Model Merging in LLMs, MLLMs, and Beyond: Methods, Theories, Applications and Opportunities

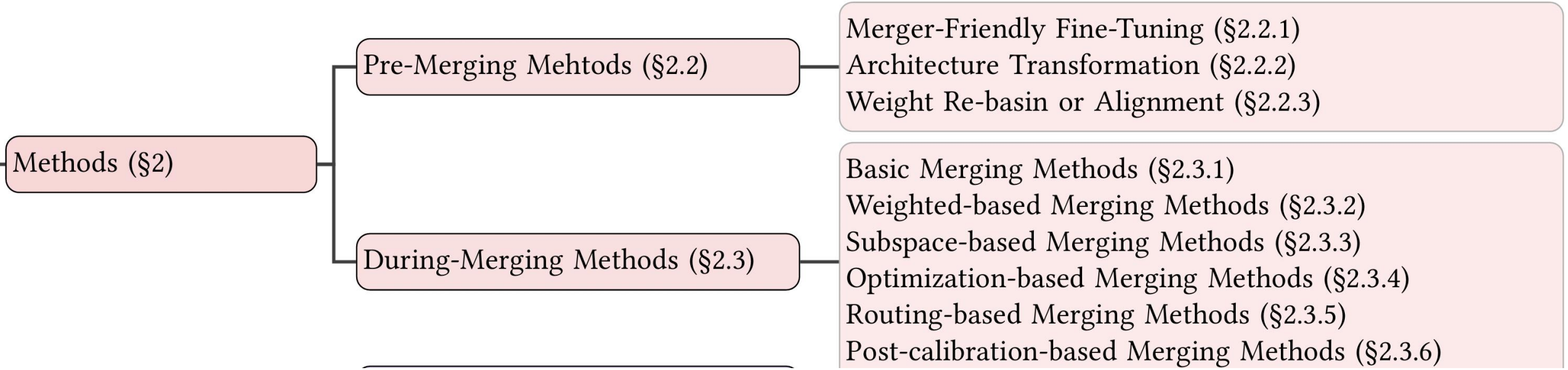
————恒然

Content

- How are existing model merging methods classified?
- Which applications can benefit from model merging?

How are existing model merging methods classified?

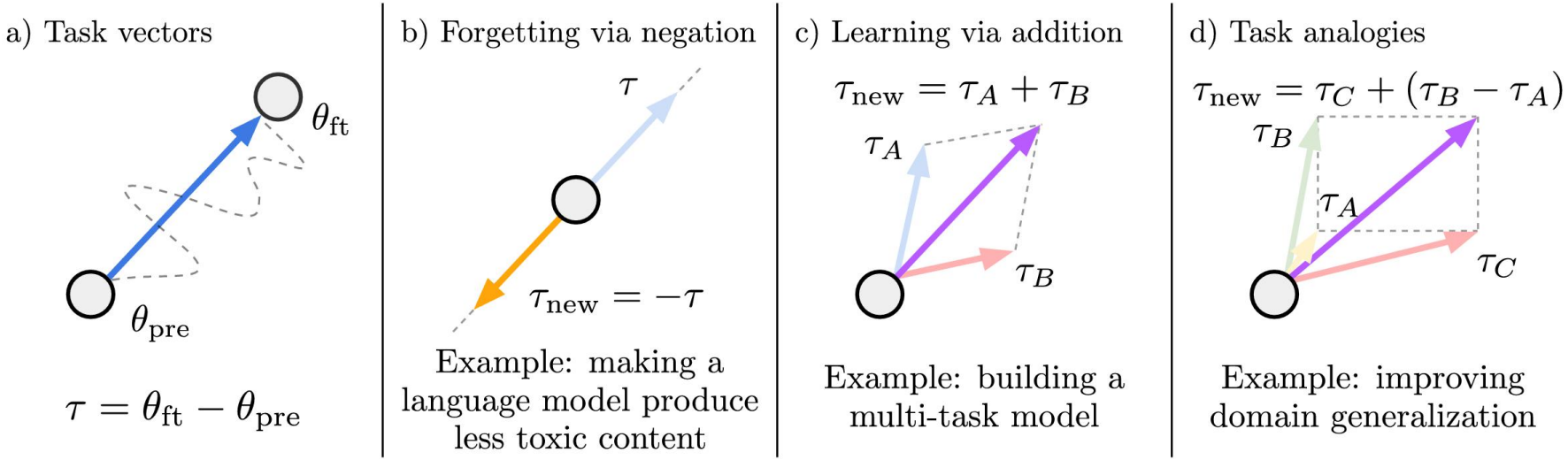
- **Pre-merging methods**: Pre-merging methods aim to create better conditions for merging.
- **During-merging methods**: Focus on designing sophisticated techniques to merge multiple models into one



Merger-Friendly Fine-Tuning

- Linearization Fine-tuning
- Sharpness-Aware Fine-Tuning
- Subspace Fine-Tuning

Editing models with task arithmetic (TA)



Figu … dels.
(a) … ts of
the … e on
the … ther
imp … n 4).
(d) … two
diff… only
vect…

Domain generalization. For many target tasks, gathering unlabeled data is easier and cheaper than collecting human annotations. When labeled data for a *target* task is not available, we can use task analogies to improve accuracy on the target task, using an *auxiliary* task for which there is labeled data and an unsupervised learning objective. For example, consider the target task of sentiment analysis using data from Yelp [102]. Using task analogies, we can construct a task vector $\hat{\tau}_{\text{yelp; sent}} = \tau_{\text{amazon; sent}} + (\tau_{\text{yelp; lm}} - \tau_{\text{amazon; lm}})$, where $\tau_{\text{amazon; sent}}$ is obtained by fine-tuning on labeled data from an auxiliary task (sentiment analysis using data from Amazon; McAuley & Leskovec [65]), and $\tau_{\text{yelp; lm}}$ and $\tau_{\text{amazon; lm}}$ are task vectors obtained via (unsupervised) language modeling on the inputs from both datasets.

