating essential task-specific skills.

(2) **Stitching**: Reintegrate these regions into the pretrained model to create a merged model.

The method is illustrated in Fig. 5. Formally, given a pretrained model with parameters $\theta_0 \in \mathbb{R}^d$ and T finetuned models $\{\theta_i\}_{i=1}^T$, each finetuned on a specific task, the task vector is defined as $\tau_i = \theta_i - \theta_0$. The localization step aims to find a binary mask $\gamma_i \in \{0,1\}^d$ for each task i that

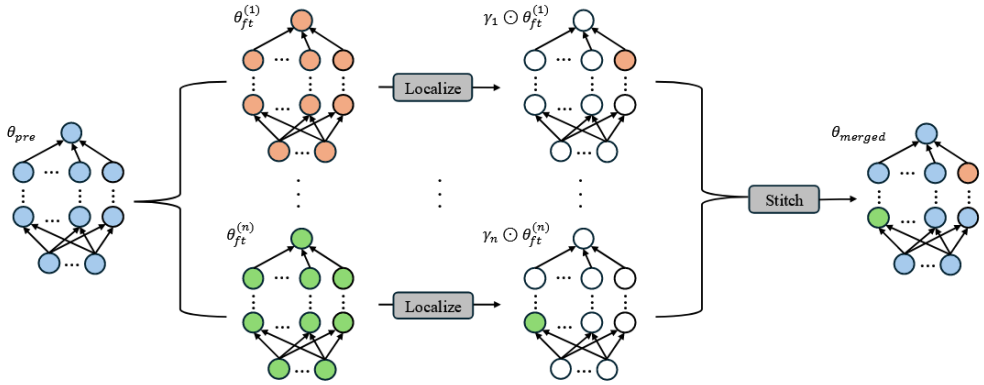

Figure 5: Localize-and-Stitch

identifies a sparse set of parameters (with sparsity level s, typically 1%) that retains most of the finetuned performance. This is formulated as an optimization problem:

$$\gamma_i = \underset{\gamma \in \{0,1\}^d : \|\gamma\|_0 \leqslant s}{\arg\min} \ell_i(\theta_0 + \gamma \odot \tau_i), \tag{30}$$

where ℓ_i is the loss function for the i-th task, and $\odot$ denotes the element-wise product. To make optimization tractable, the binary mask is reparametrized using a real-valued vector $S_i \in \mathbb{R}^d$, with $\gamma_i = \text{round}(\sigma(S_i))$, where σ is the sigmoid function. The optimization is relaxed with an L_1 penalty to control sparsity:

$$S_i = \arg\min_{S \in \mathbb{R}^d} \ell_i(\theta_0 + \sigma(S) \odot \tau_i) + \lambda \|\sigma(S)\|_1, \tag{31}$$

where λ is a regularization coefficient. This process requires minimal validation data (e.g., 8-shot) and is highly efficient, as demonstrated in (He et al., 2025). And if the validation data is not avaliable, (He et al., 2025) use strategies like (Yadav et al., 2023) as:

$$\gamma_i[|\ \tau_i\ | > \text{top-k}(|\ \tau_i\ |)] = 1, \quad \text{otherwise} \quad 0. \tag{32}$$

In the stitching step, the localized regions are integrated into the pretrained model. To handle overlaps where multiple tasks activate the same parameter positions, the masks are adjusted such that $\gamma_i'[k] = \gamma_i[k] / \sum_{j=1}^{T} \gamma_j[k]$ for each position k. The merged model is then constructed as:

$$\theta_M = \theta_0 + \sum_{i=1}^{T} (\gamma_i' \odot \tau_i). \tag{33}$$

This approach avoids tuning scaling factors, unlike methods such as Task Arithmetic (Ilharco et al., 2023), simplifying the merging process and reducing computational overhead. And it is highly interpretable, since overlaps in localized regions indicate shared task-specific skills.

4.4 DROP AND RESCALE (DARE)

DARE (Drop And REscale) (Yu et al., 2024) is a novel method designed to reduce redundancy in delta parameters of Supervised Fine-Tuned (SFT) language models and facilitate effective model merging without retraining. Delta parameters, which is the same as a task vector, is defined as $\tau_t = \theta_t - \theta_0 \in \mathbb{R}^d$, represent the difference between the parameters of an SFT model and its pretrained backbone for a task t. DARE leverages the observation that SFT delta parameters are highly redundant, typically exhibiting small value ranges (e.g., within 0.002), allowing significant sparsity without compromising performance.

The DARE method operates in two steps:

(1) **Drop**, where a fraction p of delta parameters is randomly set to zero.

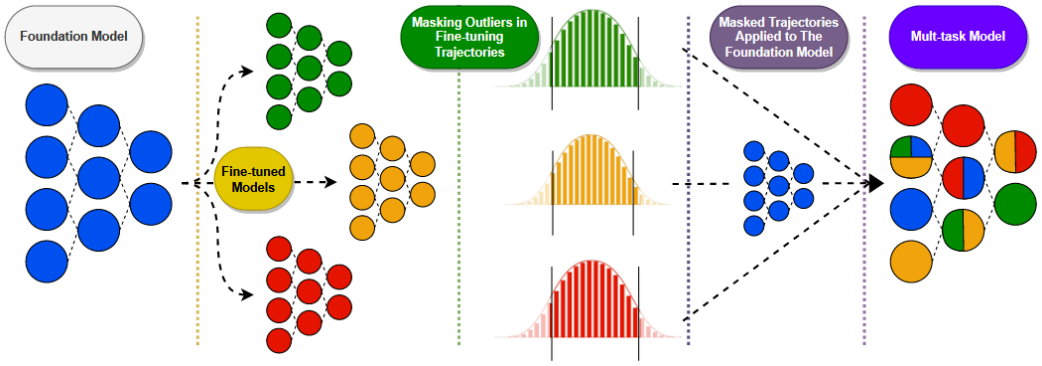

Figure 6: Model Breadcrumbs

(2) **Rescale**, where the remaining parameters are scaled by a factor of $1/(1-p)$ to preserve the expected embeddings.

Formally, for task vector τ_t, DARE is defined as:

$$m_t \sim \text{Bernoulli}(p), \tag{34}$$

$$\tilde{\tau}_t = (1 - m_t) \odot \tau_t, \tag{35}$$

$$\hat{\tau}_t = \tilde{\tau}_t / (1 - p), \tag{36}$$

where $\odot$ denotes the element-wise product, and the resulting parameters are $\hat{\theta}_t = \theta_0 + \hat{\tau}_t$. Theoretical analysis demonstrates that DARE approximates the original embeddings by ensuring the expected output of linear transformations remains unchanged when the scaling factor is set to $1/(1-p)$, as detailed in Appendix H.

For model merging, DARE serves as a plug-in to existing methods by first sparsifying the delta parameters of multiple SFT models to mitigate parameter interference. Given T SFT models $\{\theta_i\}_{i=1}^T$ finetuned from the same pretrained backbone, DARE is applied to each to obtain $\{\hat{\theta}_i\}_{i=1}^T$. These are then fused using established merging techniques, such as Task Arithmetic (Ilharco et al., 2023), where the merged model is computed as:

$$\theta_M = \theta_0 + \lambda \sum_{i=1}^{T} (\hat{\theta}_i - \theta_0) = \theta_0 + \lambda \sum_{i=1}^{T} \hat{\tau}_i \tag{37}$$

with λ as a scaling factor.

The method's efficiency, requiring only CPU-based operations, and its ability to preserve pretrained knowledge while enabling diverse task capabilities make it a versatile tool for model merging.

4.5 Model Breadcrumbs

Model Breadcrumbs (Davari & Belilovsky, 2024) presents a scalable approach for merging multiple fine-tuned models derived from the same foundation model to create a multi-task model without additional training or access to original data. By constructing sparse "breadcrumbs"—masked task vectors that filter out noise and interference—the method aggregates knowledge from auxiliary tasks while minimizing perturbations. This enables efficient model updates in a collaborative, open-source-like paradigm. The process is illustrated in Fig. 6.

Given a pretrained foundation model with parameters $\theta_0 \in \mathbb{R}^d$ and a fine-tuned model on task t with parameters $\theta_t \in \mathbb{R}^d$, the task vector $\tau_t = \theta_t - \theta_0$ captures the weight differences, including large outliers (significant deviations) and small perturbations (negligible changes). To mitigate interference, a layer-wise masking process is applied to sparsify τ_t.

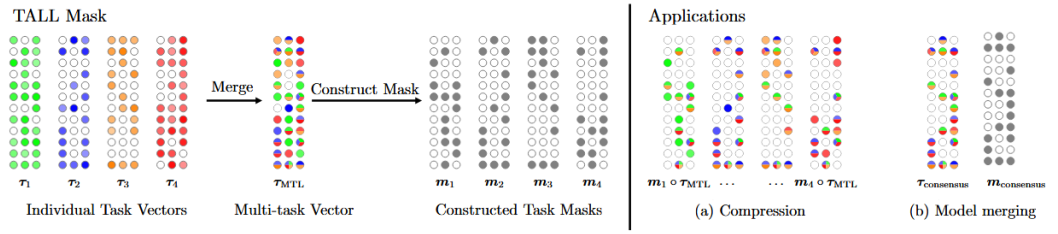

Figure 7: TALL-Mask and Consensus Merging

For each layer l, the weights are sorted by absolute magnitude, denoted as w_i^l where i is the sorted index from lowest to highest. The mask $m_l^{\beta,\gamma}$ is defined as:

$$m_l^{\beta,\gamma}[w_i^l] = \begin{cases} 0 & \text{if } i \leqslant \beta \text{ or } i \geqslant \gamma, \\ 1 & \text{otherwise,} \end{cases} \tag{38}$$

where β and γ are thresholds for the left (small values) and right (large values) tails of the magnitude distribution, respectively. Aggregating masks across all layers yields the final mask $m_t^{\beta,\gamma}$ for task t.

The masked task vector, or "breadcrumb," is obtained by element-wise multiplication: $m_t^{\beta,\gamma} \odot \tau_t$. For T tasks, the multi-task model is formed by averaging these masked vectors and adding them to the pretrained model, often with a scaling factor λ:

$$\theta_M = \theta_0 + \lambda \cdot \sum_{i=1}^{T} \left(m_i^{\beta,\gamma} \odot \tau_t \right). \tag{39}$$

This approach differs from methods like Task Arithmetic (Ilharco et al., 2023) by incorporating sparsification to reduce noise accumulation and from TIES-Merging (Yadav et al., 2023) by masking both small and large magnitudes layer-wise, rather than globally trimming low magnitudes.

Model Breadcrumbs is highly scalable, which means hyperparameters β and γ generalize across increasing numbers of tasks without per-task tuning. And it is efficient, requiring no validation sets or retraining. The method's ability to handle diverse tasks and modalities highlights its potential for community-driven model updates.

4.6 TALL-Masks and Consensus Merging

TALL-Masks and Consensus Merging (Wang et al., 2024) propose methods to localize task-specific information in merged models, addressing performance degradation in model merging due to task interference and irrelevant weights. TALL-masks identify sparse, task-specific supports in task vectors, enabling recovery of near-original performance from a multi-task vector. Consensus Merging refines existing merging techniques by eliminating selfish (task-exclusive) and catastrophic (globally irrelevant but harmful) weights, enhancing multi-task performance. The overall framework is illustrated in Fig. 7.

Formally, given a pretrained model $\theta_0 \in \mathbb{R}^d$ and fine-tuned models $\{\theta_i\}_{i=1}^T$ for T tasks, and the task vectors $\tau_t = \theta_t - \theta_0$ for task t. The multi-task vector is $\tau_{\text{MTL}} = \sum_{t=1}^T \tau_t$, and the merged model is $\theta_M = \theta_0 + \alpha\tau_{\text{MTL}}$, where $\alpha > 0$ is tuned on validation data.

For TALL-masks, a binary mask $m_t \in \{0,1\}^d$ is constructed to approximate τ_t from τ_{MTL} by minimizing the ℓ_1 distance:

$$\begin{aligned} \mathbf{m}_t^* &= \underset{\mathbf{m}_t \in \{0,1\}^d}{\arg\min} \|m_t \odot \tau_{\text{MTL}} - \tau_t\|_1 \\ &= \mathbb{1}\{|\tau_t| \geqslant |\tau_{\text{MTL}} - \tau_t|\}, \end{aligned} \tag{40}$$

where $\odot$ is the Hadamard product, and more detailed derivation are given in Appendix I.

Furthermore, (Wang et al., 2024) add a hyper-parameter λ_t to tune the amount of information for the mask to extract from multi-task vector; the smaller λ_t, the more parameters get selected by $\mathbf{m}_t$.

Finally, construct the task-specific masks based on:

$$\mathbf{m}_t = \mathbb{1}\{|\tau_t| \geqslant |\tau_{\text{MTL}} - \tau_t| \cdot \lambda_t\}, \tag{41}$$

with λ_t tuned per task (e.g., from $\{0.2, 0.3, 0.4, 0.5, 0.6\}$) to control sparsity. The approximated task vector is $\hat{\tau}_t = \mathbf{m}_t \odot \tau_{\text{MTL}}$, recovering $> 99\%$ of single-task accuracy.

Consensus Merging uses these masks to form a consensus mask for threshold $k \in \{0, \ldots, T\}$ as:

$$\mathbf{m}_{\text{consensus}} = \mathbb{1}\left\{\sum_{t\in[T]} \mathbf{m}_t \geqslant k\right\}. \tag{42}$$

By setting $k = 2$, we can retain weights relevant to at least two tasks (general weights) and discarding selfish ($m_t = 1$ for only one t) and catastrophic weights (relevant to none). The consensus task vector is

$$\tau_{\text{consensus}} = m_{\text{consensus}} \odot \tau_{\text{MTL}}, \tag{43}$$

and the merged model is $\theta_M = \theta_0 + \alpha\tau_{\text{consensus}}$.

This method boosts baselines like Task Arithmetic (Ilharco et al., 2023) by reducing interference, achieving up to 10% better normalized accuracy. And the mask intersections reveal task relationships, with universal weights forming 20% of parameters.

4.7 Concrete Subspace Learning

Concrete Subspace Learning (CONtinuous relaxation of disCRETE subspace learning)(Tang et al., 2023) addresses the challenge of task interference in multi-task model fusion by identifying a common low-dimensional subspace within the parameter space of task-specific models. It can be viewed as an enhancement to existing merging techniques, such as Task Arithmetic(Ilharco et al., 2023) and AdaMerging(Yadav et al., 2023), by leveraging shared information in this subspace to mitigate conflicts without significantly degrading performance.

Consider a pre-trained model $\theta_0 \in \mathbb{R}^d$ and a set of T fine-tuned models $\{\theta_i\}_{i=1}^T$ and the task vectors $\tau_i = \theta_i - \theta_0$ for task i. To resolve interference, Concrete Subspace Learning models the problem as a bi-level optimization to meta-learn a shared mask $m \in [0, 1]^d$ that identifies the subspace:

$$\max_m \max_w \frac{1}{T} \sum_{i=1}^{T} \text{Score}\left(\mathcal{A}(\theta_0, \{\mathcal{M}(\tau_j, m)\}_{j=1}^T, w)\right), \tag{44}$$

where $\mathcal{A}$ is a model fusion algorithm (e.g., Task Arithmetic or AdaMerging) with parameters w, $M : \mathbb{R}^d \times \mathbb{R}^d \to \mathbb{R}^d$ is the Concrete masking process, and Score is a task-specific metric. At the upper level, a shared Concrete mask $\mathbf{m}$ is learned to define the subspace; at the inner level, fusion parameters w are optimized to maximize merged model performance.

The mask m is sampled from a Concrete distribution (a differentiable relaxation of the Bernoulli distribution) parameterized by logits $x \in \mathbb{R}^d$. For a task vector τ_i, the masked version is $\tau_i' = \tau_i \odot \mathbf{m}$ (element-wise product). To preserve performance, the masked vector is rescaled:

$$\tau_i'' = \frac{\tau_i'}{\mathbb{E}_{m\sim\mathbf{m}}[m]} = \frac{\tau_i \odot \mathbf{m}}{\mathbb{E}_{m\sim\mathbf{m}}[m]}, \tag{45}$$

where the expectation is the mean of mask entries. The masked and rescaled task vectors $\{\tau_j''\}_{j=1}^T$ are then fused using $\mathcal{A}$ to obtain the merged parameters.

(Tang et al., 2023) employ a meta-learning framework with gradient-based optimization to solve Eq. 44, using unlabeled test data for test-time adaptation (e.g., via entropy loss minimization). This yields variants like **Concrete Task Arithmetic** (masking before arithmetic addition) and **Concrete AdaMerging** (masking before adaptive weighting). Empirically, they found that without rescaling, sparse masks degrade fine-tuned models, but the trick in Eq. 45 maintains task-specific information. A temperature parameter λ controls the relaxation's sharpness, with $\lambda \to 0$ approximating discrete masks. We include formal derivations of the Concrete distribution, Gumbel-Max trick, and bi-level optimization in Appendix J, and the schematic diagram is in Fig. 8.

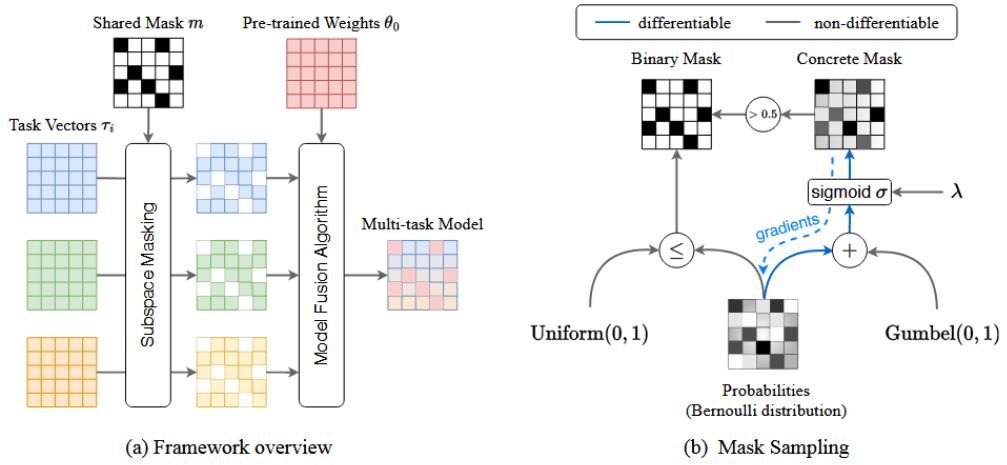

Figure 8: Concrete Subspace Learning

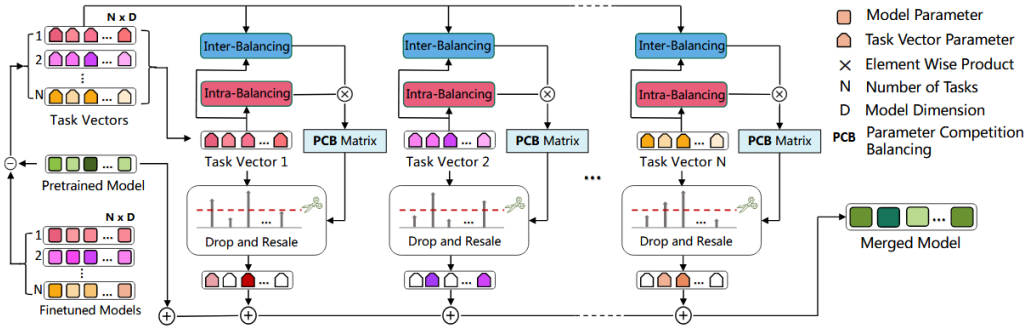

Figure 9: PCB-Merging

Generally, subspace learning methods for model merging aim to project task-specific parameters into a shared low-dimensional space, reducing interference by focusing on collective parameter impacts rather than individual attributes (e.g., magnitude or sign). This promotes a merged model that balances multi-task performance and generalization, often through meta-optimization or probabilistic masking, extending beyond naive averaging or interpolation.

4.8 Parameter Competition Balancing for Model Merging (PCB-Merging)

PCB-Merging (Parameter Competition Balancing for Model Merging)(Du et al., 2024) tackles parameter competition in model merging by balancing intra-task and inter-task conflicts through a lightweight, training-free approach. It refines task-vector-based merging by dropping low-importance parameters and rescaling the rest, enhancing multi-task performance over uniform averaging or simple task arithmetic. The schematic diagram is in Fig. 9.

Consider a pre-trained model with parameters $\theta_0 \in \mathbb{R}^d$ and T tasks and fine-tuned parameters $\{\theta_i\}_{i=1}^T$, yielding task vector $\tau_i = \theta_i - \theta_0$ for task i. PCB-Merging computes a parameter competition balancing (PCB) matrix $\beta_i \in \mathbb{R}^d$ for each τ_i via intra-balancing (self-awareness within tasks) and inter-balancing (cross-awareness across tasks). Intra-balancing emphasizes important parameters:

$$\beta_{\text{intra},i} = \text{Softmax}(T \cdot \text{Norm}(\tau_i \odot \tau_i)), \tag{46}$$

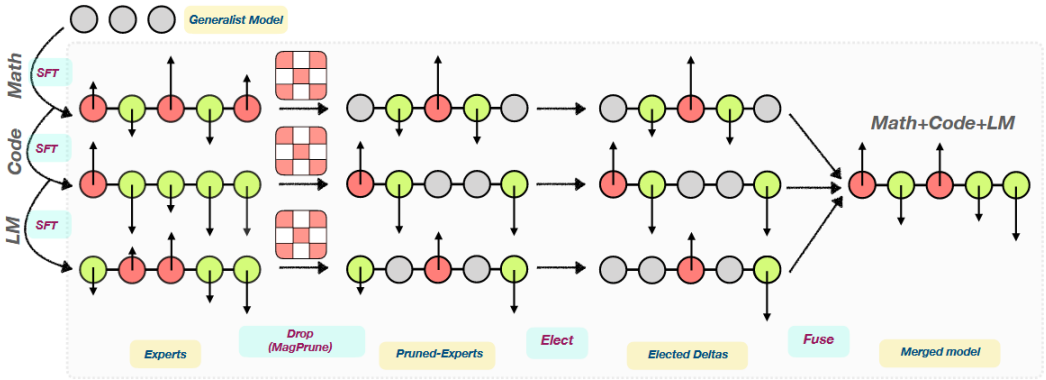

Figure 10: DELLA-Merging

where $\odot$ is element-wise product, Norm normalizes, and T regulates suppression as tasks increase. Inter-balancing assesses similarities:

$$\beta_{\text{inter},i} = \sum_{j=1}^{T} \text{Softmax}(\text{Norm}(\tau_i \odot \tau_j)). \tag{47}$$

The combined $\beta_i = \beta_{\text{intra},i} \odot \beta_{\text{inter},i}$. A mask $m_i \in \{0,1\}^d$ drops low-scoring parameters based on ratio r ($0 < r \leqslant 1$):

$$m_{i,d} = \begin{cases} 1 & \text{if } \beta_{i,d} \geqslant \text{sorted}(\beta_i)[(1-r) \times d] \\ 0 & \text{otherwise,} \end{cases} \tag{48}$$

yielding $\hat{\beta}_i = m_i \odot \beta_i$. The merged task vector is:

$$\tau_M = \frac{\sum_{i=1}^{n} (\hat{\beta}_i \odot \tau_i)}{\sum_{i=1}^{n} \hat{\beta}_i}, \tag{49}$$

and $\theta_M = \theta_0 + \lambda \tau_M$, where λ is a scaling hyperparameter.

(Du et al., 2024) optionally use evolutionary algorithms (e.g., CMA-ES) to search task-specific coefficients $\{\lambda_i\}_{i=1}^{n}$ for further gains, replacing uniform λ in Eq. 49. We include formal derivations of the balancing process, mask construction, and evolutionary search in Appendix K.

In short, parameter competition balancing methods improve merging by explicitly addressing conflicts at the parameter level, using importance scoring and similarity metrics to drop redundancies and rescale contributions. This fosters self-aware (intra-task) and cross-aware (inter-task) fusion, promoting robustness and efficiency beyond basic vector operations or uniform weighting.

4.9 Drop and RescaLe via Sampling with Magnitude (DELLA-Merging)

DELLA-Merging (Drop and rEscaLe via sampLing with mAgnitude)(Deep et al., 2024) mitigates interference in merging homologous models by introducing magnitude-based stochastic pruning of delta parameters, enhancing multi-task capabilities without additional training. It extends prior methods like Task Arithmetic(Ilharco et al., 2023), TIES-Merging(Yadav et al., 2023), and DARE(Yu et al., 2024) by incorporating a novel pruning technique, MAGPRUNE, which samples parameters inversely proportional to their magnitudes. The framework is shown in Fig. 10.

Given a pre-trained model $\theta_0 \in \mathbb{R}^d$ and T fine-tuned models $\{\theta_i\}_{i=1}^{T}$, each fine-tuned on a specific task and the delta parameters (task vector) $\tau_t = \theta_t - \theta_0$. DELLA-Merging comprises three steps: (1) Drop via MAGPRUNE to reduce interference;

(2) Elect to select directionally consistent deltas;

(3) Fuse to average and rescale.

In MAGPRUNE, for delta parameters $\{\tau_1, \ldots, \tau_T\}$ at a node (processed row-wise), rank by magnitude: $\{r_1, \ldots, r_T\} = \text{rank}(\{|\tau_1|, \ldots, |\tau_T|\})$, with $r_i \in \{0, \ldots, T-1\}$ (higher rank for larger magnitude). Assign drop probabilities:

$$\begin{aligned} \Delta_i &= \frac{\epsilon}{T} \cdot r_i, \\ p_{\min} &= p - \frac{\epsilon}{2}, \\ p_i &= p_{\min} + \Delta_i, \end{aligned} \tag{50}$$

where p is average drop rate, ϵ controls spread.

Sample masks $\mathbf{m}_i \sim \text{Bernoulli}(p_i)$, drop:

$$\tilde{\tau}_i = (\mathbf{1} - \mathbf{m}_i) \odot \tau_i, \tag{51}$$

and rescale

$$\hat{\tau}_i = \tilde{\tau}_i / (1 - p_i), \tag{52}$$

the analysis about rescale factor is the same as DARE(Yu et al., 2024), which is shown in Appendix H.

In Elect, for position k, compute sign sum

$$S = \text{sgn}(\sum_{t=1}^{T} \hat{\tau}_{t,k}), \tag{53}$$

and select

$$C = \{t \mid \text{sgn}(\hat{\tau}_{t,k}) = S\}, \tag{54}$$

In Fuse, average element-wisely:

$$\tau_k^{\text{avg}} = \frac{1}{|C|} \sum_{t \in C} \hat{\tau}_{t,k}, \tag{55}$$

and form

$$\theta_M = \theta_0 + \lambda \cdot \tau^{\text{avg}}, \tag{56}$$

where λ is a scaling factor.

(Deep et al., 2024) show that DELLA subsumes baselines:

(1) Uniform $p_i = p$ (i.e. $\epsilon = 0$) yields DARE(Yu et al., 2024)'s random drop;

(2) $p_i = 1$ for bottom $n - K$ and 0 for top K yields TIES(Yadav et al., 2023)'s top-K;

(3) $p = 0$ yields no drop (Task Arithmetic(Ilharco et al., 2023) variant).

DELLA aim to resolve task conflicts by selectively dropping and rescaling delta parameters, focusing on magnitude or direction to minimize interference. This enables efficient fusion of domain experts into a generalist model, improving upon uniform or random operations by preserving key task-specific changes.

5 SUBSPACE-BASED METHODS

Subspace-based methods for model merging critically leverage the intrinsic geometric structure of neural network parameter spaces. These techniques delve into the high-dimensional parameter space to identify and utilize specific subspaces where task-specific knowledge is effectively represented or where models can be better aligned. While foundational methods like **Isotropic Merging** (Marczak et al., 2025) simplify this by implicitly approximating each model's posterior distribution as an isotropic Gaussian, which leads to simple parameter averaging, more advanced approaches often explicitly employ matrix decomposition methods, such as Singular Value Decomposition (SVD) or spectral analysis. The core objective across these methods is to isolate, filter, or transform the most informative components of model parameters or task vectors. By operating within these identified (often lower-dimensional) subspaces, these methods aim to reduce interference between disparate task knowledge and achieve more robust and performant merged models.

5.1 TASK SINGULAR VECTORS MERGING (TSV-MERGING)

TSV-Merging (Task Singular Vectors Merging) (Gargiulo et al., 2025) introduces a structured approach to model merging by analyzing task vectors at the layer level through singular value decomposition (SVD), focusing on reducing task interference via low-rank approximations and decorrelation. This method leverages the inherent low-rank nature of per-layer task matrices to compress and merge models efficiently. Formally, for a set of task matrices $\{\Delta_i\}_{i=1}^T$ at a given layer, where $\Delta_i = \theta_i^{(l)} - \theta_0^{(l)}$ represents the weight difference for task i, the SVD is computed as $\Delta_i = U_i \Sigma_i V_i^\top$. The singular vectors U_i and V_i are termed Task Singular Vectors (TSV), and the low-rank approximation retains only the top-k components:

$$\hat{\Delta}_i = \sum_{j=1}^{k} \sigma_j^i u_j^i (v_j^i)^\top = \hat{U}_i \hat{\Sigma}_i \hat{V}_i^\top, \tag{57}$$

where σ_j^i are the singular values, u_j^i and v_j^i are the corresponding left and right singular vectors, and $\hat{U}_i$, $\hat{\Sigma}_i$, $\hat{V}_i^\top$ are the truncated SVD results. The correlation between the number of tasks T and the rank preserved k is interesting, for more information, please refer to Appendix C.2. For compression (TSV-Compress or TSV-C), when the task t is known, only the relevant low-rank $\hat{\Delta}_t$ is used, achieving up to 10× compression while preserving 99% accuracy. For merging without known tasks (TSV-Merge or TSV-M), interference is quantified via Singular Task Interference (STI):

$$\mathrm{STI}(\{\Delta_i\}_{i=1}^T) = \|(U^\top U - I)\Sigma(V^\top V - I)\|_1, \tag{58}$$

where $U = [U_1 \cdots U_T]$, $V = [V_1 \cdots V_T]^\top$, and Σ is the block-diagonal matrix of $\{\Sigma_i\}$.

To reduce STI, TSV-M applies whitening to decorrelate TSVs by solving an orthogonal Procrustes problem for orthogonal matrices $\hat{U}_\perp$ and $\hat{V}_\perp$:

$$\min_{\hat{U}_\perp} \|\hat{U}_\perp - \hat{U}\|_F \quad \text{s.t.} \quad \hat{U}_\perp^\top \hat{U}_\perp = I, \tag{59}$$

with closed-form solution via SVD $\hat{U} = PDQ^\top$ yielding $\hat{U}_\perp = PQ^\top$, which is similar to $\hat{V}_\perp$. And it is equvalant to the whitening process, more detailed derivation is in Appendix C.1.

The merged layer weights are then $\theta_M^{(l)} = \theta_0^{(l)} + \alpha \hat{M}$, where $\hat{M} = \hat{U}_\perp \hat{\Sigma} \hat{V}_\perp^\top$ and $\hat{\Sigma}$ contains the top-k singular values in each Σ_i. Intuitively, TSV-M compresses task matrices and decorrelates their singular vectors to minimize interference, outperforming prior methods by explicitly addressing geometric overlaps in the weight space without requiring validation data or labels.

5.2 ISOTROPIC MODEL MERGING (ISO-C AND ISO-CTS)

Isotropic Model Merging (Marczak et al., 2025) enhances model merging by analyzing task matrices through singular value decomposition (SVD) and ensuring balanced representation of task directions via isotropic scaling and subspace enhancement. This approach addresses the performance gap in merged models by focusing on subspace alignment between task-specific and merged matrices. In order to empirically quantify the overlap between subspaces, (Marczak et al., 2025) proposed the Subspace Alignment Ratio (SAR) metric and defined SAR between a task matrix Δ_i and a generic merged task matrix Δ_M as

$$\mathrm{SAR}(\Delta_i, \Delta_M; k_M) = \frac{\|\Pi_{k_M,M} \Delta_i\|_F}{\|\Delta_i\|_F}, \tag{60}$$

where $\Pi_{k_M,M} = U_{k_M,M} U_{k_M,M}^\top$ is the projection matrix onto the subspace spanned by the top k_M left-singular vectors of Δ_M. The columns of $U_{k_M,M}$ are obtained from the SVD decomposition of Δ_M, and the number of singular vectors used (k_M) is determined from the merged task matrix Δ_M by minimizing the approximation error with $\epsilon = 0.05$:

$$\begin{aligned} k_M &= \min\{k : \|\Delta_M - \Pi_{k,M}\Delta_M\|_F \leqslant \epsilon \|\Delta_M\|_F\} \\ &= \min\left\{k : \frac{\sum_{i=k+1}^r \sigma_i^2}{\sum_{i=1}^r \sigma_i^2} \leqslant \epsilon^2\right\} \end{aligned} \tag{61}$$

where $\Sigma = \text{diag}(\sigma_1, \sigma_2, \ldots, \sigma_r)$ contains the singular values of Δ_M, and the proof of equivalence is in Appendix D.1. SAR correlates strongly with merged model performance, motivating isotropic adjustments.

For Iso-C (Isotropic-Common), start with Task Arithmetic merged matrix $\Delta_{\text{TA}} = \sum_{t=1}^{T} \Delta_t$, compute its SVD $\Delta_{\text{TA}} = U\Sigma V^\top$, and replace Σ with an isotropic diagonal matrix $\tilde{\Sigma}$ where all entries are $\overline{\sigma} = \frac{1}{r}\sum_{i=1}^{r} \sigma_i$ (with $r = \min(m, n)$ the matrix rank):

$$\Delta_{\text{Iso-C}} = U\tilde{\Sigma}V^\top = \overline{\sigma}UV^\top. \tag{62}$$

For Iso-CTS (Isotropic-Common with Task-Specific), enhance the common subspace by iteratively replacing the least significant singular vectors of $\Delta_{\text{Iso-C}}$ with task-specific directions from individual Δ_t, prioritizing underrepresented tasks based on SAR. Firstly, retain the top-k singular vectors associated with the common subspace(i.e. the subspace identified by Δ_{TA}) $U^{1:k}$ and $V^{1:k}$. Next, project each task-specific matrix Δ_i onto the subspace *orthogonal* to the common subspace:

$$\overline{\Delta}_t = \Delta_t - \hat{U}\hat{U}^\top\Delta_t. \tag{63}$$

Then compute the SVD of $\overline{\Delta} = \overline{U}_t\overline{\Sigma}_t\overline{V}_t^\top$ and retain the top $s = \frac{r-k}{T}$ directions for each task t: $\overline{U}^{1:k}$, $\overline{\Sigma}^{1:k}$ and $\overline{V}^{1:k}$. The orthogonal projection Eq. 63 guarantees that both the left- and right-singular vectors of $\overline{\Delta}_t$, representing task-specific directions, are orthogonal to the subspace spanned by the common directions (given by $U^{1:k}$).

Combine common and task-specific matrices: $U_* = [U^{1:k} \mid \overline{U}_1^{1:s} \mid \cdots \mid \overline{U}_T^{1:s}]$ and $V_* = [V^{1:k} \mid \overline{V}_1^{1:s} \mid \cdots \mid \overline{V}_T^{1:s}]$. Moreover, orthogonalize U_* and V_* like in **TSV-merging**: compute the SVD of $U_* = P_{U_*}\Sigma_{U_*}Q_{U_*}^\top$ and $V_* = P_{V_*}\Sigma_{V_*}Q_{V_*}^\top$ and get the whitened matrices like

$$U_\perp = P_{U_*}Q_{U_*}^\top \quad V_\perp = P_{V_*}Q_{V_*}^\top \tag{64}$$

Last, isotropic scaling and reconstruct. Define the scaling factor as

$$\overline{\sigma} = \frac{1}{r}\left(\sum_{i=1}^{k}\sigma_i^{\text{cm}} + \sum_{i=1}^{T}\sum_{j=1}^{s}\sigma_{i,j}^{\text{ts}}\right), \tag{65}$$

and get the merged task matrix

$$\Delta_{\text{Iso-CTS}} = \overline{\sigma}U_\perp V_\perp^\top. \tag{66}$$

Finally, the merged model is:

$$\theta_M^{(l)} = \theta_0^{(l)} + \alpha\Delta_{\text{Iso}}^{(l)}, \tag{67}$$

where Δ_{Iso} is $\Delta_{\text{Iso-C}}$ or $\Delta_{\text{Iso-CTS}}$.

Intuitively, Iso-CTS flattens the singular value spectrum to balance directions and incorporates task-specific subspaces to preserve unique features, achieving state-of-the-art performance especially for large numbers of tasks without additional training. More theoretical properties are shown in Appenix D.

5.3 Centered Arithmetic with Rank-reduced Task Vectors (CART)

STAR (Centered Arithmetic with Rank-reduced Task Vectors) (Choi et al., 2025) proposes a novel training-free model merging approach that re-examines weight averaging through the lens of task arithmetic, introducing centered task vectors and low-rank approximations to mitigate task interference. Unlike traditional task arithmetic, which defines task vectors as $\tau_t = \theta_t - \theta_0$ relative to a pre-trained initialization θ_0, CART redefines task vectors as $\overline{\tau}_t = \theta_t - \theta\text{avg}$, where $\theta_{\text{avg}} = \frac{1}{T}\sum_{t=1}^{T}\theta_t$ is the average of the fine-tuned model parameters across T tasks. This centering ensures that the task vectors sum to zero, reducing interference among tasks, as demonstrated through spectral analysis and empirical results.