Ilharco, G., Ribeiro, M. T., Wortsman, M., Gururangan, S., Schmidt, L., Hajishirzi, H., & Farhadi, A. (2022). Editing models with task arithmetic. *ICLR2023*.

Table 1: **Forgetting image classification tasks via negation**. Results are shown for CLIP models, reporting average accuracy (%) on the eight target tasks we wish to forget (Cars, DTD, EuroSAT, GTSRB, MNIST, RESISC45, SUN397 and SVHN), and the control task (ImageNet). Negating task vectors reduce the accuracy of a pre-trained ViT-L/14 by 45.8 percentage points on the target tasks, with little loss on the control task. Additional details and results are shown in Appendix B.

Method	ViT-B/32		ViT-B/16		ViT-L/14	
	Target (↓)	Control (↑)	Target (↓)	Control (↑)	Target (↓)	Control (↑)
Pre-trained	48.3	63.4	55.2	68.3	64.8	75.5
Fine-tuned	90.2	48.2	92.5	58.3	94.0	72.6
Gradient ascent	2.73	0.25	1.93	0.68	3.93	16.3
Random vector	45.7	61.5	53.1	66.0	60.9	72.9
Negative task vector	24.0	60.9	21.3	65.4	19.0	72.9

Table 2: **Making language models less toxic with negative task vectors.** Results are shown for the GPT-2 Large model. Negative task vectors decrease the amount of toxic generations by 6×, while resulting in a model with comparable perplexity on a control task (WikiText-103). Additional details and results are shown in Appendix C.

Method	% toxic generations (↓)	Avg. toxicity score (↓)	WikiText-103 perplexity (↓)
Pre-trained	4.8	0.06	16.4
Fine-tuned	57	0.56	16.6
Gradient ascent	0.0	0.45	$>10^{10}$
Fine-tuned on non-toxic	1.8	0.03	17.2
Random vector	4.8	0.06	16.4
Negative task vector	0.8	0.01	16.9

Non-toxic samples from Civil Comments (toxicity scores smaller than 0.2).

Pre-Merging Methods

- **Merger-Friendly Fine-Tuning**
- Architecture Transformation
- Weight Re-basin or Alignment

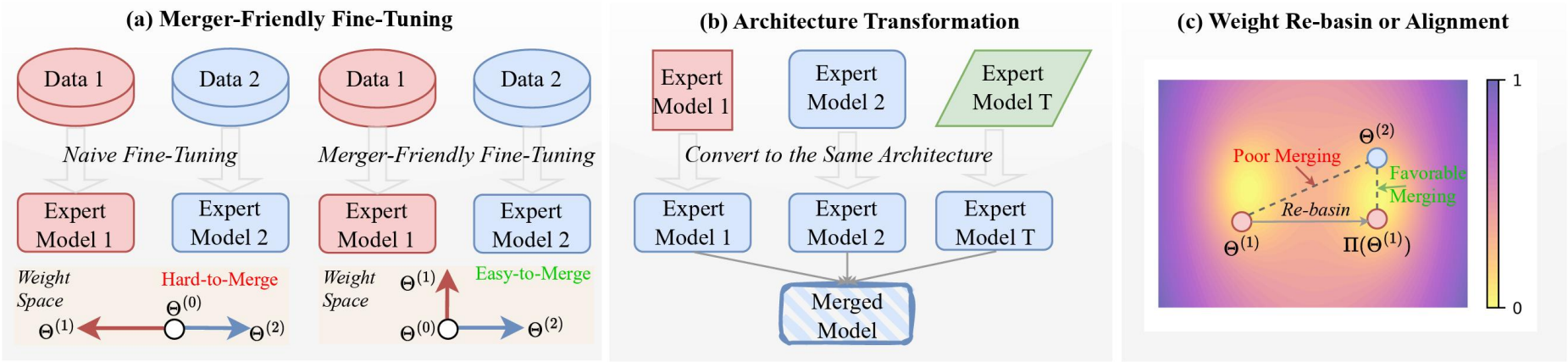


Fig. 3. (a) Illustration of merge-friendly fine-tuning: unlike standard fine-tuning, which yields models that are hard to merge, merge-friendly fine-tuning produces models that are easier to merge. (b) Illustration of an architectural transformation that converts multiple heterogeneous models into a homogeneous architecture, enabling direct parameter-level merging. (c) Illustration of weight/parameter alignment: permuting the neural network model $\Theta^{(1)}$ so that it aligns with $\Theta^{(2)}$.