CART applies a low-rank approximation to these centered task matrices using Singular Value Decomposition (SVD). Let a centered task matrix $\overline{\Delta}_t$ undergo SVD, retaining only the top-k singular

vectors to form a low-rank approximation $\text{SVD}_k(\overline{\Delta}_t) = \sum_{i=1}^{k} \sigma_i u_i v_i^\top$. The merged model parameters are then computed layer-wise as:

$$\theta_M^{(l)} = \theta_{\text{avg}}^{(l)} + \lambda \sum_{t=1}^{T} \text{SVD}_k(\theta_t^{(l)} - \theta_{\text{avg}}^{(l)}), \tag{68}$$

where $\lambda \in \mathbb{R}$ is a scaling hyperparameter. When $k = r$, this reduces to standard weight averaging, and when $k = 0$, the summation of task vectors becomes zero, resulting in $\theta_{\text{avg}}^{(l)}$. The key innovation lies in selecting an intermediate rank k (empirically found to be around 8% of the full rank), which yields a "bell-curve" performance pattern, peaking significantly above traditional task arithmetic and its variants, such as Ties-Merging (Yadav et al., 2023) and AdaMerging (Yang et al., 2024b). To quantify task interference, CART introduces the Row Space Interference metric $I(k)$, defined as:

$$I(k) := \sum_{i=1}^{T} \sum_{j=1, j\neq i}^{T} \left\| \left(\tilde{\Sigma}_i \tilde{V}_i\right)^\top \left(\tilde{\Sigma}_j \tilde{V}_j\right) \right\|_F, \tag{69}$$

where $\tilde{V}_t$ represents the top-k right singular vectors of the centered task matrix $\overline{\Delta}_t$, and $\tilde{\Sigma}_t$ is a diagonal matrix of normalized singular values. Lower $I(k)$ values indicate reduced interference, and empirical results show that centered task matrices consistently exhibit lower $I(k)$ compared to original task matrices across all ranks. Additionally, the Reconstruction Error $R(k)$ is defined as:

$$R(k) := \sum_{i=1}^{T} \|\theta_i - \theta_{\text{avg}} - \text{SVD}_k(\theta_i - \theta_{\text{avg}})\|_F^2, \tag{70}$$

which measures how well the low-rank approximation reconstructs the centered task vectors. Analysis reveals that $R(k)$ decreases sharply in the low-rank regime, while $I(k)$ increases gradually, explaining the optimal performance at a specific rank. More detailed theoretical derivations of these metrics are provided in Appendix E.

5.4 Spectral Truncation and Rescale (STAR)

STAR (Spectral Truncation and Rescale) (Lee et al., 2025b) improves model merging by transforming task matrices into their spectral spaces using singular value decomposition (SVD), truncating small singular components to mitigate merging conflicts, and rescaling the remaining singular values to preserve the original nuclear norm of the matrices. This method addresses the performance degradation in merged models as the number of tasks increases by removing noisy or redundant dimensions in the spectral domain, effectively reducing interference between task vectors without requiring access to training data or additional fine-tuning. To motivate the benefits of spectral truncation, (Lee et al., 2025b) provides an upper bound on the conflict measure $\|Bx\|$ for data points x from task A's manifold when merging matrices A and B (corresponding to task vectors δ_{T_1} and δ_{T_2}), showing that truncation lowers this bound and potentially reduces conflicts (proof in Appendix F).

For each weight matrix Δ_i in layer l of task i, compute its SVD: $\Delta_i = U\Sigma V^\top$, where $\Sigma = \text{diag}(\sigma_1, \sigma_2, \ldots, \sigma_r)$ contains the singular values sorted in descending order, and r is the matrix rank. Determine the truncation rank r adaptively by retaining the minimal number of singular values that capture at least $\eta\%$ of the total spectral energy:

$$r = \arg\min_k \left\{ k : \frac{\sum_{i=1}^{k} \sigma_i}{\sum_{i=1}^{R} \sigma_i} \geqslant \frac{\eta}{100} \right\}, \tag{71}$$

where $R = \min(m, n)$ is the full possible rank of the matrix (with dimensions $m \times n$), and η is a tunable hyperparameter (typically set to 40 for robustness). This adaptive truncation removes redundant dimensions associated with small singular values, which often correspond to noise or minor details.

After truncation, rescale the retained singular values to restore the original nuclear norm (trace norm) of the matrix, ensuring the "size" of the task matrix is preserved and preventing performance loss on individual tasks. The rescaled singular values are:

$$\sigma'_k = \frac{\sum_{i=1}^{R} \sigma_i}{\sum_{i=1}^{r} \sigma_i} \cdot \sigma_k, \quad \forall k \in [1, r]. \tag{72}$$

Reconstruct the truncated and rescaled matrix as $\Delta_{i,\text{out}} = \sum_{k=1}^{r} u_k \sigma'_k v_k^\top$. Finally, merge the processed task vectors via simple averaging:

$$\Delta_M = \frac{1}{T} \sum_{i=1}^{T} \Delta_{i,\text{out}}, \tag{73}$$

and obtain the merged model parameters:

$$\theta_M = \theta_0 + \Delta_M. \tag{74}$$

Intuitively, STAR leverages SVD to extract principal components while discarding noise in a task- and layer-specific manner, with rescaling compensating for the truncation to maintain the original matrix strength.

5.5 Adaptive Rank Pruning (AdaRank)

AdaRank (Adaptive Rank Pruning) (Lee et al., 2025a) enhances model merging by adaptively selecting beneficial singular directions from task vectors through test-time adaptation, dynamically pruning components that cause cross-task interference to optimize multi-task performance. This approach addresses limitations in prior SVD-based methods, such as suboptimal top-k truncation and fixed-rank application across tasks and layers, which often lead to interference and performance gaps compared to individual fine-tuned models. To motivate the need for adaptive pruning, (Lee et al., 2025a) empirically demonstrates that dominant singular components can increase net multi-task loss by interfering with dissimilar tasks, and that task vectors exhibit varying effective ranks across tasks and layers, necessitating a flexible selection strategy over naive fixed truncation.

For each task matrix $\Delta_i = \theta_i - \theta_0$(neglecting the index of layer) and task i, compute its SVD: $\Delta_i = U_i \Sigma_i (V_i)^\top$, where $\Sigma_i = \text{diag}(\sigma_{i1}, \sigma_{i2}, \ldots, \sigma_{ir})$ contains the singular values sorted in descending order, and r is the matrix rank. Introduce a binary mask vector $B_i \in \{0,1\}^{1\times d}$ and task i, where each element indicates whether the corresponding singular component is preserved (1) or pruned (0). The merged parameters in layer l under the set of masks $B = \{B_1, B_2, \ldots, B_T\}$ are then given by:

$$\theta_M(B) = \theta_0 + \lambda \sum_{i=1}^{T} U_i \left(\text{diag}(B_i) \odot \Sigma_i\right) (V_i)^\top, \tag{75}$$

where λ is a adjustable or learnable layer-wise scaling coefficient (optimized jointly with the masks), and $\odot$ denotes element-wise multiplication. This formulation generalizes standard task arithmetic (when all B_i elements are 1) and top-k truncation (when $B_{ir} = 1$ for $r \leqslant k$ and 0 otherwise), allowing for arbitrary pruning patterns to filter out interfering components.

To optimize the binary masks B (collectively across all layers and tasks) without access to training data or labels, employ test-time adaptation using Shannon entropy minimization as a surrogate objective. Minimize the sum of output entropies H_i for each task:

$$\arg\min_{B} \sum_{i=1}^{T} \mathbb{E}_{x_i \in D_i} \left[H_i(f(\theta(B), x_i)) \right], \tag{76}$$

where D_i is a small unlabeled test dataset for task i, and $f(\theta(B), x_i)$ is the model output parameterized by the merged parameters $\theta(B)$. Optimize B using the Straight-Through Estimator (STE), treating it as continuous during backward passes via a sigmoid function while rounding to binary values in forward passes. Once optimized, deploy the merged model by applying the learned masks in the reconstruction formula, incurring no additional storage overhead.

Intuitively, AdaRank leverages SVD to decompose task vectors into principal directions while using entropy-guided adaptation to prune adversarial components in a task-specific and layer-specific manner, mitigating interference and preserving essential information for superior multi-task merging.

5.6 Knowledge Orientation Through SVD(KnOTS)

KnOTS(Knowledge Orientation Through SVD) (Stoica et al., 2024) enhances merging of LoRA-finetuned models by aligning their task updates through SVD on concatenated weight deltas, transforming them into a shared subspace where existing merging methods can be effectively applied

without data or gradients. This method addresses the challenge that LoRA models exhibit low alignment in their updates compared to full-finetuned counterparts(measured by centered kernel alignment), leading to poor direct merging performance, while KnOTS significantly increases alignment.

For each task i with task matrix $\Delta_i = \theta_i - \theta_0 \in \mathbb{R}^{m\times n}$(neglecting the index of layer) from T LoRA models (rank $r \ll \min(m,n)$) finetuned from shared pretrained weights θ_0, concatenate column-wise across tasks: $[\Delta_1; \Delta_2; \ldots; \Delta_T] \in \mathbb{R}^{m\times Tn}$. Compute the SVD:

$$\text{SVD}\left([\Delta_1; \ldots; \Delta_T]\right) = U\Sigma V^\top = U\Sigma[V_1; \ldots; V_T]^\top, \tag{77}$$

where $U \in \mathbb{R}^{m\times k}$, $\Sigma \in \text{Diag}(\mathbb{R}^k)$, concatenated $V^\top \in \mathbb{R}^{k\times Tn}$ splits into task-specific $V_i \in \mathbb{R}^{k\times n}$, and $k \leqslant \min(O, Tn, Tr)$. The shared $U\Sigma$ forms an aligned output basis with scales, while V_i are input transformations aligned to this basis. Apply a merging method (e.g., TIES with scaling α_i) to the V_i to obtain V_M:

$$V_M = \sum_{i=1}^{n} \alpha_i V_i, \tag{78}$$

after optional pruning/rescaling on ΣV_i for magnitude-based methods. Reconstruct the merged update:

$$\Delta_M = U\Sigma(V_M)^\top, \tag{79}$$

and add to θ_0 for the merged model.

Intuitively, KnOTS jointly decomposes LoRA updates to align their functional subspaces by sharing output basis and scales while merging input-aligned components, reducing interference and enabling robust multi-task merging without additional training.

6 MoE-based Methods

MoE-based (Mixture-of-Experts) methods for model merging draw inspiration from the Mixture-of-Experts architectural paradigm, where different "expert" subnetworks specialize in distinct tasks or data characteristics. Applied to model merging, these strategies move beyond static parameter averages or direct parameter modifications. Instead, they introduce mechanisms to dynamically activate or weight specific knowledge components (often derived from individual fine-tuned models or their task-specific parameters) based on the input or task context. This dynamic integration aims to mitigate task interference, improve adaptability to heterogeneous data, and enhance overall performance by selectively leveraging the most relevant "expert" knowledge for a given input, thereby providing a more flexible and efficient solution for creating multi-task models.

6.1 Sparse Mixture of Low-Rank Experts(SMILE)

SMILE (Sparse Mixture of Low-Rank Experts)(Tang et al., 2024a) introduces a zero-shot approach to construct a Mixture-of-Experts (MoE) model from multiple fine-tuned foundation models without requiring additional data or training. Unlike traditional merging methods that average parameters directly, SMILE upscales source models into a sparse MoE architecture by leveraging subspace analysis to mitigate parameter interference and preserve task-specific knowledge.

The method begins with a subspace perspective on fine-tuning, decomposing the weight updates of fine-tuned models using singular value decomposition (SVD) to identify how pre-trained knowledge is preserved while adapting to downstream tasks. Key observations include: (1) fine-tuning largely retains the most important pre-trained subspaces but utilizes less significant or unused dimensions for task adaptation; and (2) parameter interference, which arises from conflicting updates across tasks, is inherently challenging in the original parameter space but becomes manageable by expanding model dimensions.

Formally, consider T fine-tuned models with weight matrices $W_i = W + \Delta W_i$ (where W_0 is the shared pre-trained weight) and biases $b_i = b_0 + \Delta b_i$. Parameter interference is formulated as an optimization problem to minimize the discrepancy between a merged model's output and individual task outputs:

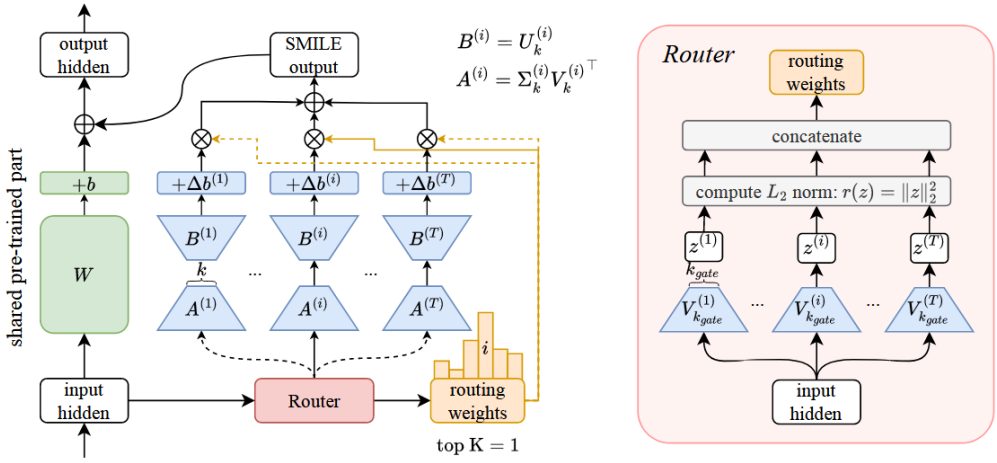

Figure 11: SMILE

$$\min_{\lambda} \|y_M - y_i\|_2^2, \tag{80}$$

where y_M is a weighted combination of fine-tuned parts, and the error is bounded by weight and bias terms (detailed derivation in Appendix L). To resolve this, SMILE constructs an MoE with a shared pre-trained component, a router, and low-rank experts derived from low-rank approximations of ΔW_i via SVD:

$$\Delta W_i \approx U_i^{1:k} \Sigma_i^{1:k} (V_i^{1:k})^\top, \tag{81}$$

where k is the approximation rank. The router computes routing weights based on the ℓ^2 norm of input projections onto the top k_{gate} dimensions of $V_i^{1:k}$:

$$r_i = \left\| (V_i^{1:k_{\text{gate}}})^\top x \right\|_2, \quad p_i = \text{softmax}_i(r_i), \tag{82}$$

followed by top-K selection for sparse activation. The final output combines the pre-trained part with dynamically weighted experts:

$$\begin{aligned} y &= (Wx + b) + \sum_{i=1}^{T} \frac{\lambda_i}{\sum_{j=1}^{T} \lambda_j} \left(U_i^{1:k} \Sigma_i^{1:k} (V_i^{1:k})^\top x + \Delta b_i \right) \\ \lambda_i &= \begin{cases} p_i, & p_i \in \text{TopK}\left(\{p_j\}_{j=1}^{T}, K \right) \\ 0, & \text{otherwise} \end{cases} \\ p_i &= \text{softmax}_i\left(r_i\right) = \text{softmax}_i\left(\left\| (V_i^{1:k})^\top x \right\|_2 \right). \end{aligned} \tag{83}$$

The architecture of SMILE is shown in Fig. 11 and the derivation is in Appendix L.

In general, SMILE exemplifies MoE-based merging by dynamically routing inputs to low-rank task experts, offering a flexible, zero-shot alternative that balances efficiency and multi-task capability while addressing interference through dimensional expansion and subspace-aware decomposition.

6.2 FREE-MERGING

FREE-Merging (Fourier Transform for Efficient Model Merging)(Zheng & Wang, 2025) proposes a two-stage, training-free approach to model merging that addresses task interference in the frequency domain while balancing performance and deployment costs. It consists of FR-Merging

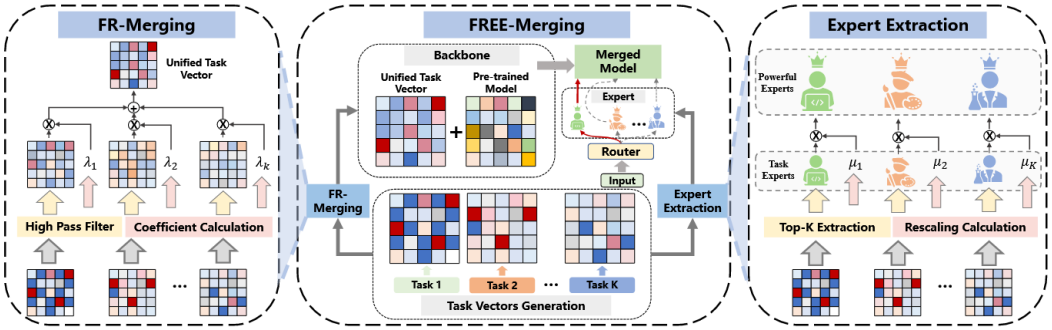

Figure 12: FREE-Merging

for optimizing the merged backbone and lightweight task-specific experts for compensating information loss, enabling a single model to handle multiple tasks with minimal overhead. The workflow is in Fig. 12.

The method identifies that task interference, a primary cause of performance degradation in merging, is particularly evident in the frequency domain of model parameters. Low-frequency components, which capture global structures and task-specific information, exhibit severe misalignment across fine-tuned models, leading to interference during merging. Existing spatial-domain methods fail to mitigate this effectively. To address it, FR-Merging applies high-pass filtering to task vectors to remove harmful low-frequency interference while preserving high-frequency details and model redundancy.

Formally, for a task vector $v(x, y)$, FR-Merging transforms it as:

$$G(x, y) = \mathcal{F}^{-1}\left\{H(\eta, \gamma) \cdot \mathcal{F}\{v(x, y)\}\right\}, \tag{84}$$

where $\mathcal{F}$ and $\mathcal{F}^{-1}$ are the Fourier and inverse Fourier transforms, and $H(\eta, \gamma)$ is the high-pass filter:

$$H(\eta, \gamma) = \begin{cases} 1, & \sqrt{\eta^2 + \gamma^2} \geqslant D_0 \\ 0, & \sqrt{\eta^2 + \gamma^2} < D_0 \end{cases}, \tag{85}$$

with D_0 as the cutoff frequency. Merging coefficients λ_i for task vectors v_i are computed as:

$$\lambda_i = \mathbb{E}(v_i) \left(\sum_{j=1}^{K} \mathbb{E}(v_j) \right)^{-1}, \tag{86}$$

where $\mathbb{E}$ represents the average, ensuring consistent outputs. This constructs a high-quality backbone with reduced interference.

To recover inevitable information loss from filtering, FREE-Merging introduces lightweight experts (about 1% of parameters) extracted from the most changed parameters in task vectors, rescaled for efficiency. For task vector v_i, the expert $e(v_i)$ is:

$$e(v_i) = \mu_i M(v_i, d), \quad \mu_i = -\frac{\mathbb{E}(M(v_i, d)) \cdot \log(d)}{\lambda_i \cdot \mathbb{E}(v_i)}, \tag{87}$$

where $M(v_i, d)$ selects the top-d (percentage) parameters, and μ_i is the rescaling factor.

A router dynamically activates experts based on input, inspired by Mixture-of-Experts (MoE), without increasing inference latency. Theoretical support includes a "no free lunch" theorem proving that merging requires additional information to retain full capabilities (proof in Appendix M).

Generally, FREE-Merging advances MoE-based merging by incorporating frequency-domain filtering and dynamic experts, providing an efficient, adaptable framework that minimizes task interference and resource demands for multi-task model integration.

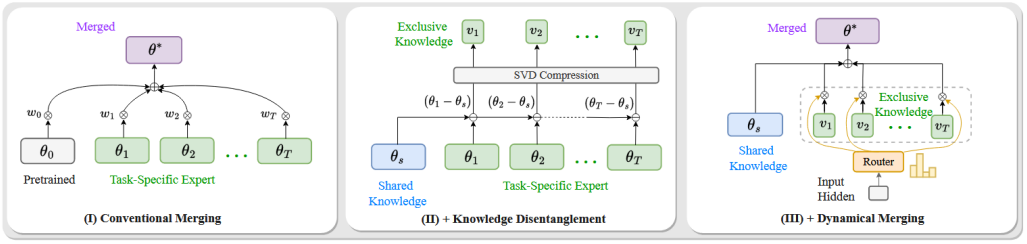

Figure 13: Twin-Merging

6.3 TWIN-MERGING

Twin-Merging (Dynamic Integration of Modular Expertise in Model Merging) (Lu et al., 2024) addresses the challenges of parameter interference and heterogeneous test data in model merging by modularizing knowledge into shared and exclusive components and dynamically integrating them via a routing mechanism. This two-stage approach narrows the performance gap between merged models and individual fine-tuned experts, enabling a single multitask model without additional training. The framework of Twin-Merging is shown in Fig. 13

The method identifies that interference arises from mixed shared (beneficial across tasks) and exclusive (task-specific) knowledge in fine-tuned models. Directly merging exclusive knowledge can degrade performance due to conflicts, while a static merged model struggles with diverse inputs. Twin-Merging first modularizes knowledge: shared knowledge is extracted by averaging task vectors and compressing via SVD, then exclusive knowledge is isolated as the difference between task experts and the shared model, also compressed for efficiency.

Formally, for T task-specific models with weights θ_t (fine-tuned from pre-trained θ_0), the shared expert θ_s is defined as:

$$\theta_s = \theta_0 + \lambda \sum_{t=1}^{T} (\theta_t - \theta_0), \tag{88}$$

where λ is a scaling factor. Exclusive knowledge for task t is $\Delta\theta_t = \theta_t - \theta_s$, compressed using SVD:

$$\Delta\theta_t \approx \hat{U}\hat{\Sigma}\hat{V}^\top, \tag{89}$$

retaining top-k singular values for sparsity (e.g., 99.9% reduction with minimal loss).

In the dynamic merging stage, a router selects and combines shared and exclusive knowledge based on input x:

$$\theta^* = \theta_s + \sum_{t=1}^{T} g_t(x) \cdot \Delta\theta_t, \tag{90}$$

where $g_t(x)$ are routing weights from a lightweight MLP router trained briefly on synthetic data, which can be expressed as follows:

$$\theta^* = \theta_s + \sum_{t=1}^{T} w_t * \mathrm{SVD}_r(\theta_t - \theta_s), \tag{91}$$

$$\{w_1, \ldots, w_T\} = \mathrm{softmax}(\mathcal{R}\left(\mathrm{Emb}(x); \phi\right)), \tag{92}$$

here, $\mathrm{Emb}(x)$ represents the sequence of the last-layer token embeddings from the shared expert $f(x; \theta_s)$.

In short, Twin-Merging enhances MoE-based merging by disentangling and dynamically routing modular knowledge, offering a flexible, training-free solution that mitigates interference and adapts to multitask heterogeneity while maintaining efficiency.

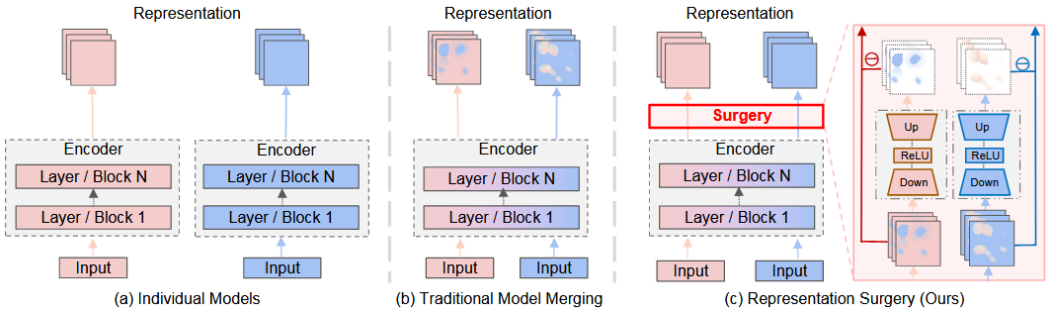

Figure 14: Representation Surgery

6.4 Representation Surgery

Representation Surgery (Surgery)(Yang et al., 2024a) addresses the critical issue of representation bias in model merging, where the merged model's feature representations deviate significantly from those of individual task-specific models, leading to degraded multi-task learning (MTL) performance. This method introduces a lightweight, task-specific module that corrects these biases post-merging, enhancing the merged model's ability to retain the representational capabilities of individual models without requiring raw training data. The schematic diagram is shown in Fig. 14.

The approach is motivated by empirical observations that representation bias—visualized via t-SNE as discrepancies between merged and individual model representations—persists across tasks, architectures, and merging methods like Weight Averaging, Task Arithmetic, and Ties-Merging. This bias is a primary cause of poor MTL performance. Surgery tackles this by appending a task-specific module that takes the merged model's representation as input and filters out biases to align it with the individual model's representation.

Formally, let $f_M(x;\theta_M)$ denote the merged model with parameters θ_M, producing representation $h_M = f_M(x)$. For task t, the individual model $f_t(x;\theta_t)$ yields representation h_t. The Surgery module for task t $S_t(h_M;\phi_t)$, parameterized by ϕ_t, minimizes the representation bias:

$$\min_{\phi_t} \mathbb{E}_x \left\| S_t(h_M;\phi_t) - h_t \right\|_2^2, \tag{93}$$

where the expectation is over input x. The optimization in Eq. 93 aims to align the merged model's representation h_M with the individual model's h_t. Without raw data, a surrogate objective uses proxy data (e.g., synthetic inputs or pre-trained model outputs) to estimate $\mathbb{E}_x$. For a proxy dataset D_{proxy}, the objective becomes:

$$\min_{\phi_t} \frac{1}{|D_{\text{proxy}}|} \sum_{x \in D_{\text{proxy}}} \left\| S_t(f_M(x;\theta_M);\phi_t) - f_t(x;\theta_t) \right\|_2^2. \tag{94}$$

The module S_t is lightweight, typically a small MLP, adding minimal parameters. And it is implemented as a two-layer MLP with ReLU activation in (Yang et al., 2024a), parameterized by $\phi_t = \{W_1, b_1, W_2, b_2\}$. For input $h_M \in \mathbb{R}^d$, the output is:

$$S_t(h_M;\phi_t) = W_2 \cdot \text{ReLU}(W_1 h_M + b_1) + b_2, \tag{95}$$

where $W_1 \in \mathbb{R}^{d' \times d}$, $W_2 \in \mathbb{R}^{d \times d'}$, and $d' \ll d$ ensures lightweight design. Optimization uses gradient descent on the surrogate loss.

Surgery is orthogonal to existing merging methods, enhancing schemes like Task Arithmetic(Ilharco et al., 2023) and AdaMerging(Yang et al., 2024b), and this orthogonality stems from its post-processing nature. For any merged model $\theta_M = \mathcal{A}(\{\lambda_t\}_{t=1}^T, \{\theta_t\}_{t=1}^T)$, Surgery applies S_t to h_M without altering θ_M.

Generally speaking, Representation Surgery advances MoE-based merging by introducing task-specific bias correction, offering a flexible, data-free approach that mitigates representation discrepancies and enhances MTL performance across diverse merging strategies.

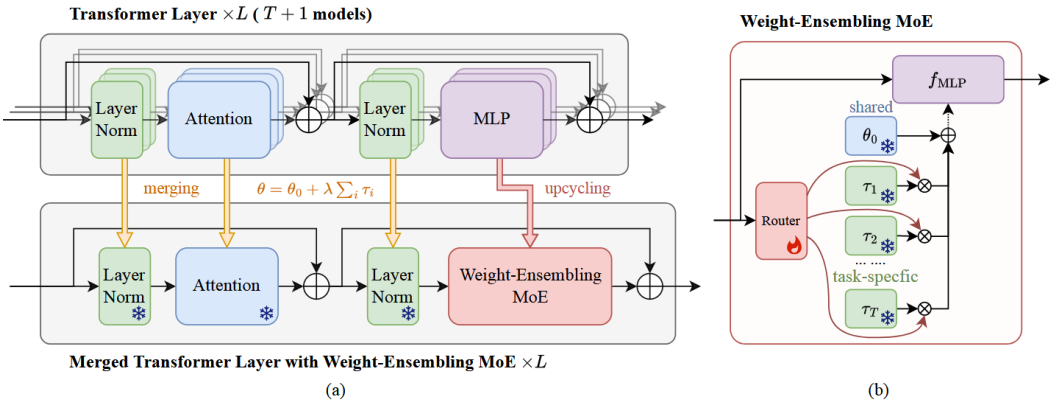

Figure 15: Weight-Ensembling Mixture of Experts (WEMoE)

6.5 Weight-Ensembling Mixture of Experts (WEMoE)

WEMoE (Weight-Ensembling Mixture of Experts) (Tang et al., 2024b) introduces a novel approach to model merging by leveraging a Mixture-of-Experts (MoE) paradigm to dynamically integrate shared and task-specific knowledge, addressing the limitations of static merging methods. Unlike traditional approaches that seek a fixed solution within the original parameter space, WEMoE upscales the multilayer perceptron (MLP) components of Transformer layers into a weight-ensembling MoE module, enabling input-conditioned knowledge integration. This method mitigates parameter interference and enhances adaptability to diverse tasks, achieving superior multi-task performance. The framework of WEMoE is illustrated in Figure 15.

WEMoE tackles the challenge of parameter interference by separating shared knowledge, encapsulated in the pre-trained model's parameters θ_0, from task-specific knowledge, represented by task vectors $\tau_i = \theta_i - \theta_0$. The method merges most Transformer layer parameters using Task Arithmetic(Ilharco et al., 2023), defined as $\theta_M = \theta_0 + \lambda \sum_{i=1}^{T} \tau_i$, where λ is a scaling hyperparameter. However, for the MLP components, WEMoE employs a dynamic approach by upcycling them into a weight-ensembling MoE module.

The WEMoE module comprises three key components: a router, pre-trained MLP weights θ_{MLP_0}, and task vectors $\{\tau_{\text{MLP}_i} = \theta_{\text{MLP}_i} - \theta_{\text{MLP}_0}\}_{i=1}^{T}$. The router, a lightweight MLP, processes input tokens $h_{\text{in}_{1:N}}$ to generate routing weights

$$w = \text{mean}(r(h_{\text{in}_{1:N}})) \in \mathbb{R}^{T \times 1}. \tag{96}$$

These weights dynamically combine the task-specific knowledge with the shared knowledge, producing input-conditioned MLP weights:

$$\theta_{\text{MLP}_M} = \theta_{\text{MLP}_0} + D_\tau w, \tag{97}$$

where $D_\tau \in \mathbb{R}^{|\theta_{\text{MLP}}| \times T}$ is a dictionary matrix of task vectors, and the output is computed as

$$h_{\text{out}_{1:N}} = f_{\text{MLP}}(h_{\text{in}_{1:N}}; \theta_{\text{MLP}}). \tag{98}$$

The router is initialized with Gaussian-distributed weights and biases set to ensure initial routing weights approximate λ, and it is fine-tuned using test-time adaptation with entropy loss minimization on unlabeled test data:

$$L_{\text{entropy}} \approx -\frac{1}{|D|} \sum_{i=1}^{|D|} \sum_{c=1}^{C} p(\hat{y}_c|x_i) \log p(\hat{y}_c|x_i). \tag{99}$$

In summary, WEMoE enhances MoE-based model merging by dynamically integrating shared and task-specific knowledge through a weight-ensembling MoE module. Its dynamic integration allows WEMoE to adaptively select and combine knowledge based on input, significantly reducing interference and improving flexibility. Moreover, this approach offers a flexible, efficient, and robust solution for multi-task model merging, effectively addressing parameter interference and input heterogeneity while maintaining high performance across diverse tasks.

6.6 BUILDING YOUR OWN MULTI-TASK MODEL (BYOM)

BYOM (Building Your Own Multi-Task Model) (Jiang et al., 2024) proposes a parameter-efficient and data-free approach to merge task-specific fine-tuned models into a unified multi-task model, addressing task interference without requiring retraining or access to task-specific data. BYOM introduces two methods, BYOM-FFT for fully fine-tuned models and BYOM-LoRA for LoRA-fine-tuned models, both of which inject compressed task-specific knowledge into a shared model to achieve performance comparable to single-task models. And it tackles task interference by preserving task-specific knowledge while leveraging shared knowledge from a pre-trained model θ_0.

For fully fine-tuned models $\{\theta_i\}_{i=1}^T$, BYOM-FFT first computes a shared model using Task Arithmetic (Ilharco et al., 2023), defined as $\theta^\star = \theta_0 + \lambda \sum_{t=1}^T (\theta_t - \theta_0)$, where λ is a scaling hyperparameter. Task-specific knowledge is then extracted as

$$u_t = \theta_t - \theta^\star, \tag{100}$$

and compressed into a sparse vector $\hat{u}_t(m)$ by retaining the top-$m\%$ (e.g., 1%, 10%) of values based on magnitude:

$$\hat{u}_t(m) = \text{keep top-}m\% \text{ of } u_t \text{based on magnitude.} \tag{101}$$

During inference, the model for task t is $\theta^\star + \hat{u}_t(m)$, combining shared knowledge from $\theta^\star$ with task-specific knowledge from $\hat{u}_t(m)$. This process is data-efficient and computation-efficient as it requires no training or gradient computations.

For LoRA-fine-tuned models, where task-specific models are expressed as $\theta_t = \theta_0 + A_t B_t^\top$ with low-rank matrices $A_t \in \mathbb{R}^{d_{\text{out}} \times r}$ and $B_t \in \mathbb{R}^{d_{\text{in}} \times r}$, BYOM-LoRA compresses task-specific knowledge using singular value decomposition (SVD). The matrix $A_t B_t^\top$ is decomposed as $A_t B_t^\top = U_t \Sigma_t V_t^\top$, and a lower-rank approximation is obtained by retaining the top-q singular values ($q \ll r$):

$$A_t B_t^\top \approx U_t^{1:k} \Sigma_t^{1:k} (V_t^{1:k})^\top, \tag{102}$$

where $U_t^{1:k} \in \mathbb{R}^{d_{\text{out}} \times k}$, $V_t^{1:k} \in \mathbb{R}^{d_{\text{in}} \times k}$, and $\Sigma_t^{1:k} \in \mathbb{R}^{k \times k}$, corresponding to the first k singular vectors or singular values respectively. In inference, the model for task t is $\theta_0 + U_t^{1:k} \Sigma_t^{1:k} (V_t^{1:k})^\top$, significantly reducing the parameter count while preserving task-specific knowledge.

In general, BYOM provides a data-free, computation-efficient, and high-performing solution for multi-task model merging by injecting compressed task-specific knowledge into a shared model. BYOM-FFT and BYOM-LoRA effectively mitigate task interference, offering comparable performance to single-task models while significantly reducing storage and computational costs, making them highly practical for real-world multi-task applications.

7 DETAILED CLASSIFICATION

In this section, we make a more detailed classification on the methods mentioned above. Our classification based on three key aspects:

(1) **Data Involvement Level (DIL)** describes the extent to which a method requires access to datasets during the merging process. This ranges from completely data-agnostic approaches that need no data beyond the fine-tuned models and the pre-trained model themselves (using the abbreviation **NDI** to indicate No Data Involved), to those requiring validation data for hyperparameter tuning or selection (using the abbreviation **VDI** to indicate Validation Data Involved), or test-time adaptation for dynamic merging (using the abbreviation **TTA** to indicate Test-time Adaptation).

(2) **Storage of Additional Components (SAC)** indicates whether the method necessitates saving extra data beyond the final merged model, such as sparse masks, expert models, or other modules (Yes/No).

(3) **Computational Efficiency (CE)** evaluates the computational cost associated with the merging operation, distinguishing between highly efficient methods (**High**), those with medium efficiency (**Medium**), and computationally intensive approaches (**Low**).

The detailed classification is shown in Table 1.

Methods	**DIL**	**SAC**	**CE**
Simple Average	NDI	No	High
Fisher Average	VDI	No	Medium
RegMean	VDI	No	High
Task Arithmetic	VDI	No	High
AdaMerging	TTA	No	Low
Uncertainty-Based Gradient Matching	VDI	No	Medium
NAN	NDI	No	High
TIES-Merging	VDI	No	High
EMR-Merging	NDI	Yes	Medium
Localize-and-Stitch	VDI	No	Low
DARE	VDI	No	High
Model Breadcrumbs	VDI	No	High
Consensus Merging	VDI	No	High
Concrete Subspace Learning	TTA	No	Low
PCB-Merging	VDI	No	Medium
DELLA-Merging	NDI	No	High
TSV-Mering	VDI	No	Medium
Iso-C	VDI	No	Medium
Iso-CTS	VDI	No	Medium
CART	VDI	No	Medium
STAR	VDI	No	Medium
AdaRank	TTA	No	High
KnOTS	VDI	No	Medium
SMILE	VDI	Yes	Medium
FREE-Merging	VDI	Yes	Medium
Twin-Merging	VDI	Yes	Medium
Representation Surgery	VDI	Yes	High
WEMoE	VDI	Yes	Medium
BYOM	NDI	Yes	Medium

Table 1: A Detailed Classification of the above Methods

8 EXPERIMENTS

To thoroughly evaluate the efficacy of various model merging strategies, we conduct experiments centered on the challenging task of merging multiple vision models. Our primary experimental setup involves merging eight independently fine-tuned Vision Transformer (ViT) models, utilizing the **ViT-B-32** architecture as the shared backbone. This particular architecture, being a widely recognized and robust pre-trained model, serves as an excellent foundation for assessing how different merging techniques integrate knowledge from diverse downstream tasks.