Merger-Friendly Fine-Tuning

- **Linearization Fine-tuning**
- Sharpness-Aware Fine-Tuning
- Subspace Fine-Tuning

Linearization Fine-tuning

- Weight Disentanglement
  - Different directions of the weight space correspond to functional changes in disjoint regions of the input space.

Property 3 (Weight disentanglement)**.** *A parametric function $f : \mathcal{X} \times \Theta \to \mathcal{Y}$ is weight disentangled with respect to a set of task vectors $\mathcal{T} = \{\boldsymbol{\tau}_t\}_{t\in[T]}$ and the corresponding supports $\mathcal{D} = \{\mathcal{D}_t\}_{t\in[T]}$ if*

$$f\left(\boldsymbol{x}; \boldsymbol{\theta}_0 + \sum_{t=1}^{T} \alpha_t \boldsymbol{\tau}_t\right) = \sum_{t=1}^{T} g_t(\boldsymbol{x}; \alpha_t \boldsymbol{\tau}_t) + g_0(\boldsymbol{x}), \tag{4}$$

where $g_t(\boldsymbol{x}; \alpha_t \boldsymbol{\tau}_t) = \mathbf{0}$ for $\boldsymbol{x} \notin \mathcal{D}_t$ and $t = 1, \ldots, T$, and $g_0(\boldsymbol{x}) = 0$ for $\boldsymbol{x} \in \bigcup_{t\in[T]} \mathcal{D}_t$.

That is, we apply the fine-tuned task vectors $\boldsymbol{\tau} = \boldsymbol{\theta}^\star - \boldsymbol{\theta}_0$ to the linear approximation of f at $\boldsymbol{\theta}_0$, *i.e.*,

$$f_{\text{lin}}(\boldsymbol{x}; \boldsymbol{\theta}_0 + \boldsymbol{\tau}) = f(\boldsymbol{x}; \boldsymbol{\theta}_0) + \boldsymbol{\tau}^\top \nabla_{\boldsymbol{\theta}} f(\boldsymbol{x}; \boldsymbol{\theta}_0), \tag{3}$$

and we check whether $f_{\text{lin}}(\cdot\,; \boldsymbol{\theta}^\star)$ performs similarly to $f(\cdot\,; \boldsymbol{\theta}^\star)$[2].

Ortiz-Jimenez, G., Favero, A., & Frossard, P. (2023). Task arithmetic in the tangent space: Improved editing of pre-trained models. *Advances in Neural Information Processing Systems*, *36*, 66727-66754.

Sharpness-Aware Fine-Tuning

- Sharpness—aware minimization to encourage a flatter task—specific loss landscape

In this work, we claim that we need to achieve both (1) *less performance gap between a merged model and each fine-tuned model (i.e., less parameter interference)* and (2) *generalization performance of each fine-tuned model* on each respective dataset. As such, we aim to design a new objective function for fine-tuning to achieve these two objectives:

$$\boldsymbol{\theta}_t = \arg\min_{\boldsymbol{\theta}} \underbrace{\mathcal{L}(\boldsymbol{\theta}_{\text{merge}}(\boldsymbol{\theta}); \mathcal{D}^{(t)}) - \mathcal{L}(\boldsymbol{\theta}; \mathcal{D}^{(t)})}_{\text{Objective (1)}} + \underbrace{\mathcal{L}(\boldsymbol{\theta}; \mathcal{D}^{(t)})}_{\text{Objective (2)}}, \tag{5}$$

$$\min_{\boldsymbol{\theta}} \mathcal{L}(\boldsymbol{\theta} + \hat{\boldsymbol{\epsilon}}; \mathcal{D}) \quad \text{where} \quad \hat{\boldsymbol{\epsilon}} = \rho \frac{\boldsymbol{\theta}^2 \nabla_{\boldsymbol{\theta}} \mathcal{L}(\boldsymbol{\theta}; \mathcal{D})}{\|\nabla_{\boldsymbol{\theta}} \mathcal{L}(\boldsymbol{\theta}; \mathcal{D})\|}. \tag{7}$$

Lee, Y., Jung, J., & Baik, S. (2025). Mitigating parameter interference in model merging via sharpness-aware fine-tuning. ICLR2025.

Subspace Fine-Tuning

- Unlike the full-parameter fine-tuning strategies mentioned above, some approaches fine-tune only a subset of parameters instead of the whole set.

- OSRM fine-tunes the model via LoRA in an orthogonal subspace, thereby mitigating unintended interference across tasks.

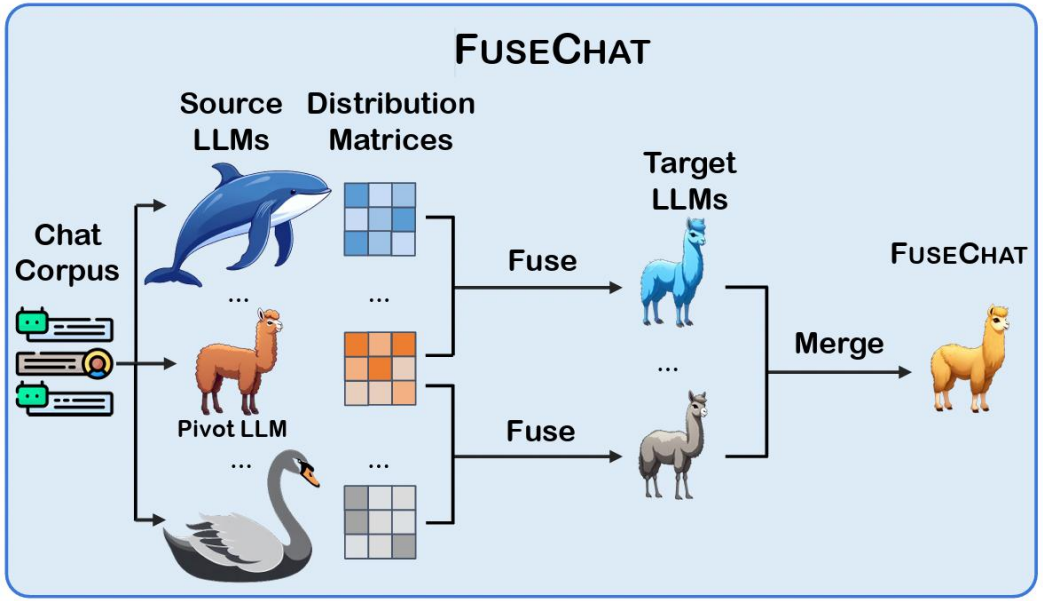
Zhang, H., & Zhou, J. (2025, July). Unraveling lora interference: Orthogonal subspaces for robust model merging. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)* (pp. 26459-26472).

Pre-Merging Methods

- Merger-Friendly Fine-Tuning
- **Architecture Transformation**
- Weight Re-basin or Alignment

Architecture Transformation

- FuseLLM and FusionChat propose to merge chat LLMs with diverse architectures and scales (e.g., NH2Mixtral-8x7B, NH2-Solar-10.7B, OpenChat-3.5-7B in their practical applications)
  - FusionChat first uses knowledge distillation to transform all the architectures to match that of OpenChat-3.5-7B, and then performs the merge operation



Fanqi Wan, Ziyi Yang, Longguang Zhong, Xiaojun Quan, Xinting Huang, and Wei Bi. 2024. FuseChat: Knowledge Fusion of Chat Models. arXiv preprint arXiv:2402.16107 (2024).

Pre-Merging Methods

- Merger-Friendly Fine-Tuning
- Architecture Transformation
- **Weight Re-basin or Alignment**

Weight Re-basin or Alignment

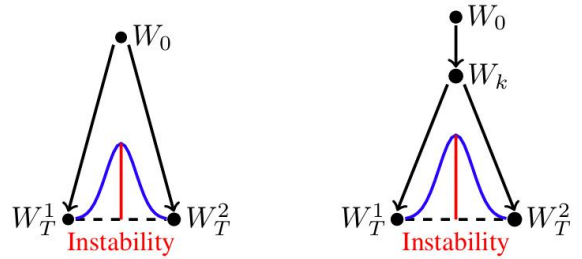


Figure 1. A diagram of instability analysis from step 0 (left) and step k (right) when comparing networks using linear interpolation.

- The linear mode connectivity (LMC) property of deep neural networks demonstrates that there is a connected path between multiple local minima of deep neural networks along which the loss remains nearly constant.

We use $\mathrm{Err}_{\mathcal{D}}(\boldsymbol{\theta})$ to denote the classification error of the network $f(\boldsymbol{\theta};\cdot)$ on the dataset $\mathcal{D}$.

Linear Mode Connectivity (LMC). We recall the notion of LMC in Definition 1.

Definition 1 (**Linear Mode Connectivity**). *Given a test dataset $\mathcal{D}$ and two modes*[3] *$\boldsymbol{\theta}_A$ and $\boldsymbol{\theta}_B$ such that $\mathrm{Err}_{\mathcal{D}}(\boldsymbol{\theta}_A) \approx \mathrm{Err}_{\mathcal{D}}(\boldsymbol{\theta}_B)$, we say $\boldsymbol{\theta}_A$ and $\boldsymbol{\theta}_B$ are linearly connected if they satisfy*

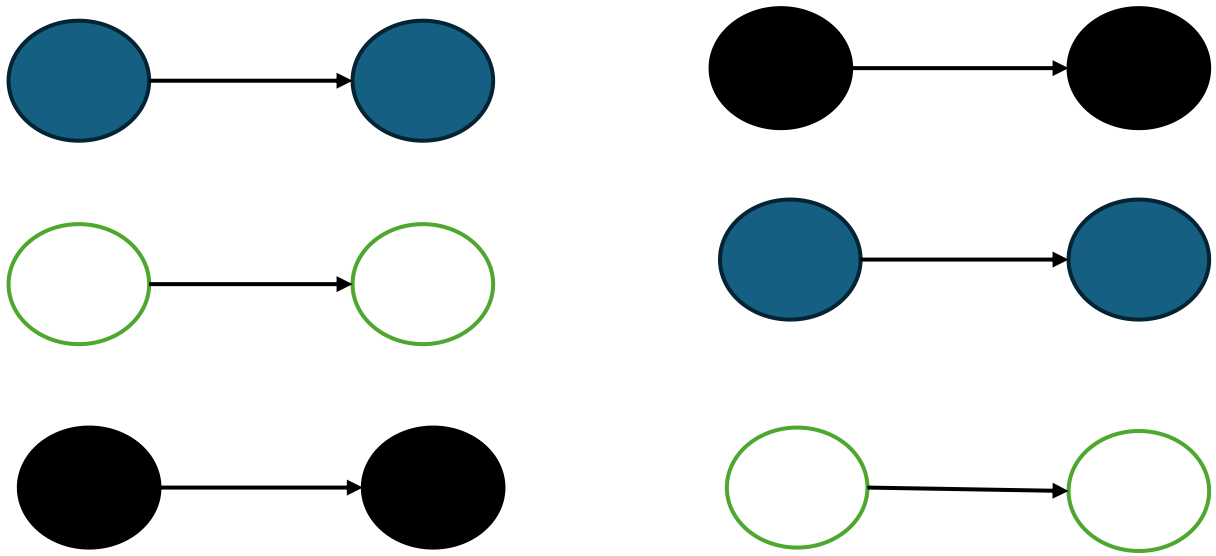
$$\mathrm{Err}_{\mathcal{D}}(\alpha\boldsymbol{\theta}_A + (1-\alpha)\boldsymbol{\theta}_B) \approx \mathrm{Err}_{\mathcal{D}}(\boldsymbol{\theta}_A), \qquad \forall\alpha \in [0,1]. \tag{2}$$