Each of the eight ViT-B-32 models is fine-tuned on a distinct image classification dataset. This choice of datasets is critical as it allows us to investigate the performance of merged models across a spectrum of visual recognition challenges, ranging from fine-grained categorization to scene understanding and digit recognition. Following the experimental methodology established in prior works,

Methods	SUN397	Cars	RESISC45	EuroSAT	SVHN	GTSRB	MNIST	DTD	Avg Acc
Individual	79.2	77.7	96.1	99.8	97.5	98.7	99.7	79.4	91.0
Simple Average	64.7	63.3	71.4	72.7	64.2	52.8	87.5	50.1	65.8
Task Arithmetic	55.1	54.9	66.7	77.2	80.2	69.7	97.3	50.1	68.9
LW AdaMerging	64.4	68.3	79.2	93.5	87.3	92.4	97.5	58.4	80.1
LW AdaMerging++	66.9	68.5	81.9	93.7	89.4	89.3	98.3	60.3	81.0
Ties-Merging	58.6	70.7	79.9	86.2	72.1	98.3	54.2	60.3	72.5
EMR-Mering	75.2	72.8	93.5	99.5	96.9	98.1	99.6	74.4	88.7
DARE	54.5	54.0	66.1	74.4	80.5	69.2	97.2	50.6	68.3
TSV-Mering	69.1	70.7	85.5	94.3	92.0	91.9	99.3	69.1	84.0
Iso-C	74.0	72.7	88.0	94.6	88.1	89.7	99.0	69.1	84.4
Iso-CTS	74.6	74.3	89.0	93.6	84.5	90.0	98.8	69.8	84.3
CART	69.3	71.2	87.7	96.9	89.5	94.0	99.2	73.0	85.1
STAR	54.1	54.2	64.8	77.1	79.1	68.9	96.8	49.6	68.1

Table 2: Comparison of different methods across various datasets.

particularly those involving task arithmetic and model merging for vision transformers (Ilharco et al., 2023; Zheng & Wang, 2025), we select the following eight datasets:

1. **Cars**(Krause et al., 2013): A dataset focused on fine-grained recognition of car models, requiring attention to subtle visual details.

2. **DTD (Describable Textures Dataset)**(Cimpoi et al., 2014): Consists of images of different textures, posing a challenge for models to capture abstract textural features.

3. **EuroSAT**(Helber et al., 2019): A land use and land cover classification dataset derived from Sentinel-2 satellite images, representative of remote sensing tasks.

4. **GTSRB (German Traffic Sign Recognition Benchmark)**(Stallkamp et al., 2011): A benchmark for recognizing various traffic signs, demanding high accuracy and robustness to variations.

5. **MNIST**(Deng, 2012): A classic dataset of handwritten digits, often used as a foundational test for classification performance.

6. **RESISC45 (Remote Sensing Image Scene Classification)**(Cheng et al., 2017): Another remote sensing dataset for scene classification, similar to EuroSAT but with a broader range of scene types.

7. **SUN397**(Xiao et al., 2010): A large-scale scene recognition dataset that includes a diverse set of indoor and outdoor scenes.

8. **SVHN (Street View House Numbers)**(Netzer et al., 2011): Comprises images of house numbers cropped from Google Street View, challenging due to variations in font, background, and illumination.

By merging models trained on these disparate datasets, we can thoroughly analyze how each merging method addresses task interference, preserves individual task performance, and potentially enhances generalization capabilities in a multi-task setting.

We selected several typical methods for implement. The following are the experimental results.

The experiments above do not include all methods in this overview, we have selected a few representative methods, excluding MoE-based methods. The experimental results in the table clearly show that the **TTA** methods scores higher than the similar **VDI** methods (LW AdaMerging ¿ Task Arithmetic), which in turn scores higher than the similar **NDI** methods (Task Arithmetic ¿ Simple Average). Furthermore, methods with additional module storage generally perform better than the other similar methods. Moreover, subspace-based methods generally score higher and do not require additional storage, which has inspired us to further explore this type of method.

9 CONCLUSION

In this overview, we have systematically explored the evolving landscape of model merging techniques, particularly in the context of large-scale deep learning models such as LLMs and foundation models. By categorizing existing methods into four primary groups—Weighted Average Strategies, Sparsification Strategies, Subspace-based Methods, and MoE-based Methods—we have highlighted their unique approaches to addressing key challenges like task interference, high-dimensional parameter conflicts, and computational inefficiency.

These methodologies collectively demonstrate that model merging is a powerful, data-efficient alternative to traditional multi-task learning or ensemble methods, preserving the strengths of individually fine-tuned models while minimizing the need for raw training data or extensive retraining. Empirical results across various studies show significant improvements in accuracy, robustness, and generalization, with some techniques narrowing the performance gap to multi-task baselines to as little as 1-3%. However, challenges remain, including scalability to an ever-increasing number of tasks, handling heterogeneous model architectures, and ensuring interpretability in merged parameters.

Looking ahead, future research could focus on hybrid frameworks that combine elements from multiple categories, such as integrating sparsification with MoE routing for even greater efficiency like (Zheng & Wang, 2025). Exploring model merging in emerging domains like multimodal learning, federated systems, and continual adaptation will be crucial. Additionally, theoretical advancements in understanding parameter interference through geometric or information-theoretic lenses could unlock more principled merging algorithms. Ultimately, model merging holds immense promise for democratizing access to advanced AI capabilities, fostering collaborative model development, and paving the way for more sustainable and versatile AI systems in real-world applications.

REFERENCES

Rich Caruana. Multitask learning. *Mach. Learn.*, 28(1):41–75, July 1997. ISSN 0885-6125. doi: 10.1023/A:1007379606734. URL https://doi.org/10.1023/A:1007379606734.

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. doi: 10.1109/JPROC.2017.2675998.

Jiho Choi, Donggyun Kim, Chanhyuk Lee, and Seunghoon Hong. Revisiting Weight Averaging for Model Merging, April 2025.

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

Mircea Cimpoi, Subhransu Maji, Iasonas Kokkinos, Sammy Mohamed, and Andrea Vedaldi. Describing textures in the wild. In *2014 IEEE Conference on Computer Vision and Pattern Recognition*, pp. 3606–3613, 2014. doi: 10.1109/CVPR.2014.461.

Nico Daheim, Thomas Möllenhoff, Edoardo Maria Ponti, Iryna Gurevych, and Mohammad Emtiyaz Khan. Model Merging by Uncertainty-Based Gradient Matching, August 2024.

MohammadReza Davari and Eugene Belilovsky. Model Breadcrumbs: Scaling Multi-Task Model Merging with Sparse Masks, August 2024.

Pala Tej Deep, Rishabh Bhardwaj, and Soujanya Poria. DELLA-Merging: Reducing Interference in Model Merging through Magnitude-Based Sampling, June 2024.

Li Deng. The mnist database of handwritten digit images for machine learning research [best of the web]. *IEEE Signal Processing Magazine*, 29(6):141–142, 2012. doi: 10.1109/MSP.2012.2211477.

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, October 2024.

Antonio Andrea Gargiulo, Donato Crisostomi, Maria Sofia Bucarelli, Simone Scardapane, Fabrizio Silvestri, and Emanuele Rodolà. Task Singular Vectors: Reducing Task Interference in Model Merging, April 2025.

Yifei He, Yuzheng Hu, Yong Lin, Tong Zhang, and Han Zhao. Localize-and-Stitch: Efficient Model Merging via Sparse Task Arithmetic, January 2025.

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. doi: 10.1109/JSTARS.2019.2918242.

Chenyu Huang, Peng Ye, Tao Chen, Tong He, Xiangyu Yue, and Wanli Ouyang. EMR-MERGING: Tuning-Free High-Performance Model Merging.

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, 2022. URL https://arxiv.org/abs/2208.05592.

Gabriel Ilharco, Marco Tulio Ribeiro, Mitchell Wortsman, Suchin Gururangan, Ludwig Schmidt, Hannaneh Hajishirzi, and Ali Farhadi. Editing Models with Task Arithmetic, March 2023.

Weisen Jiang, Baijiong Lin, Han Shi, Yu Zhang, Zhenguo Li, and James T. Kwok. BYOM: Building Your Own Multi-Task Model For Free, February 2024.

Xisen Jin, Xiang Ren, Daniel Preotiuc-Pietro, and Pengxiang Cheng. Dataless Knowledge Fusion by Merging Weights of Language Models, May 2025.

Jonathan Krause, Michael Stark, Jia Deng, and Li Fei-Fei. 3d object representations for fine-grained categorization. In *2013 IEEE International Conference on Computer Vision Workshops*, pp. 554–561, 2013. doi: 10.1109/ICCVW.2013.77.

Chanhyuk Lee, Jiho Choi, Chanryeol Lee, Donggyun Kim, and Seunghoon Hong. AdaRank: Adaptive Rank Pruning for Enhanced Model Merging, March 2025a.

Yu-Ang Lee, Ching-Yun Ko, Tejaswini Pedapati, I.-Hsin Chung, Mi-Yen Yeh, and Pin-Yu Chen. STAR: Spectral Truncation and Rescale for Model Merging, February 2025b.

Zhenyi Lu, Chenghao Fan, Wei Wei, Xiaoye Qu, Dangyang Chen, and Yu Cheng. Twin-Merging: Dynamic Integration of Modular Expertise in Model Merging, October 2024.

Daniel Marczak, Simone Magistri, Sebastian Cygert, Bartłomiej Twardowski, Andrew D. Bagdanov, and Joost van de Weijer. No Task Left Behind: Isotropic Model Merging with Common and Task-Specific Subspaces, June 2025.

Michael Matena and Colin Raffel. Merging Models with Fisher-Weighted Averaging, August 2022.

Yuval Netzer, Tao Wang, Adam Coates, Alessandro Bissacco, Bo Wu, and Andrew Y. Ng. Reading digits in natural images with unsupervised feature learning. In *NIPS Workshop on Deep Learning and Unsupervised Feature Learning 2011*, 2011. URL http://ufldl.stanford.edu/housenumbers/nips2011_housenumbers.pdf.

Omer Sagi and Lior Rokach. Ensemble learning: A survey. *WIREs Data Mining and Knowledge Discovery*, 8(4):e1249, 2018. doi: https://doi.org/10.1002/widm.1249. URL https://wires.onlinelibrary.wiley.com/doi/abs/10.1002/widm.1249.

Chongjie Si, Kangtao Lv, Jingjing Jiang, Yadao Wang, Yongwei Wang, Xiaokang Yang, Wenbo Su, Bo Zheng, and Wei Shen. NAN: A Training-Free Solution to Coefficient Estimation in Model Merging, May 2025.

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, 2011. doi: 10.1109/IJCNN.2011.6033395.

George Stoica, Pratik Ramesh, Boglarka Ecsedi, Leshem Choshen, and Judy Hoffman. Model merging with SVD to tie the Knots, October 2024.

Anke Tang, Li Shen, Yong Luo, Liang Ding, Han Hu, Bo Du, and Dacheng Tao. Concrete Subspace Learning based Interference Elimination for Multi-task Model Fusion, December 2023.

Anke Tang, Li Shen, Yong Luo, Shuai Xie, Han Hu, Lefei Zhang, Bo Du, and Dacheng Tao. SMILE: Zero-Shot Sparse Mixture of Low-Rank Experts Construction From Pre-Trained Foundation Models, August 2024a.

Anke Tang, Li Shen, Yong Luo, Nan Yin, Lefei Zhang, and Dacheng Tao. Merging Multi-Task Models via Weight-Ensembling Mixture of Experts, June 2024b.

Ke Wang, Nikolaos Dimitriadis, Guillermo Ortiz-Jimenez, François Fleuret, and Pascal Frossard. Localizing Task Information for Improved Model Merging and Compression, May 2024.

Mitchell Wortsman, Gabriel Ilharco, Samir Yitzhak Gadre, Rebecca Roelofs, Raphael Gontijo-Lopes, Ari S. Morcos, Hongseok Namkoong, Ali Farhadi, Yair Carmon, Simon Kornblith, and Ludwig Schmidt. Model soups: Averaging weights of multiple fine-tuned models improves accuracy without increasing inference time, July 2022.

Jianxiong Xiao, James Hays, Krista A. Ehinger, Aude Oliva, and Antonio Torralba. Sun database: Large-scale scene recognition from abbey to zoo. In *2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pp. 3485–3492, 2010. doi: 10.1109/CVPR.2010.5539970.

Prateek Yadav, Derek Tam, Leshem Choshen, Colin Raffel, and Mohit Bansal. Ties-merging: Resolving interference when merging models, 2023. URL https://arxiv.org/abs/2306.01708.

Enneng Yang, Li Shen, Zhenyi Wang, Guibing Guo, Xiaojun Chen, Xingwei Wang, and Dacheng Tao. Representation Surgery for Multi-Task Model Merging, May 2024a.

Enneng Yang, Zhenyi Wang, Li Shen, Shiwei Liu, Guibing Guo, Xingwei Wang, and Dacheng Tao. AdaMerging: Adaptive Model Merging for Multi-Task Learning, May 2024b.

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, June 2024.

Shenghe Zheng and Hongzhi Wang. FREE-Merging: Fourier Transform for Efficient Model Merging, March 2025.

A Derivation for the complete formulation of RegMean

Consider merging T linear models, we have the optimization problem formulation,

$$\min_{W} \sum_{i=1}^{T} \|W^\top X_i - W_i^\top X_i\|^2 + \sum_{i=1}^{T} \text{tr}\left[(W - W_i)^T \Lambda_i (W - W_i)\right] \tag{103}$$

where for all i, $W, W_i \in \mathbb{R}^{m\times n}$, $X_i \in \mathbb{R}^{N_i \times m}$, and $\Lambda_i = \text{diag}(\lambda_{i1}, \lambda_{i2}, \ldots, \lambda_{iT}) \geq 0$. The second term is a regularization term that encourages W to be close to W_i, where λ_{ij} is the regularization strength for j-th row of W_i. Here, λ_{ij} can be set as any non-negative values. The optimal solution for this problem is

$$W_M = \left[\sum_{i=1}^{T} \left(X_i^\top X_i + \Lambda_i\right)\right]^{-1} \sum_{i=1}^{T} \left[\left(X_i^\top X_i + \Lambda_i\right) W_i\right]. \tag{104}$$

Proof. We compute the gradient of the objective function (noted as L) w.r.t the merged weight W

$$\frac{\partial L}{\partial W} = \sum_{i=1}^{T} \left(-2X_i^\top X_i W_i + 2X_i^\top X_i W\right) + \sum_{i=1}^{T} \left(-2\Lambda W_i + 2\Lambda W\right) \tag{105}$$

We see L is convex w.r.t. W. Therefore, we may find minizer of L by letting $\frac{\partial L}{\partial W} = 0$.

$$\sum_{i=1}^{T} \left(X_i^\top X_i W_i + \Lambda W_i \right) = \sum_{i=1}^{T} \left(X_i^\top X_i + \Lambda \right) W^* \tag{106}$$

$$W^* = \left[\sum_{i=1}^{T} \left(X_i^\top X_i + \Lambda_i \right) \right]^{-1} \sum_{i=1}^{T} \left[\left(X_i^\top X_i + \Lambda_i \right) W_i \right] \tag{107}$$

Usually, in linear regression, the regularization strength Λ_i is manually specified as a constant value. However, (Jin et al., 2025) states that the scale of $X_i^\top X_i$ may differ a lot across models, layers, or datasets. Therefore, they let Λ_i to scale with $X_i^\top X_i$, and set $\Lambda_i = \gamma \operatorname{diag}(X_i^\top X_i)$, where γ is a fixed scalar, so that,

$$W_M = \left[\sum_{i=1}^{T} \left(X_i^\top X_i + \gamma \operatorname{diag}\left(X_i^\top X_i \right) \right) \right]^{-1} \sum_{i=1}^{T} \left[\left(X_i^\top X_i + \gamma \operatorname{diag}\left(X_i^\top X_i \right) \right) W_i \right] \tag{108}$$

□

B Derivation for Uncertainty-Based Gradient Matching

The derivation begins by defining a target model θ_M trained jointly on all datasets:

$$\theta_M = \arg\max_{\theta} \sum_{t=1}^{T} \alpha_t \bar{\ell}_t(\theta) + \frac{1}{2} \|\theta - \theta_0\|^2_{\mathbf{H}_0}. \tag{109}$$

Individual models θ_t are fine-tuned from θ_0:

$$\theta_t = \arg\min_{\theta} \bar{\ell}_t(\theta) + \frac{1}{2} \|\theta - \theta_0\|^2_{\mathbf{H}_0}. \tag{110}$$

The stationarity conditions are:

$$\mathbf{H}_0(\theta_M - \theta_0) = -\sum_{t=1}^{T} \alpha_t \nabla \bar{\ell}_t(\theta_M), \tag{111}$$

$$\mathbf{H}_0(\theta_t - \theta_0) = -\nabla \bar{\ell}_t(\theta_t). \tag{112}$$

Multiplying the second by α_t, summing over t, and subtracting from the first yields:

$$\theta_M - \left(\theta_0 + \sum_{t=1}^{T} \alpha_t (\theta_t - \theta_0) \right) = -\sum_{t=1}^{T} \alpha_t \mathbf{H}_0^{-1} \left(\nabla \bar{\ell}_t(\theta_M) - \nabla \bar{\ell}_t(\theta_t) \right). \tag{113}$$

This shows the error in task arithmetic equals the weighted gradient mismatch. To reduce it, apply a first-order Taylor approximation:

$$\nabla \bar{\ell}_t(\theta_M) \approx \nabla \bar{\ell}_t(\theta_t) + \mathbf{H}_t(\theta_M - \theta_t). \tag{114}$$

Substituting and rearranging gives:

$$\begin{aligned} \theta_M - \theta_0 &\approx \sum_{t=1}^{T} \alpha_t (\theta_t - \theta_0) - \sum_{t=1}^{T} \alpha_t \mathbf{H}_0^{-1} \mathbf{H}_t (\theta_M - \theta_t) \\ &= \sum_{t=1}^{T} \alpha_t (\theta_t - \theta_0) - \sum_{t=1}^{T} \alpha_t \mathbf{H}_0^{-1} \mathbf{H}_t \left[(\theta_M - \theta_0) - (\theta_t - \theta_0) \right] \end{aligned} \tag{115}$$

Solving for θ_M:

$$\left(\mathbf{H}_0 + \sum_{t=1}^{T} \alpha_t \mathbf{H}_t \right) (\theta_M - \theta_0) \approx \sum_{t=1}^{T} \alpha_t (\mathbf{H}_0 + \mathbf{H}_t)(\theta_t - \theta_0), \tag{116}$$

yielding the merged model:

$$\hat{\theta}_M = \theta_0 + \sum_{t=1}^{T} \alpha_t \left(\bar{\mathbf{H}}^{-1}\mathbf{H}_{0+t}\right)(\theta_t - \theta_0). \tag{117}$$

This is the MAP of the Laplace-approximated posterior

$$q_\alpha(\theta \mid D_{1:T}) \propto p(\theta) \prod_{t=1}^{T} e^{-\alpha_t \cdot \frac{1}{2}\|\theta-\theta_t\|^2_{\mathbf{H}_t}}. \tag{118}$$

C Derivation for Task Singular Vector Merging

C.1 Equivalence between Whitening and Procrustes

For merging (TSV-M), form reduced concatenations $\hat{U} = [\hat{U}_1 \cdots \hat{U}_T]$, $\hat{V}^\top = [\hat{V}_1^\top; \cdots; \hat{V}_T^\top]$, $\hat{\Sigma} = \mathrm{blkdiag}(\hat{\Sigma}_1, \ldots, \hat{\Sigma}_T)$. To reduce the measure of task interference, (Gargiulo et al., 2025) decorrelate the TSVs of different tasks (encoded as columns in $\hat{U}$ and $\hat{V}$) by whitening these matrices to minimize their correlations. This can be achieved by applying the transformation $X \mapsto X(X^\top X)^{-1/2}$ to both $\hat{U}$ and $\hat{V}$.