As Definition 1 shows, $\boldsymbol{\theta}_A$ and $\boldsymbol{\theta}_B$ satisfy LMC if the error metric on the linear path connecting their weights is nearly constant. There are two known methods to obtain linearly connected modes, the *spawning method* [9, 7] and the *permutation method* [6, 1].

Zhou, Z., Yang, Y., Yang, X., Yan, J., & Hu, W. (2023). Going beyond linear mode connectivity: The layerwise linear feature connectivity. *Advances in neural information processing systems*, *36*, 60853-60877.
Frankle, J., Dziugaite, G. K., Roy, D., & Carbin, M. (2020, November). Linear mode connectivity and the lottery ticket hypothesis. In *International conference on machine learning* (pp. 3259-3269). PMLR.

How to achieve LMC

- Spawning: A network is randomly initialized, trained for a small number of epochs, and then spawned into two copies which continue to be independently trained using different SGD randomnesses.
- Permutation: Two networks are independently trained, and the neurons of one model are permuted to match the neurons of the other model while maintaining a functionally equivalent network.



1. LMC 的定义

对于两个神经网络权重 v 和 v'，定义它们的线性插值为：

$$v_\lambda = \lambda v + (1-\lambda)v', \quad \lambda \in [0,1]$$

LMC 要求：对于所有 $\lambda \in [0,1]$，插值模型 v_λ 的损失 $\mathcal{L}(v_\lambda)$ 不应显著高于原始模型损失的加权平均：

$$\mathcal{L}(v_\lambda) \lesssim \lambda \mathcal{L}(v) + (1-\lambda)\mathcal{L}(v')$$

2. ℓ_{LMC} 损失函数的设计

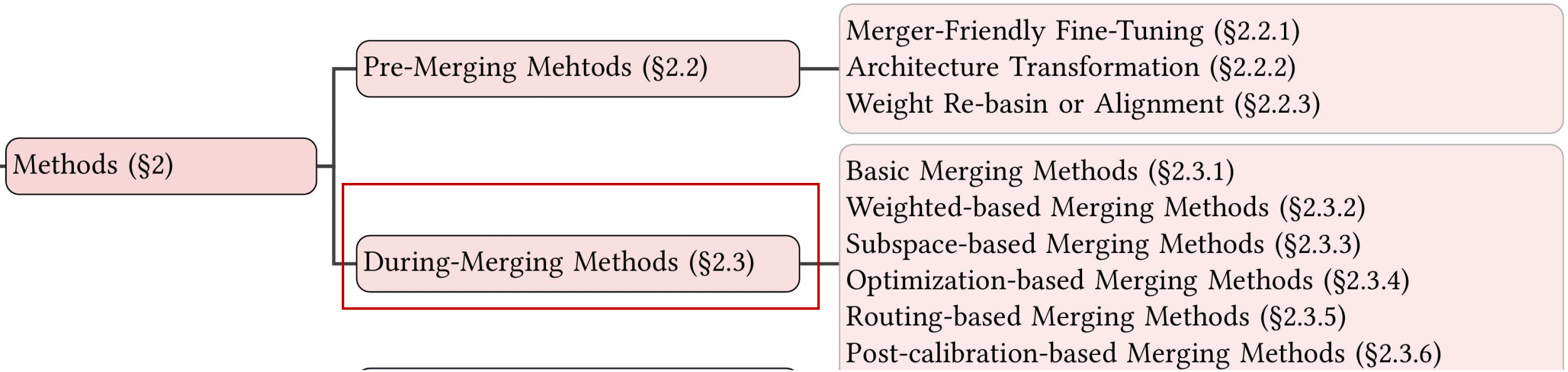
为了让 DEEP-ALIGN 学会输出一个 permutation g（即 $F(v,v')$），使得 v 和 $g_\# v'$ 之间具有良好的 LMC，作者引入了以下损失：

$$\ell_{\text{LMC}}(v,v',\theta) = \mathbb{E}_{\lambda\sim U(0,1)}\left[\mathcal{L}\big(\lambda v + (1-\lambda)\, F(v,v';\theta)_\# v'\big)\right]$$

其中：

- $F(v,v';\theta)$ 是 DEEP-ALIGN 模型预测的 permutation 序列；
- $F(v,v';\theta)_\# v'$ 表示用这个 permutation 重排 v' 后的权重；
- $\mathcal{L}$ 是原始任务损失（如分类交叉熵）；
- λ 在训练过程中随机采样。

Navon, A., Shamsian, A., Fetaya, E., Chechik, G., Dym, N., & Maron, H. (2023). Equivariant deep weight space alignment. PMLR2024


Methods (§2)
Pre-Merging Mehtods (§2.2)
Merger-Friendly Fine-Tuning (§2.2.1)
Architecture Transformation (§2.2.2)
Weight Re-basin or Alignment (§2.2.3)
During-Merging Methods (§2.3)
Basic Merging Methods (§2.3.1)
Weighted-based Merging Methods (§2.3.2)
Subspace-based Merging Methods (§2.3.3)
Optimization-based Merging Methods (§2.3.4)
Routing-based Merging Methods (§2.3.5)
Post-calibration-based Merging Methods (§2.3.6)

During Merging Methods

- Basic methods
- Weighted-based methods
- Subspace-based methods
- Optimization-based methods
- Routing-based methods
- Other merging methods

Basic methods

- One of the most straightforward approaches to model merging is to directly weighted average the parameters of multiple models.
- $\theta^{(merge)} = \frac{1}{T}\sum_{t=1}^{T}\theta^{(t)}$
- Task Arithmetic: $\theta^{(merge)} = \theta^{(0)} + \lambda\sum_{t=1}^{T}\tau_t$

Weighted-based methods

- Different models (or task vectors) represent different functions, and intuitively, different functions have varying degrees of importance.
- The goal of the weighted merging method is to find the optimal coefficients.

AdaMerging

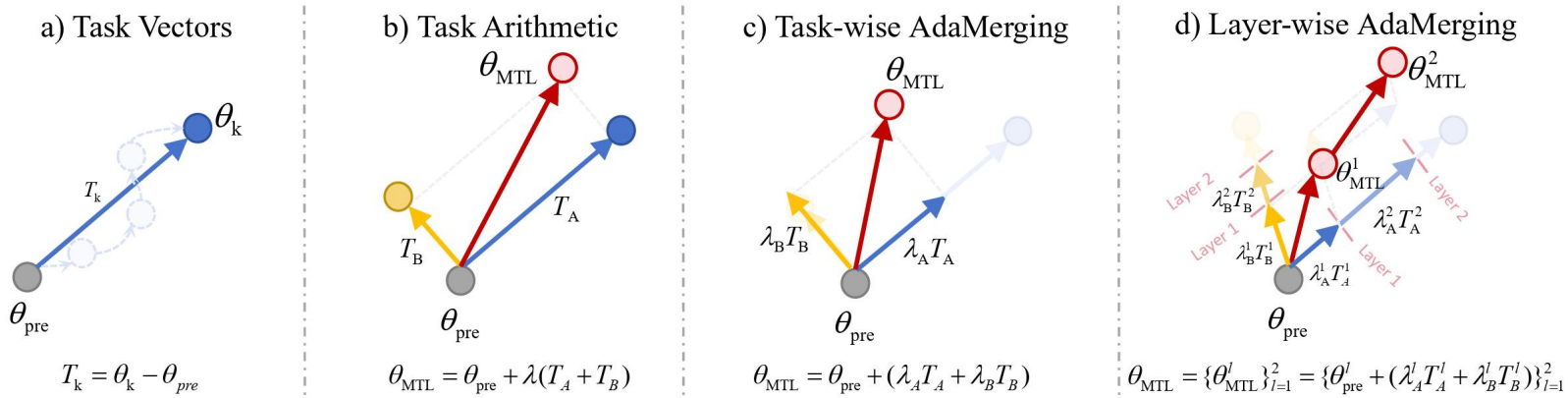


Figure 2: (a) Definition of "task vector", the task vector T_k is obtained by subtracting the pre-trained weights θ_{pre} from the model weights θ_k fine-tuned on the data of task k. (b) *Task Arithmetic* (Ilharco et al., 2023) for MTL, which assigns same merging coefficient λ to each task vector T_k ($k \in \{A, B\}$). (c) `Task-wise AdaMerging` for MTL, which learns a distinct merging coefficient λ_k to each task vector T_k ($k \in \{A, B\}$). (d) `Layer-wise AdaMerging` for MTL, which learns a distinct merging coefficient λ^l_k to each layer l ($l \in \{1, 2\}$) of the task vector T_k ($k \in \{A, B\}$).

$$\min_{\lambda_1,\lambda_2,\ldots,\lambda_K} \sum_{k=1}^{K} \sum_{x_i \in \mathcal{B}_k} H(f_{\theta_{MTL}}(x_i)) \text{ , where } \theta_{MTL} = \theta_{\text{pre}} + \sum_{k=1}^{K} \lambda_k T_k, \quad \text{H() is the Shannon entropy}$$

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

🔧 1. 定义可学习合并系数

- **任务级 AdaMerging**：为每个任务的“任务向量” $T_k = \theta_k - \theta_{\text{pre}}$ 分配一个可学习的系数 λ_k，合并后的模型为：

$$\theta_{\text{MTL}} = \theta_{\text{pre}} + \sum_{k=1}^{K} \lambda_k T_k$$

- **层级 AdaMerging**：为每个任务向量的每一层 T_k^l 分配一个独立的可学习系数 λ_k^l，合并后的模型为：

$$\theta_{\text{MTL}}^l = \theta_{\text{pre}}^l + \sum_{k=1}^{K} \lambda_k^l T_k^l$$