For improved numerical stability, we reformulate this whitening as an orthogonal Procrustes problem, seeking the orthogonal matrix $\hat{U}_\perp$ that minimizes the projection error:

$$\min_{\hat{U}_\perp} \|\hat{U}_\perp - \hat{U}\|_F \quad \text{s.t.} \quad \hat{U}_\perp^\top \hat{U}_\perp = I. \tag{119}$$

This problem admits a closed-form solution via the SVD $\hat{U} = PDQ^\top$, yielding $\hat{U}_\perp = PQ^\top$ (similarly $\hat{V}_\perp = PQ^\top$ for $\hat{V} = PDQ^\top$).

Proposition 1. *The transformations $X \mapsto X(X^\top X)^{-1/2}$ (whitening) and $X \mapsto PQ^\top$ (Procrustes), where $X = PDQ^\top$ is the SVD of X, are equivalent.*

Proof. Given $X = PDQ^\top$, $X^\top X = QD^2Q^\top$, so $(X^\top X)^{-1/2} = QD^{-1}Q^\top$. Thus:

$$X(X^\top X)^{-1/2} = (PDQ^\top)(QD^{-1}Q^\top) = PDID^{-1}Q^\top = PQ^\top. \tag{120}$$

The merged aggregation is $\hat{M} = \hat{U}_\perp \hat{\Sigma} \hat{V}_\perp^\top$, yielding the final weights. □

C.2 Correlation between T and k

Theorem 1. *Let $T \in \mathbb{N}$ such that $T > 4$. Define $U = [U_1, \ldots, U_T]$ as the matrix obtained by concatenating T orthogonal matrices U_i, each of shape $n \times n$. Let $\widehat{U} = [\widehat{U}_1, \ldots, \widehat{U}_T]$ be the matrix formed by truncating each U_i to its first k columns. Denote by X and $\widehat{X}$ the matrices resulting from Procrustes orthonormalization of U and $\widehat{U}$, respectively. If $k \leqslant n\frac{T-2\sqrt{T}}{T}$, then $\|U - X\|_F \geqslant \|\widehat{U} - \widehat{X}\|_F$.*

Proof. Let us consider the SVD decomposition of U and $\widehat{U}$: $U = P_u \Sigma_u P_v$ and $\widehat{U} = R_u \widehat{\Sigma_u} R_v$. X and $\widehat{X}$ obtain as $X = P_u P_v^\top$, $\widehat{X} = R_u R_v^\top$ respectively. We first consider the Frobenius norm of $\|X - U\|_F$. Notice that the singular values of U are the square root of the eigenvalues of $\Sigma_u = UU^\top$.

$UU^\top = \sum_{i=1}^{T} U_i U_i^\top = TI_n$. As a consequence, the eigenvalues of $UU^\top$ are all equal to T and the singular values are all equal to $\sqrt{T}$.

$$\begin{aligned} \|X - U\|_F &= \|P_u P_v^\top - P_u \Sigma_u P_v^\top\|_F \\ &= \|P_u(I - \Sigma_u)P_v^\top\|_F \\ &= \|I_n - \Sigma_u\|_F \\ &= \|I_n - \sqrt{T} I_n\|_F \\ &= \sqrt{n}(\sqrt{T} - 1). \end{aligned} \tag{121}$$

We are now left to compute $\|\widehat{X} - \widehat{U}\|_F$. In this case, we are not able to compute the exact norm without other assumptions, but we can provide an upper bound that gives us a sufficient condition to prove our statement. As before

$$\begin{aligned} \|\widehat{X} - \widehat{U}\|_F &= \|I_n - \widehat{\Sigma_u}\|_F \\ &= \sqrt{\sum_{i=1}^{n} (\hat{\sigma}_i - 1)^2}, \end{aligned} \tag{122}$$

where $\hat{\sigma}$ are the singular values of $\widehat{U}$.

Notice that $\sum_{i=1}^{n} \hat{\sigma}_i^2 = \mathrm{tr}(\widehat{U}\widehat{U}^\top) = \mathrm{tr}(\widehat{U}^\top\widehat{U})$ and $\widehat{U}^\top\widehat{U}$ is a $Tk \times Tk$ matrix with diagonal elements equal to one, so $\mathrm{tr}(\widehat{U}^\top\widehat{U}) = kT$.

Moreover,

$$\begin{aligned} \sum_{i=1}^{n} \hat{\sigma}_i &= \sum_{i=1}^{n} \sqrt{\lambda_i(\widehat{U}\widehat{U}^\top)} \\ &\geqslant \sqrt{\sum_{i=1}^{n} \lambda_i(\widehat{U}\widehat{U}^\top)} \\ &= \sqrt{\mathrm{tr}(\widehat{U}\widehat{U}^\top)} = \sqrt{Tk}. \end{aligned} \tag{123}$$

Notice that this upper bound is tight, indeed the sum of the singular values of $\widehat{U}$ must lie within: $\sqrt{kT} \leqslant \sum_{i=1}^{n} \hat{\sigma}_i \leqslant kT$. The minimum $\sqrt{kT}$ is achieved if all matrices U_i are equal, on the other end, the maximum kT is achieved if the kT columns are orthonormal.

Putting everything together,

$$\begin{aligned} \|\widehat{X} - \widehat{U}\|_F &= \sqrt{n + \sum_{i=1}^{n} \hat{\sigma}_i^2 - 2\sum_{i=1}^{n} \hat{\sigma}_i} \\ &= \sqrt{n + kT - 2\sum_{i=1}^{n} \hat{\sigma}_i} \\ &\leqslant \sqrt{n + kT - 2\sqrt{kT}}. \end{aligned} \tag{124}$$

So we have to check for what values of k it holds that $\sqrt{n + kT - 2\sqrt{kT}} \leqslant \sqrt{n}(\sqrt{T} - 1)$.

We have that $\sqrt{n + kT - 2\sqrt{kT}} \leqslant \sqrt{n + kT} \leqslant \sqrt{n}(\sqrt{T} - 1)$.

The inequality is satisfied if $k \leqslant n\frac{T - 2\sqrt{T}}{T}$. This concludes the proof. Since k is a positive number the inequality is meaningful for $T > 4$. □

D THEORETICAL PROPERTIES OF ISO-C

In this Appendix, we discuss the theoretical properties of Iso-C by explicitly showing the connection between spectral skewness and the increased subspace dimensionality k_{M} in the merged model achieved by Iso-C, which leads to a higher Subspace Alignment Ratio (SAR). Moreover, we explain why the increased SAR reduces inter-task interference. Finally, we highlight the limitations of Iso-C that lead to the development of Iso-CTS.

D.1 SPECTRAL SKEWNESS AND THE DEFINITION OF k_{M}

In this Section, we show that the number of dominant components k_{M} of the merged model Δ_{M} is directly influenced by the skewness of its singular value spectrum. Using the singular value decom-

position (SVD), let $\Delta_{\text{M}} = U\Sigma V^T$, where $\Sigma = \text{diag}(\sigma_1, \ldots, \sigma_r)$. By the definition of Frobenius norm:

$$\|\Delta_{\text{M}}\|_F^2 = \sum_{i=1}^{r} \sigma_i^2, \quad \|\Delta_{\text{M}} - \Pi_{k,\text{M}}\Delta_{\text{M}}\|_F^2 = \sum_{i=k+1}^{r} \sigma_i^2. \tag{125}$$

Hence, the relative approximation error becomes:

$$\frac{\|\Delta_{\text{M}} - \Pi_{k,\text{M}}\Delta_{\text{M}}\|_F^2}{\|\Delta_{\text{M}}\|_F^2} = \frac{\sum_{i=k+1}^{r} \sigma_i^2}{\sum_{i=1}^{r} \sigma_i^2}. \tag{126}$$

Accordingly, k_{M} can be defined in terms of singular values:

$$k_{\text{M}} = \min\left\{k : \frac{\sum_{i=k+1}^{r} \sigma_i^2}{\sum_{i=1}^{r} \sigma_i^2} \leqslant \epsilon^2\right\}. \tag{127}$$

This formulation explicitly shows how the skewness of the spectrum $\{\sigma_i\}$ controls k_{M}. When Δ_{M} has a skewed spectrum (e.g. $\sigma_1^2 \gg \sum_{i=2}^{r} \sigma_i^2$), a small k_{M} is sufficient to meet the error bound. This explains why Task Arithmetic Δ_{TA} ($\beta = 0$ in Figure 4a) — which has a skewed spectrum — yields a smaller k_{TA} than Iso-C, whose flatter spectrum leads to a larger $k_{\text{Iso-C}}$. Therefore, expressing k_{M} directly in terms of singular values highlights the link between the spectral skewness and subspace dimensionality.

D.2 Iso-C increases Subspace Alignment Ratio (SAR)

In this Section, we formally show how Iso-C increases Subspace Alignment Ratio (SAR) by expanding the effective subspace dimensionality of the merged model — from k_{TA} in Task Arithmetic to $k_{\text{Iso-C}}$ in Iso-C.

The rank k_{M} defines the effective rank of the subspace identified by the merged model and it is directly determined directly by its spectrum (as discussed Appendix D.1). Let k_{TA} be the effective rank of Δ_{TA}, and define

$$T = \{u_1, .., u_{k_{\text{TA}}}\}$$

as the orthonormal basis formed by those k_{TA} singular vectors. Flattening the spectrum of Δ_{TA}, yields $\Delta_{\text{Iso-C}}$ with effective rank $k_{\text{Iso-C}} > k_{\text{TA}}$ (as discussed in Appendix D.1). This flattening modifies only the singular values of TA, leaving the singular vectors unchanged. Therefore, the original subspace T is contained within the larger subspace spanned by the top singular vectors of $\Delta_{\text{Iso-C}}$, defined as:

$$I = \{u_1, .., u_{k_{\text{TA}}}, .., u_{k_{\text{Iso-C}}}\}.$$

Thus, by construction, we have $T \subset I$.

For simplicity, let $\Pi_T = \Pi_{k_{\text{TA}},\text{TA}}$ and $\Pi_I = \Pi_{k_{\text{Iso-C}},\text{Iso-C}}$ denote the projection operators onto the subspaces spanned by T and I, respectively. Since $T \subset I$, for any matrix Δ_t it holds that:

$$\begin{aligned}\text{SAR}(\Delta_t, \Delta_{\text{TA}}; k_{\text{TA}}) &= \frac{\|\Pi_T \Delta_t\|_F}{\|\Delta_t\|_F} \leqslant \frac{\|\Pi_I \Delta_t\|_F}{\|\Delta_t\|_F} \\ &= \text{SAR}(\Delta_t, \Delta_{\text{Iso-C}}; k_{\text{Iso-C}}),\end{aligned} \tag{128}$$

This inequality holds because by definition:

$$\frac{\|\Pi_T \Delta_t\|_F^2}{\|\Delta_t\|_F^2} = \frac{\sum_{i=1}^{k_{\text{TA}}} \sum_j \langle u_i, \Delta_t^{(j)} \rangle^2}{\|\Delta_t\|_F^2} \leqslant \frac{\sum_{i=1}^{k_{\text{TA}}} \sum_j \langle u_i, \Delta_t^{(j)} \rangle^2 + \sum_{i=k_{\text{TA}}+1}^{k_{\text{Iso-C}}} \sum_j \langle u_i, \Delta_t^{(j)} \rangle^2}{\|\Delta_t\|_F^2} = \frac{\|\Pi_I \Delta_t\|_F^2}{\|\Delta_t\|_F^2}, \tag{129}$$

where $\Delta_t^{(j)}$ denotes the j-th column of Δ_t.

The equality in Equation equation **??** holds only if the additional vectors added to the basis T — that is $\{u_{k_{\text{TA}}+1}, \ldots, u_{k_{\text{Iso-C}}}\}$ — are orthogonal to each $\Delta_t^{(j)}$ or, equivalently, if they do not intersect the column space of Δ_t (i.e. its left singular vectors).

Hence, in general a lower k_{M} yields smaller or equal SAR than a larger k_{M}. However, our empirical findings show that enriching the basis T with singular vectors corresponding to smaller singular values in original task arithmetic spectrum (i.e. $\{u_{k_{\text{TA}}+1}, \ldots, u_{k_{\text{Iso-C}}}\}$) consistently increases the

alignment ratio, implying that these vectors are relevant for representing each task matrix Δ_t and not orthogonal to its left singular vectors.

This analysis formally supports the claim that increasing the effective rank k_{M} of the merged matrix — achieved by spectrum flattening in Iso-C — leads to a higher Subspace Alignment Ratio.

D.3 ISO-C MITIGATES INTER-TASK INTERFERENCE

Iso-C increases the Subspace Alignment Ratio (SAR), which quantifies how well the principal directions of a task matrix align with the principal directions of the merged model. In this Section, we demonstrate how a higher SAR contributes to mitigate inter-task interference by analyzing the relationship between subspace alignment and changes in internal activations following merging. Specifically, we define the interference as the degradation in a task's internal representation due to merging — that is, the deviation between the activations of the merged model and those of the corresponding single-task fine-tuned model. Intuitively, we can minimize the task interference by ensuring that the internal representations of task j remain stable after merging.

Let θ_0 be the pre-trained weights for a layer l. Define the task matrix $\Delta_j = \theta_j - \theta_0$ and the merged task matrix Δ_{M} for the layer l. Then, for an input $x_j^{(l)}$, we desire that the post-merging activation $h_j^{(l)} = (\theta_0 + \alpha\Delta_{\mathrm{M}})x_j^{(l)}$, with α chosen on a validation set, be close to the task-specific activation $\hat{h}_j^{(l)} = (\theta_0 + \Delta_j)x_j^{(l)}$. Hence, we can quantify the interference as:

$$\|\hat{h}_j^{(l)} - h_j^{(l)}\| = \|(\Delta_j - \alpha\Delta_{\mathrm{M}})x_j^{(l)}\| \leqslant \|\Delta_j - \alpha\Delta_{\mathrm{M}}\| \cdot \|x_j^{(l)}\|. \tag{130}$$

To show that the interference is lower when the Subspace Alignment Ratio (SAR) between Δ_j and Δ_{M} is higher, we decompose Δ_j into components aligned with and orthogonal to Δ_{M}:

$$\begin{aligned} &\Delta_j = \Delta_j^{\parallel} + \Delta_j^{\perp} \quad \text{where} \quad \Delta_j^{\parallel} = \Pi_{k_{\mathrm{M}},\mathrm{M}}\Delta_j, \\ &\Delta_j^{\perp} = (I - \Pi_{k_{\mathrm{M}},\mathrm{M}})\Delta_j, \end{aligned} \tag{131}$$

and $\Pi_{k_{\mathrm{M}},\mathrm{M}}$ is the projection matrix onto the subspace spanned by the top k_{M} left-singular vectors of Δ_{M} (see Equation (6) for the definition of k_{M}). The Subspace Alignment Ratio is then:

$$\mathrm{SAR}(\Delta_j, \Delta_{\mathrm{M}}; k_{\mathrm{M}}) = \frac{\|\Pi_{k_{\mathrm{M}},\mathrm{M}}\Delta_j\|_F}{\|\Delta_j\|_F} = \frac{\|\Delta_j^{\parallel}\|_F}{\|\Delta_j^{\parallel} + \Delta_j^{\perp}\|_F}. \tag{132}$$

Similarly, decomposing Δ_{M} into $\Delta_{\mathrm{M}}^{\parallel}$ and $\Delta_{\mathrm{M}}^{\perp}$ and substituting Equation equation 131 in Equation equation 130, the interference becomes:

$$\|\Delta_j - \alpha\Delta_{\mathrm{M}}\| = \|\Delta_j^{\parallel} - \alpha\Delta_{\mathrm{M}}^{\parallel} + \Delta_j^{\perp} - \alpha\Delta_{\mathrm{M}}^{\perp}\| \approx \|\Delta_j^{\parallel} - \alpha\Delta_{\mathrm{M}}^{\parallel} + \Delta_j^{\perp}\|, \tag{133}$$

since k_{M} minimizes the approximation error of Δ_{M}, leading to $\|\Delta_{\mathrm{M}}^{\perp}\| \approx 0$.

If the SAR defined in Equation equation 132 is close to 1, then $\|\Delta_j^{\perp}\|$ is small, so the interference in Equation equation 133 mainly depends on $\|\Delta_j^{\parallel} - \alpha\Delta_{\mathrm{M}}^{\parallel}\|$. Conversely, if SAR is near zero, the large orthogonal component $\Delta_j^{\perp}$ increases the overall interference, regardless of the choice of α. Even with an optimal α chosen via validation, interference cannot be reduced below the norm of the orthogonal component.

Iso-C increases the SAR of Δ_t with the merged model — bringing it close to 1, as shown in the paper — by flattening the singular values. Thus, the optimal α can adjust the merged model such that interference is minimized. In contrast, Task Arithmetic (TA), with SAR varying across tasks, exhibits interference that cannot be reduced below the norm of the orthogonal component. We experimentally evaluate that the interference is lower for Iso-C than TA in Appendix D.2.

D.4 LIMITATIONS OF ISO-C THAT MOTIVATE ISO-CTS

This Section details the limitations of Iso-C that motivate the development of Iso-CTS. Specifically, Iso-C relies on the singular vectors obtained through Task Arithmetic to perform model merging. As

a result, it tends to underrepresent tasks whose dominant directions have lower intensity compared to the majority, particularly when those directions are orthogonal to the shared (common) directions. This limitation becomes increasingly pronounced as the number and diversity of tasks increase.

To make this limitation explicit, we formalize the computation — via SVD — of the first left singular vector in Task Arithmetic, used by Iso-C, as the variance maximization problem:

$$\begin{aligned} u_1 &= \arg\max_{\|u\|=1} \|\Delta_{\text{TA}}^\top u\|^2 \\ &= u^\top \left(\sum_{t=1}^{T} \Delta_t \Delta_t^\top \right) u + u^\top \left(\sum_{\substack{t,s=1 \\ t \neq s}}^{T} \Delta_t \Delta_s^\top \right) u. \end{aligned} \tag{134}$$

If a particular task Δ_j has dominant directions with significantly lower intensity compared to the other tasks (i.e. lower Frobenius Norm), then its individual contributions $\Delta_j \Delta_j^\top$ to the total variance becomes smaller. Similarly, cross terms involving Δ_j will also be comparatively small. Therefore, task j explicitly contributes less to the maximized variance captured by the first principal singular direction.

Moreover, if the directions of Δ_j are orthogonal or nearly orthogonal to u_1, (i.e. $u_1^\top \Delta_j = 0$), task j contributes minimally or not at all along this principal direction. Similar considerations apply to subsequent singular vectors $u_2, \dots u_k$, defining the common subspace. Finally, as the number of tasks T increases and tasks become more diverse, it becomes increasingly likely that tasks with distinct but smaller-magnitude directions will be underrepresented or absent in the dominant singular directions identified by the task arithmetic decomposition.