- **AdaMerging++**：在 Ties-Merging 的基础上进行相同的系数学习，即使用经过冲突处理的任务向量 $\Phi(T_k)$ 进行合并。

🧠 2. 无监督代理目标：熵最小化

- 由于无法访问原始训练数据，AdaMerging 使用**未标注的多任务测试数据**作为优化依据。
- 核心假设：**预测熵越低，模型越确定，预测损失也越低**。论文通过实验验证了熵与交叉熵损失之间存在高度正相关（Spearman 相关系数高达 0.87）。
- 因此，优化目标为：

$$\min_{\{\lambda_k\}} \sum_{k=1}^{K} \sum_{x_i \in \mathcal{B}_k} H(f_{\theta_{\text{MTL}}}(x_i))$$

其中 $H(\cdot)$ 是 Shannon 熵，$\mathcal{B}_k$ 是任务 k 的测试样本批次。

SLERP

$$\lambda_1^* = \frac{\sin((1-\lambda)\cdot\rho)}{\sin(\rho)} \text{ and } \lambda_2^* \frac{\sin(\lambda\cdot\rho)}{\sin(\rho)}$$

$$\rho = \arccos \frac{\tau_1 \cdot \tau_2}{|\tau_1| \cdot |\tau_2|}$$

Fisher—Merging

$$\theta^{(merge)} = \sum_{t=1}^{T} \frac{F^t}{\sum_{t=1}^{T} F^t} \theta^t$$

Ft is the diagonal of the Fisher information matrix with respect to task t

Matena, M. S., & Raffel, C. A. (2022). Merging models with fisher-weighted averaging. *Advances in Neural Information Processing Systems*, *35*, 17703-17716.

Charles Goddard, Shamane Siriwardhana, Malikeh Ehghaghi, Luke Meyers, Vlad Karpukhin, Brian Benedict, Mark McQuade, and Jacob Solawetz. 2024. Arcee's MergeKit: A Toolkit for Merging Large Language Models. arXiv preprint arXiv:2403.13257 (2024).

当参数本身是一个**矩阵**时（例如在多元统计分析中的协方差矩阵、因子载荷矩阵或回归系数矩阵），Fisher Information Matrix 的概念仍然适用，但需要将其理解为关于矩阵中**所有元素的联合信息度量**。简单来说，我们通常会把矩阵参数“拉直”成一个长向量，然后定义该向量的 Fisher Information Matrix。

1. 基本思想

假设参数矩阵为 Θ，维度为 $p \times q$。数据 X 的分布依赖于 Θ，对数似然函数为 $\ell(\Theta; X)$。

- **得分（Score）**：关于 Θ 的梯度是一个 $p \times q$ 的矩阵，记为 $S(\Theta) = \frac{\partial \ell}{\partial \Theta}$（即每个元素对 Θ_{ij} 求偏导）。
- **Fisher Information** 衡量的是得分矩阵的“方差”。但由于矩阵的方差难以直接用一个矩阵表示，通常我们通过 **向量化** 将其转化为向量形式：

  $$\boldsymbol{\theta} = \mathrm{vec}(\Theta) \quad (\text{长度为 } pq \text{ 的列向量})$$

  此时得分向量为 $\boldsymbol{s}(\boldsymbol{\theta}) = \mathrm{vec}(S(\Theta))$。

那么，参数矩阵 Θ 的 Fisher Information Matrix 定义为 $\boldsymbol{\theta}$ 的 Fisher Information Matrix，即一个 $pq \times pq$ 的矩阵：

$$\mathcal{I}(\boldsymbol{\theta}) = \mathbb{E}\left[\boldsymbol{s}(\boldsymbol{\theta})\boldsymbol{s}(\boldsymbol{\theta})^\top\right] = -\mathbb{E}\left[\frac{\partial^2 \ell}{\partial \boldsymbol{\theta} \partial \boldsymbol{\theta}^\top}\right].$$

Subspace-based methods

- Sparse Subspace Merging
  - TIES—Merging：

2. 修剪 (Trim)

保留每个任务向量中幅度最大的 top-$k\%$ 的参数，其余重置为 0：

$$\hat{\tau}_t = \text{top-}k\%(|\tau_t|) \odot \tau_t$$

或者等价地，每个元素 $(\hat{\tau}_t)_i$ 如果幅度不是前 $k\%$ 则置为 0。

3. 选举符号 (Elect Sign)

对于每个参数维度 p，根据所有任务中该参数值的符号，计算总质量，并选择总质量较大的符号作为最终符号 γ_m^p：

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

其中 $\text{sgn}(x)$ 返回 $+1$ 或 -1（当输入为 0 时，通常可定义为 0，但实际参与合并的为修剪后的非零值）。

4. 不相交合并 (Disjoint Merge)