The goal of Iso-CTS is to address this limitation by incorporating orthogonal directions that are overlooked by the Task Arithmetic spectrum. This strategy yields the greatest improvements in settings with a large number of diverse tasks, as shown in our experimental results.

E CART Interference Analysis

In this section, we may confuse vector and matrix. Please note that the vector we refer to here may be two-dimensional.

To understand why CART's centered task vectors reduce interference, we derive the relationship between the Row Space Interference $I(k)$ and task interference, as introduced in (Choi et al., 2025).

Consider a multi-task model defined as $\theta_{\text{MTL}} = \theta_0 + \sum_{t=1}^{T} \tau_t$, where $\tau_t = \theta_t - \theta_0$ are the original task vectors, and inputs $x_{t,i}$ for task t lie close to the subspace spanned by the row space of τ_t. The task interference L is defined as:

$$L := \sum_{t=1}^{T} \sum_{i=1}^{n} \|\theta_{\text{MTL}} x_{t,i} - (\theta_0 + \tau_t) x_{t,i}\|_2^2 . \tag{135}$$

For centered task vectors $\overline{\tau}_t = \theta_t - \theta\text{avg}$, the multi-task model can be expressed as $\theta_{\text{MTL}} = \theta_{\text{avg}} + \sum_{t=1}^{T} \text{SVD}_k(\overline{\tau}_t)$. The goal is to show that a smaller $I(k)$ implies reduced task interference, where Row space Interference $I(k)$ is defined as:

$$I(k) := \sum_{i=1}^{T} \sum_{j=1, j\neq i}^{T} \left\| \left(\tilde{\Sigma}_i \tilde{V}_i \right)^\top \left(\tilde{\Sigma}_j \tilde{V}_j \right) \right\|_F . \tag{136}$$

Theorem 2. $L \leqslant O(I^2)$.

Proof. To derive the bound, we assume that inputs $x_{t,i}$ for task t lie approximately in the row space of τ_t. The interference term for task t is:

$$\|\theta_{\text{MTL}} x_{t,i} - (\theta_0 + \tau_t) x_{t,i}\|_2^2 = \left\| \sum_{j \neq t} \tau_j x_{t,i} \right\|_2^2 . \tag{137}$$

This term quantifies the impact of other task vectors τ_j (for $j \neq t$) on the transformation of input $x_{t,i}$. For centered task vectors, we use $\overline{\tau}_t = \theta_t - \theta_{\text{avg}}$, and the multi-task model is $\theta_{\text{MTL}} = \theta_{\text{avg}} + \sum_{t=1}^{T} \text{SVD}_k(\overline{\tau}_t)$, where $\text{SVD}_k(\overline{\tau}_t) = \sum_{i=1}^{k} \sigma_i u_i v_i^\top$ retains the top-k singular vectors from the Singular Value Decomposition (SVD) of $\overline{\tau}_t$.

Since $\theta_{\text{avg}} = \frac{1}{T}\sum_{t=1}^{T} \theta_t$, the centered task vectors satisfy $\sum_{t=1}^{T} \overline{\tau}_t = \sum_{t=1}^{T} (\theta_t - \theta_{\text{avg}}) = 0$, which reduces the magnitude of interference terms by ensuring that the task vectors sum to zero. We express the task vector τ_t (or $\overline{\tau}_t$) via SVD as:

$$\tau_t = \sum_{i=1}^{r} \sigma_{t,i} u_{t,i} v_{t,i}^\top, \tag{138}$$

where r is the rank of the matrix, $\sigma_{t,i}$ are the singular values, and $u_{t,i}$, $v_{t,i}$ are the left and right singular vectors, respectively. For the low-rank approximation, we use:

$$\text{SVD}_k(\tau_t) = \sum_{i=1}^{k} \sigma_{t,i} u_{t,i} v_{t,i}^\top. \tag{139}$$

Assume that the input $x_{t,i}$ lies approximately in the row space of τ_t, so we can write:

$$x_{t,i} \approx \sum_m \alpha_{t,m} v_{t,m}, \tag{140}$$

where $v_{t,m}$ are the right singular vectors of τ_t, and $\alpha_{t,m}$ are coefficients. The interference term becomes:

$$\left\| \sum_{j \neq t} \tau_j x_{t,i} \right\|_2^2 = \left\| \sum_{j \neq t} \tau_j \sum_m \alpha_{t,m} v_{t,m} \right\|_2^2. \tag{141}$$

Substituting the SVD of τ_j:

$$\tau_j x_{t,i} = \sum_{i=1}^{k} \sigma_{j,i} u_{j,i} \left(v_{j,i}^\top \sum_m \alpha_{t,m} v_{t,m} \right) = \sum_{i=1}^{k} \sum_m \sigma_{j,i} \alpha_{t,m} \left(v_{j,i}^\top v_{t,m} \right) u_{j,i}. \tag{142}$$

Thus, the interference term is:

$$\left\| \sum_{j \neq t} \tau_j x_{t,i} \right\|_2^2 = \left\| \sum_{j \neq t} \sum_{i=1}^{k} \sum_m \sigma_{j,i} \alpha_{t,m} \left(v_{j,i}^\top v_{t,m} \right) u_{j,i} \right\|_2^2. \tag{143}$$

Since the left singular vectors $u_{j,i}$ are orthonormal ($u_{j,i}^\top u_{j,i'} = \delta_{ii'}$), the squared norm is:

$$\left\| \sum_{j \neq t} \tau_j x_{t,i} \right\|_2^2 = \sum_{j \neq t} \sum_{i=1}^{k} \sum_m \sigma_{j,i}^2 \alpha_{t,m}^2 \left(v_{j,i}^\top v_{t,m} \right)^2, \tag{144}$$

assuming orthogonality between $u_{j,i}$ and $u_{j',i}$ for $j \neq j'$. The term $\left(v_{j,i}^\top v_{t,m} \right)^2$ represents the overlap between the row spaces of tasks j and t. Summing over all tasks and inputs:

$$L = \sum_{t=1}^{T} \sum_{i=1}^{n} \sum_{j \neq t} \sum_{i'=1}^{k} \sum_m \sigma_{j,i'}^2 \alpha_{t,m}^2 \left(v_{j,i'}^\top v_{t,m} \right)^2. \tag{145}$$

The Row Space Interference $I(k)$ involves normalized singular values, where $\tilde{\Sigma}_t = \text{diag}(\tilde{\sigma}_{t,1}, \ldots, \tilde{\sigma}_{t,k})$, and $\tilde{\sigma}_{t,i} = \sigma_{t,i}/\sqrt{\sum_{i=1}^{k} \sigma_{t,i}^2}$. Thus:

$$\left(\tilde{\Sigma}_i \tilde{V}_i\right)^\top \left(\tilde{\Sigma}_j \tilde{V}_j\right) = \sum_{i'=1}^{k} \tilde{\sigma}_{i,i'} \tilde{\sigma}_{j,i'} v_{i,i'}^\top v_{j,i'}, \tag{146}$$

and the Frobenius norm is:

$$\left\| \left(\tilde{\Sigma}_i \tilde{V}_i\right)^\top \left(\tilde{\Sigma}_j \tilde{V}_j\right) \right\|_F^2 = \sum_{i'=1}^{k} \sum_{m=1}^{k} \tilde{\sigma}_{i,i'}^2 \tilde{\sigma}_{j,m}^2 \left(v_{i,i'}^\top v_{j,m}\right)^2. \tag{147}$$

Assuming bounded input norms ($\sum_m \alpha_{t,m}^2 \leqslant C$) and normalized singular values, we can relate the interference term to $I(k)$:

$$L \leqslant C \sum_{t=1}^{T} \sum_{j \neq t} \sum_{i'=1}^{k} \sum_{m} \sigma_{j,i'}^2 \left(v_{j,i'}^\top v_{t,m}\right)^2. \tag{148}$$

Since $\tilde{\sigma}_{j,i'}^2 \approx \sigma_{j,i'}^2 / \sum_{i''} \sigma_{j,i''}^2$, and assuming the singular values are of similar magnitude, we approximate:

$$\sum_{i'=1}^{k} \sum_{m} \sigma_{j,i'}^2 \left(v_{j,i'}^\top v_{t,m}\right)^2 \propto \sum_{i'=1}^{k} \sum_{m} \tilde{\sigma}_{j,i'}^2 \tilde{\sigma}_{t,m}^2 \left(v_{j,i'}^\top v_{t,m}\right)^2. \tag{149}$$

Thus, the total interference is bounded by:

$$L \leqslant O\left(\sum_{i=1}^{T} \sum_{j \neq i} \left\| \left(\tilde{\Sigma}_i \tilde{V}_i\right)^\top \left(\tilde{\Sigma}_j \tilde{V}_j\right) \right\|_F^2 \right) = O(I^2). \tag{150}$$

For centered task vectors, the zero-sum property $\sum_{t=1}^{T} \overline{\tau}_t = 0$ reduces the effective interference, as the contributions of other tasks cancel out around θ_{avg}. This is empirically supported by Figure 2 in (Choi et al., 2025), which shows that centered task vectors exhibit lower $I(k)$ across all ranks compared to original task vectors. The optimal rank $k \approx 8\%$ balances the trade-off between reconstruction accuracy, measured by the Reconstruction Error:

$$R(k) := \sum_{i=1}^{T} \|\theta_i - \theta_{\text{avg}} - \text{SVD}_k(\theta_i - \theta_{\text{avg}})\|_F^2, \tag{151}$$

and interference reduction, as $R(k)$ decreases sharply in the low-rank regime while $I(k)$ increases gradually (see Figure 3 in (Choi et al., 2025)). □

F BOUNDING $\|Bx\|$

Let r_A and r_B be the original ranks of A and B, $B = \sum_{i=1}^{r_B} \sigma_i^B u_i^B (v_i^B)^T$, $x = \sum_{j=1}^{r_A} \alpha_j v_j^A$, and $\{v_i^A\}_{i=1}^{r_A}$ and $\{v_i^B\}_{i=1}^{r_B}$ are orthonormal vectors,then we have

$$\|Bx\| = \left\| \sum_i \sigma_i^B u_i^B (v_i^B)^T \sum_j \alpha_j v_j^A \right\| \tag{152}$$

$$\leqslant \sum_i \|u_i^B\| \cdot \left| \sum_j \sigma_i^B \alpha_j (v_i^B)^T v_j^A \right| \tag{a}$$

$$\leqslant \sum_i \beta \cdot \left| \sum_j (v_i^B)^T v_j^A \right| \tag{b}$$

$$\leqslant \sum_{i=1}^{r_B} \beta \sqrt{r_A} \left(\sum_{j=1}^{r_A} \left((v_i^B)^T v_j^A \right)^2 \right)^{1/2} \tag{c}$$

$$= \sum_{i=1}^{r_B} \beta \sqrt{r_A} \left(\sum_{j=1}^{r_A} \langle v_i^B, v_j^A \rangle^2 \right)^{1/2}, \tag{d}$$

where $\beta = \max_{i,j} |\sigma_i^B \alpha_j|$, and inequality (c) uses Cauchy-Schwarz inequality. Then we show that

$$1 = \|v_i^B\|^2 \tag{153}$$

$$= \left\| \sum_{j=1}^{r_A} \langle v_i^B, v_j^A \rangle v_j^A + v_i^{B \perp A} \right\|^2 \tag{e}$$

$$= \sum_{j=1}^{r_A} \|\langle v_i^B, v_j^A \rangle v_j^A\|^2 + \|v_i^{B \perp A}\|^2 \tag{f}$$

$$= \sum_{j=1}^{r_A} \langle v_i^B, v_j^A \rangle^2 + \|v_i^{B \perp A}\|^2 \tag{g}$$

$$\geqslant \sum_{j=1}^{r_A} \langle v_i^B, v_j^A \rangle^2, \tag{h}$$

where equation (e) expresses v_i^B by $\{v_i^A\}_{i=1}^{r_A}$, and $v_i^{B \perp A}$ denotes the part of v_i^B that is orthogonal to the span of $\{v_i^A\}_{i=1}^{r_A}$. Equation (f) follows Pythagorean identity since $v_1^A, v_2^A, \ldots, v_{r_A}^A, v_i^{B \perp A}$ are pairwise-orthogonal vectors. Finally, with Equation (d) and (h), we have

$$\|Bx\| \leqslant r_B \beta \sqrt{r_A}. \tag{154}$$

G THEORETICAL ANALYSIS FOR EMR-MERGING

The goal is to minimize the average ℓ^2 distance between the unified model θ_{uni} and each individual model θ_i:

$$\text{Dis} = \frac{1}{T} \sum_{i=1}^{T} \|\theta_i - \theta_{\text{uni}}\|^2 = \frac{1}{T} \sum_{i=1}^{T} \|\tau_i - \tau_{\text{uni}}\|^2. \tag{155}$$

G.1 Effectiveness of Masks

After applying masks $M_i = (\tau_i \odot \tau_{\text{uni}} > 0)$, the distance becomes:

$$\text{Dis}_M = \frac{1}{N}\sum_{i=1}^{N} \|\tau_i - M_i \odot \tau_{\text{uni}}\|^2 \leqslant \text{Dis}. \tag{156}$$

Proof. For each parameter j and task i, if $(\tau_i^j \cdot \tau_{\text{uni}}^j) \leqslant 0$, then $M_i^j = 0$, so the term changes from $(\tau_i^j - \tau_{\text{uni}}^j)^2$ to $(\tau_i^j - 0)^2 = (\tau_i^j)^2$. Since signs conflict, $|\tau_i^j - \tau_{\text{uni}}^j| > \max(|\tau_i^j|, |\tau_{\text{uni}}^j|)$, so $(\tau_i^j - \tau_{\text{uni}}^j)^2 > (\tau_i^j)^2$, reducing the distance. If signs align, $M_i^j = 1$, and the term remains unchanged. Thus, $\text{Dis}_M \leqslant \text{Dis}$. □

G.2 Effectiveness of Rescalers

After applying rescalers $\lambda_i = \frac{\sum|\tau_i|}{\sum|M_i \odot \tau_{\text{uni}}|}$, the distance is:

$$\text{Dis}_{M,\lambda} = \frac{1}{N}\sum_{i=1}^{N} \|\tau_i - \lambda_i \cdot M_i \odot \tau_{\text{uni}}\|^2 \leqslant \text{Dis}_M. \tag{157}$$

Proof. Fix a task i, and let $v = M_i \odot \tau_{\text{uni}}$. Let $f(\lambda) = \|\tau_i - \lambda v\|^2$, then $\text{Dis}_{M,\lambda} = f(\lambda_i)$ and $\text{Dis}_M = f(1)$. The objective is to show that $f(\lambda_i) \leqslant f(1)$.

$$f(\lambda) = \|\tau_i - \lambda v\|^2 = (\tau_i - \lambda v)^\top(\tau_i - \lambda v) = \|\tau_i\|^2 - 2\lambda(\tau_i^\top v) + \lambda^2\|v\|^2. \tag{158}$$

Taking the derivative:

$$\frac{df}{d\lambda} = -2(\tau_i^\top v) + 2\lambda\|v\|^2 = 0, \tag{159}$$

so the optimal $\lambda^* = \frac{\tau_i^\top v}{\|v\|^2}$.

Let $S_i = \{j : M_i^j = 1\}$ (aligned parameters). Then $\tau_i^\top v = \sum_{j\in S_i} \tau_i^j \tau_{\text{uni}}^j$, and since signs match, $\tau_i^j \tau_{\text{uni}}^j = |\tau_i^j||\tau_{\text{uni}}^j|$.

And we have

$$\lambda^* = \frac{\sum_{j\in S_i} |\tau_i^j||\tau_{\text{uni}}^j|}{\sum_{j\in S_i}(|\tau_{\text{uni}}^j|)^2} = \frac{\sum_{j=1}^{d} |\tau_i^j|}{\sum_{j\in S_i} |\tau_{\text{uni}}^j|} = \lambda_i. \tag{160}$$

Then $f(\lambda_i) \leqslant f(1)$.

Thus, averaging over tasks, $\text{Dis}_{M,\lambda} \leqslant \text{Dis}_M$, with empirical alignment ensuring the inequality. □

H Theoretical Analysis for DARE

We provide a theoretical analysis demonstrating that DARE preserves the expected output of linear transformations. This is crucial for understanding why DARE can sparsify delta parameters without significantly impacting performance, as it approximates the original embeddings.

Let $W, \Delta W \in \mathbb{R}^{m\times n}$ denote the pre-trained and delta weight matrices, respectively, and $b, \Delta b \in \mathbb{R}^m$ the corresponding biases. Given an input vector $x \in \mathbb{R}^n$, the original output embedding $h \in \mathbb{R}^m$ for the i-th dimension (where $1 \leqslant i \leqslant m$) is given by:

$$h_i = \sum_{j=1}^{n}(w_{ij} + \Delta w_{ij})x_j + (b_i + \Delta b_i), \tag{161}$$

where w_{ij} and Δw_{ij} are elements of W and ΔW, and similarly for biases.

The expectation of h_i is:

$$
\begin{aligned}
\mathbb{E}[h_i] &= \mathbb{E}\left[\sum_{j=1}^{n}(w_{ij}+\Delta w_{ij})x_j + (b_i+\Delta b_i)\right] \\
&= \sum_{j=1}^{n} x_j\mathbb{E}[w_{ij}] + \mathbb{E}[b_i] + \sum_{j=1}^{n} x_j\mathbb{E}[\Delta w_{ij}] + \mathbb{E}[\Delta b_i] \\
&= \sum_{j=1}^{n} w_{ij}x_j + b_i + \sum_{j=1}^{n}\Delta w_{ij}x_j + \Delta b_i \\
&= h_i^{\text{PRE}} + \Delta h_i,
\end{aligned}
$$

where $h_i^{\text{PRE}} = \sum_{j=1}^{n} w_{ij}x_j + b_i$ is the pre-trained contribution, and $\Delta h_i = \sum_{j=1}^{n}\Delta w_{ij}x_j + \Delta b_i$ is the delta contribution.

Applying DARE: Delta parameters are randomly dropped with probability p, and the remaining are rescaled by γ. The modified delta parameters become $\Delta\hat{W} \in \mathbb{R}^{m\times n}$ and $\Delta\hat{b} \in \mathbb{R}^{m}$. The modified output $\hat{h}_i$ is:

$$
\hat{h}_i = \sum_{j=1}^{n}(w_{ij}+\Delta\hat{w}_{ij})x_j + (b_i+\Delta\hat{b}_i). \tag{162}
$$

The expectation is:

$$
\begin{aligned}
\mathbb{E}[\hat{h}_i] &= \sum_{j=1}^{n} x_j\mathbb{E}[w_{ij}] + \mathbb{E}[b_i] + \sum_{j=1}^{n} x_j\mathbb{E}[\Delta\hat{w}_{ij}] + \mathbb{E}[\Delta\hat{b}_i] \\
&= h_i^{\text{PRE}} + \sum_{j=1}^{n} x_j\left[(1-p)\cdot\gamma\cdot\Delta w_{ij} + p\cdot 0\right] + (1-p)\cdot\gamma\cdot\Delta b_i + p\cdot 0 \\
&= h_i^{\text{PRE}} + (1-p)\cdot\gamma\cdot\left(\sum_{j=1}^{n}\Delta w_{ij}x_j + \Delta b_i\right) \\
&= h_i^{\text{PRE}} + (1-p)\cdot\gamma\cdot\Delta h_i.
\end{aligned}
$$

By setting $\gamma = 1/(1-p)$, we obtain:

$$
\mathbb{E}[\hat{h}_i] = h_i^{\text{PRE}} + \Delta h_i = \mathbb{E}[h_i]. \tag{163}
$$

Thus, DARE ensures that the expected outputs before and after its application are identical, approximating the original embeddings and preserving model performance despite sparsification.

This analysis assumes deterministic inputs and focuses on expectations over the random dropping process. In practice, the effectiveness of DARE increases with model size, allowing higher drop rates p (e.g., up to 0.99 for large models) while maintaining performance.

I Derivation for TALL-Mask

This short section shows the derivation of the mask criterion in more detail:

$$\begin{aligned}
\mathbf{m}_t^* &= \underset{\mathbf{m}_t \in \{0,1\}^d}{\operatorname{argmin}} \left(\|\mathbf{m}_t \circ \tau_{\mathrm{MTL}} - \tau_t\|_1 \right) \\
&= \underset{\mathbf{m}_t \in \{0,1\}^d}{\operatorname{argmin}} \sum_{n=1}^{d} \left| m_t^{(n)} \cdot \tau_{\mathrm{MTL}}^{(n)} - \tau_t^{(n)} \right| \\
\Longrightarrow m_t^{(n)*} &= \underset{m_t^{(n)} \in \{0,1\}}{\operatorname{argmin}} \left| m_t^{(n)} \cdot \tau_{\mathrm{MTL}}^{(n)} - \tau_t^{(n)} \right| \\
&= \begin{cases} 1 & \text{if } \left|\tau_t^{(n)}\right| \geqslant \left| \tau_{\mathrm{MTL}}^{(n)} - \tau_t^{(n)} \right| \\ 0 & \text{otherwise} \end{cases} \\
&= \mathbb{1}\left\{ \left|\tau_t^{(n)}\right| \geqslant \left| \tau_{\mathrm{MTL}}^{(n)} - \tau_t^{(n)} \right| \right\}
\end{aligned} \tag{164}$$

The superscript (n) represents the n-th element or the n-th sub-problem. Aggregating over all sub-problems yields that the optimal mask is given by:

$$\mathbf{m}_t^* = \mathbb{1}\{|\tau_t| \geqslant |\tau_{\mathrm{MTL}} - \tau_t|\} \tag{165}$$

J Derivation for the Complete Formulation of Concrete Subspace Learning

J.1 Gumbel-Max Trick and Bernoulli Sampling

A binary mask can be sampled from a Bernoulli distribution Bernoulli$(p = \sigma(x))$, where σ is the sigmoid function and x are logits. However, discrete sampling is non-differentiable. The Gumbel-Max trick treats Bernoulli as a 2-class categorical distribution. Let p_0 and p_1 be unnormalized probabilities for values 0 and 1, with $\mathbb{P}(m = 1) = p_1/(p_0 + p_1) = \sigma(x)$.

The event $m = 1$ occurs if $g_1 + \log p_1 > g_0 + \log p_0$, where $g_0, g_1 \sim$ Gumbel$(0, 1)$ (independent standard Gumbel variables). Thus,

$$\mathbb{P}(m = 1) = \mathbb{P}(g_1 - g_0 + \log p_1 - \log p_0 > 0). \tag{166}$$

Since $g_1 - g_0 \sim$ Logistic$(0, 1)$, this can be reparameterized as $g_1 - g_0 = \log u - \log(1 - u)$ for $u \sim$ Uniform$(0, 1)$:

$$\mathbb{P}(m = 1) = \mathbb{P}\left(\log \frac{u}{1-u} + \log \frac{\sigma(x)}{1-\sigma(x)} > 0 \right), \quad u \sim \text{Uniform}(0, 1). \tag{167}$$

J.1.1 Concrete Distribution Relaxation

The Concrete distribution relaxes the discrete Bernoulli to a continuous $[0, 1]$ approximation for differentiability. Introduce temperature $\lambda > 0$ to soften the decision:

$$\hat{m} = \sigma\left(\left(\log \frac{u}{1-u} + \log \frac{\sigma(x)}{1-\sigma(x)} \right) / \lambda \right). \tag{168}$$

As $\lambda \to 0$, $\hat{m}$ approaches a Bernoulli sample (step function); higher λ smooths it. This enables gradient backpropagation through sampling.

J.2 Bi-Level Optimization and Meta-Learning

The bi-level problem in Eq. 44 is solved via meta-learning. Initialize mask logits x to zeros. In the inner loop (for fusion parameters w):

- Mask and rescale task vectors:

$$\begin{aligned}
V' &= \{\tau_i \circ m\}_{i=1}^T, \\
V'' &= \{\tau_i' / \mathbb{E}[m]\}_{i=1}^T.
\end{aligned} \tag{169}$$

- Merge: $\theta_M = \mathcal{A}(\theta_0, V''; w)$.
- If w is optimizable, update on unlabeled batches D_i from tasks s_i using loss L_i (e.g., entropy): $w' = w - \alpha \nabla_w (\sum_i L_i(f(\theta), D_i))$, then re-merge θ.

In the outer loop (for logits x): Update $x = x - \beta \nabla_x (\sum_i L_i(f(\theta), D_i))$ on unlabeled data until convergence. This finds a subspace beneficial across tasks.

K Derivation for the Complete Formulation of PCB-Merging

K.1 Intra-Balancing and Inter-Balancing Derivation

Intra-balancing derives from emphasizing parameter magnitudes within a task while suppressing redundancies, scaled by task count T to intensify competition:

$$\beta_{\text{intra},i,d} = \frac{\exp(T \cdot \text{Norm}(\tau_{i,d}^2))}{\sum_{k=1}^{d} \exp(T \cdot \text{Norm}(\tau_{i,k}^2))}, \tag{170}$$

where Norm typically min-max normalizes to [0,1]. This softmax amplifies high-magnitude parameters, deemed critical.

Inter-balancing captures cross-task correlations by averaging softened similarities:

$$\beta_{\text{inter},i,d} = \sum_{j=1}^{n} \frac{\exp(\text{Norm}(\tau_{i,d} \cdot \tau_{j,d}))}{\sum_{k=1}^{d} \exp(\text{Norm}(\tau_{i,k} \cdot \tau_{j,k}))}. \tag{171}$$

The element-wise product $\beta_i = \beta_{\text{intra},i} \odot \beta_{\text{inter},i}$ integrates both, scoring each parameter's overall importance.

K.2 Mask Construction and Drop-Rescale

The mask thresholds β_i at the $(1-r)$-quantile of its sorted values, dropping the bottom $r \times 100\%$ parameters. The rescaled merge in Eq. 49 normalizes by the sum of importance scores, ensuring balanced contributions from retained parameters.

K.3 Evolutionary Coefficient Search

For task-specific λ_i, CMA-ES optimizes $\max_{\{\lambda_i\}} \text{ValAcc}(\theta_M)$, where $\theta_M = \theta_0 + \sum_i (\hat{\beta}_i \odot \lambda_i \tau_i) / \sum_i \hat{\beta}_i$, and ValAcc is validation accuracy. CMA-ES samples from a multivariate normal $\mathcal{N}(\mu, \Sigma)$, updates mean μ and covariance Σ iteratively based on top performers, converging in generations (e.g., population size 20, iterations 50). This exploits search space structure for efficient hyperparameter tuning without gradients.

L Derivations for SMILE

L.1 Subspace Basis via Orthonormal Vectors (Theorem 1 from (Tang et al., 2024a))

Theorem 3. *Given two sets of orthonormal vectors $\{u_i\}_{i=1}^{p} \subset \mathbb{R}^m$ and $\{v_i\}_{i=1}^{q} \subset \mathbb{R}^n$ (with $1 \leqslant p \leqslant m$ and $1 \leqslant q \leqslant n$), the set of matrices $\{u_i v_j^\top\}_{i=1,j=1}^{p,q}$ forms an orthonormal basis for a subspace of $\mathbb{R}^{m \times n}$ with dimension pq.*

Proof. Let $x_{ij} = u_i v_j^\top$. The Frobenius inner product is

$$\langle x_{ab}, x_{cd} \rangle = \text{tr}(u_a v_b^\top v_d u_c^\top) \tag{172}$$

Orthonormality follows from cases:

(1) $a = c, b = d$ yields 1;

(2) $b \neq d$ yields 0;

(3) $b = d, a \neq c$ yields 0. □

Linear independence: Assume $\sum_{i=1}^{p}\sum_{j=1}^{q}\alpha_{ij}x_{ij} = 0$. Inner product with x_{ab} yields $\alpha_{ab} = 0$ for all a, b, contradicting non-zero α. Thus, it forms a basis of dimension pq.

Remarks: Any matrix $A \in \mathbb{R}^{m\times n}$ expands as

$$A = \sum_{i=1}^{m}\sum_{j=1}^{n}\langle A, u_i v_j^\top\rangle u_i v_j^\top. \tag{173}$$

Subsets of $\{u_i\}, \{v_i\}$ span subspaces of dimension $|U||V|$.

L.2 Fine-Tuning Decomposition via SVD

For pre-trained weight $W_0 = U^{1:r}\Sigma^{1:r}(V^{1:r})^\top$ (reduced SVD, rank r), for task i, fine-tuned $W_i = W + \Delta W_i$. Output $y = W_i x + b_i$ decomposes as:

Pre-trained part:

$$\sum_{j=1}^{r}\sigma_j\langle x, v_j\rangle u_j + b. \tag{174}$$

Fine-tuned part:

$$\sum_{j=1}^{m}\sum_{k=1}^{n}\delta_{jk}\langle x, v_k\rangle u_j + \Delta b, \tag{175}$$

where $\delta_{jk} = \langle \Delta W, u_j v_k^\top\rangle = (U^\top \Delta W V)_{jk}$.

Subspace partitioning: Zone I (top r_{half} singular values, 50% cumulative sum), Zone II (remaining to r), Zone III (beyond r).

L.3 Parameter Interference Optimization (Full Formulation)

For merged $W_M = \sum_{t=1}^{T}\lambda_i W_i$, output is

$$y_M = \text{pre-trained} + \sum_{j=1}^{m}\sum_{k=1}^{n}\left(\sum_{t=1}^{T}\lambda_t\delta_{t,jk}\right)\langle x, v_k\rangle u_j + \sum_{t=1}^{T}\lambda_t\Delta b_t. \tag{176}$$

Minimize:

$$\min_{\lambda_i}\|y_M - y_i\|_2^2 \leqslant \left\|\sum_{j=1}^{m}\sum_{k=1}^{n}\left(\sum_{t=1}^{T}\lambda_t\delta_{t,jk} - \delta_{i,jk}\right)\langle x_i, v_k\rangle u_j\right\|_2^2 + \left\|\sum_{t=1}^{T}\lambda_t\Delta b_t - \Delta b_i\right\|_2^2, \tag{177}$$

which is obtained through the triangle inequality.

Bias solution: $\lambda_i = (\Delta B^\top \Delta B)^{-1}\Delta B \Delta b_i$ (task-dependent; fix $\Delta b_i = 0$ to simplify). Weight interference mitigated by dimension expansion.

L.4 Low-Rank Approximation Optimality (Theorem 2 from (Tang et al., 2024a))

Theorem 4. *Given $W \in \mathbb{R}^{m\times n}$, its rank-$k$ approximation $W_k = U_k\Sigma_k V_k^\top$ minimizes $\|W - W_k\|_F$.*

L.5 SMILE Output (Full Equation)

$$y = (Wx + b) + \sum_{i=1}^{T}\lambda_i\left(U_i^{1:k}\Sigma_i^{1:k}(V_i^{1:k})^\top x + \Delta b_i\right), \tag{178}$$

where $\lambda_i = p_i$ if $p_i \in \text{TopK}(\{p_j\}_{j=1}^{T}, K)$, else 0; $p_i = \text{softmax}_i(\|V_{k_{\text{gate}}}^{(i)\top}x\|_2)$.

Complexity: $O(T(mk + n(k + k_{\text{gate}})))$ additional parameters; activates $nTk_{\text{gate}} + K(mk + nk + m)$ per token.

M Derivations for FREE-Merging

M.1 No Free Lunch Theorem for Model Merging (Theorem 5.1 from (Zheng & Wang, 2025))

Theorem 5. *Merged model θ_M cannot simultaneously retain the capabilities of $\{\theta_k\}_{k=1}^K$ without introducing additional information.*

Proof. In this section, we provide a proof for Theorem 5. We begin by introducing the notation used in this proof.

1. **Notations and Preliminaries**
   To illustrate the proof, let's consider two fine-tuned models with parameters θ_A and θ_B. For clarity, we focus on a single-layer MLP (without activation functions), though we will explain at the end how our proof generalizes to all cases. The merged result is given by:
   $$\theta_M = \lambda_1 G(\theta_A) + \lambda_2 G(\theta_B), \tag{179}$$
   where $G(\cdot)$ represents a transformation operation applied to the parameters. Assume that θ_A corresponds to task input X_A, and θ_B corresponds to task input X_B.

   The goal is to prove that, without additional storage or extra training for θ_A and θ_B, simply storing θ_M cannot perfectly retain the capabilities of both θ_A and θ_B.

2. **Details of Proof**:
   Next, we will analyze the cases separately.

   - **Without transformations**:
     Assume that we do not apply transformations to the output of the merged network. The output of the model f_θ can be expressed as:
     $$\text{output}_f = \theta x. \tag{180}$$
     Suppose θ_M can perfectly retain the capabilities of both θ_A and θ_B without introducing new knowledge. That is, for $\forall x_A \in X_A, x_B \in X_B$, we have:
     $$\theta_M x_A = \theta_A x_A, \quad \theta_M x_B = \theta_B x_B. \tag{181}$$
     Then we have:
     $$[\lambda_1 G(\theta_A) + \lambda_2 G(\theta_B)] x_A = \theta_A x_A, \tag{182}$$
     $$[\lambda_1 G(\theta_A) + \lambda_2 G(\theta_B)] x_B = \theta_B x_B. \tag{183}$$
     Assume that $\exists x_M = \mu_1 x_A + \mu_2 x_B \in X_A$, where $\mu_1, \mu_2 \in \mathbb{N}$. Substituting, we can obtain:
     $$\begin{aligned} \theta_M x_M &= \theta_A x_M \\ \Rightarrow \theta_M (x_A - x_M) &= \theta_A (x_A - x_M) \\ \Rightarrow \theta_M [(1-\mu_1) x_A - \mu_2 x_B] &= \theta_A [(1-\mu_1) x_A - \mu_2 x_B] \\ \Rightarrow \mu_2 \theta_M x_B &= \mu_2 \theta_A x_B \\ \Rightarrow \mu_2 \theta_B x_B &= \mu_2 \theta_A x_B. \end{aligned}$$
     Due to:
     $$\mu_2 \theta_M x_B = \mu_2 \theta_B x_B, \tag{184}$$
     thus, if Eq. (14) and Eq. (15) hold for $\forall x_A \in X_A, x_B \in X_B$, then $\mu_2 = 0$. Therefore, we have:
     $$\forall \mu_2 \neq 0, \nexists x_M = \mu_1 x_A + \mu_2 x_B \in X_A. \tag{185}$$
     Eq. (16) indicates that X_A is not continuous. However, we usually consider the input space of the model to be continuous. Even if it is discontinuous, we expect the model to be robust to slight perturbations in the input.
     Thus, the assumption that θ_M can perfectly retain θ_A and θ_B without extra data or training does not hold in this case.

- **With transformations**:
  Assuming we apply certain transformations to the output of the merged network, the output of the model f_θ can be expressed as:

$$\text{output}_f = h(\theta x), \tag{186}$$

  where $h(\cdot)$ represents a transformation applied to the output.
  By the universal approximation theorem, $h(\cdot)$ can be approximated by a two-layer MLP with sigmoid activation. Then, assuming θ_M can perfectly retain the capabilities of both θ_A and θ_B, then for $\forall x_A \in X_A, x_B \in X_B$, we have:

$$W_2[\text{sigmoid}(W_1\theta_M x_A)] = \theta_A x_A, \tag{187}$$

$$W_2[\text{sigmoid}(W_1\theta_M x_B)] = \theta_B x_B. \tag{188}$$

  Substituting, we obtain:

$$\frac{W_2}{1+e^{-W_1\theta_M x_A}} = \theta_A x_A, \tag{189}$$

$$\frac{W_2}{1+e^{-W_1\theta_M x_B}} = \theta_B x_B. \tag{190}$$

  Then we have:

$$W_2 = \theta_A x_A(1+e^{-W_1\theta_M x_A}) = \theta_B x_B(1+e^{-W_1\theta_M x_B}). \tag{191}$$

  Thus, we can obtain the following equation:

$$\theta_A x_A(e^{W_1\theta_M x_A}+1)e^{W_1\theta_M x_B} = \theta_B x_B(e^{W_1\theta_M x_B}+1)e^{W_1\theta_M x_A}. \tag{192}$$

  Therefore,

$$(\theta_A x_A - \theta_A x_B)(e^{W_1\theta_M x_B}e^{W_1\theta_M x_A} + e^{W_1\theta_M x_B}) = \theta_B x_B(e^{W_1\theta_M x_B} + e^{W_1\theta_M x_A}). \tag{193}$$

  Based on Eq. (22) and Eq. (26), since they hold for $\forall x_A \in X_A, x_B \in X_B$, at least one of W_1 or W_2 can be expressed as $\Phi(\theta_A, \theta_B)$, where Φ represents a function.
  In other words, only by additionally storing part of the information from θ_A or θ_B and processing θ_M at the output can θ_M perfectly retain the capabilities of both θ_A and θ_B. Therefore, the assumption that θ_M can perfectly retain both θ_A and θ_B does not hold in this case.
  Next, we briefly outline the rationale for extending this result to all layers of a neural network. Our assumptions involve only a single layer of the neural network, which already cannot perfectly retain capabilities without introducing new data. Therefore, it is even less feasible for the entire network to achieve this. Thus, this extension is reasonable.

In summary, the proposition is proven.

□

M.2 Expert Rescaling Derivation

The rescaling factor μ_i in Eq. 87 ensures output consistency post-extraction. Starting from the goal of maintaining

$$\mathbb{E}(e(v_i)) \approx \lambda_i \mathbb{E}(v_i) \tag{194}$$

while compressing to top-d parameters, we derive

$$\mu_i = -\frac{\mathbb{E}(M(v_i, d)) \cdot \log(d)}{\lambda_i \cdot \mathbb{E}(v_i)} \tag{195}$$

by balancing magnitude selection and logarithmic compression to minimize deviation, preserving task expertise with minimal parameters.