对于每个参数维度 p，定义集合 $\mathcal{A}^p$ 为所有与最终符号 γ_m^p 一致的任务：

$$\mathcal{A}^p = \{t \in [n] \mid \text{sgn}(\hat{\tau}_t^p) = \gamma_m^p\}$$

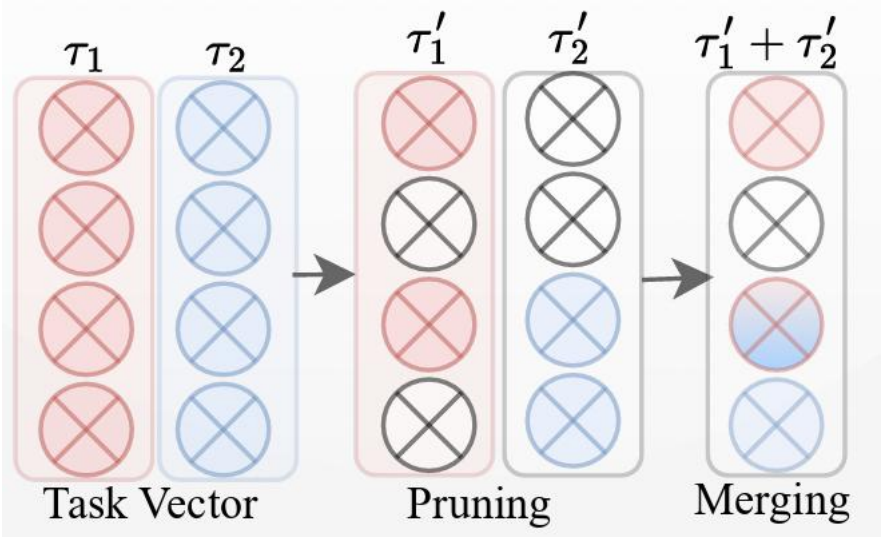
然后计算这些一致任务的任务向量值的均值，作为合并后的任务向量 τ_m：

$$\tau_m^p = \frac{1}{|\mathcal{A}^p|} \sum_{t \in \mathcal{A}^p} \hat{\tau}_t^p$$

5. 生成最终模型

最后，将合并后的任务向量乘以缩放因子 λ 加回到初始模型参数：

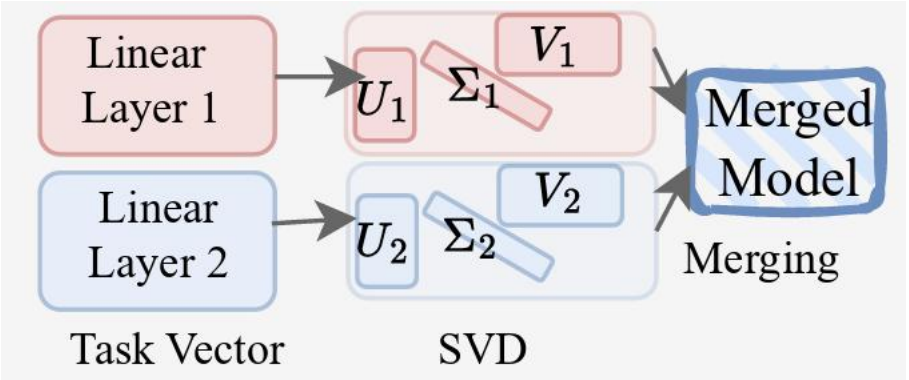
$$\theta_m = \theta_{\text{init}} + \lambda \cdot \tau_m$$



Yadav, P., Tam, D., Choshen, L., Raffel, C. A., & Bansal, M. (2023). Ties-merging: Resolving interference when merging models. *Advances in neural information processing systems*, *36*, 7093-7115.

Subspace-based methods

- **Low-Rank Subspace Merging**: Low-rank subspace methods extract the most informative components via matrix factorization for merging, while discarding less important components that are more likely to cause conflicts.
- TSV-M [56] applies SVD to task vectors, concatenates the left and right singular vectors, and then orthogonalizes them to remove redundant bases



Gargiulo, A. A., Crisostomi, D., Bucarelli, M. S., Scardapane, S., Silvestri, F., & Rodola, E. (2025). Task singular vectors: Reducing task interference in model merging. In *CVPR'25* (pp. 18695-18705).

6. 算法流程总结（Algorithm 1）

1. **输入**：每个任务在每一层的任务矩阵 $\Delta_i^{(l)}$，缩放因子 α。
2. **对每个任务**：
   - 计算 SVD：$\Delta_i = U_i \Sigma_i V_i^\top$，
   - 保留前 k 个奇异成分（$k = \text{rank}/T$）。
3. **拼接**所有任务的保留奇异向量和奇异值：
   - $U \leftarrow [U_1 | U_2 | \ldots | U_T]$，
   - $\Sigma \leftarrow \text{block-diag}(\Sigma_1, \ldots, \Sigma_T)$，
   - $V \leftarrow [V_1 | V_2 | \ldots | V_T]$。
4. **正交化**：
   - 对 U 做 SVD 得 $P_U D_U Q_U^\top$，令 $U_\perp = P_U Q_U^\top$，
   - 对 V 做 SVD 得 $P_V D_V Q_V^\top$，令 $V_\perp = P_V Q_V^\top$。
5. **重构**合并矩阵：$\hat{M} \leftarrow U_\perp \Sigma V_\perp^\top$。
6. **更新模型权重**：$\theta_{\text{MT}} \leftarrow \theta_{\text{pre}} + \alpha \hat{M}$。
7. **输出**合并后的模型权重。

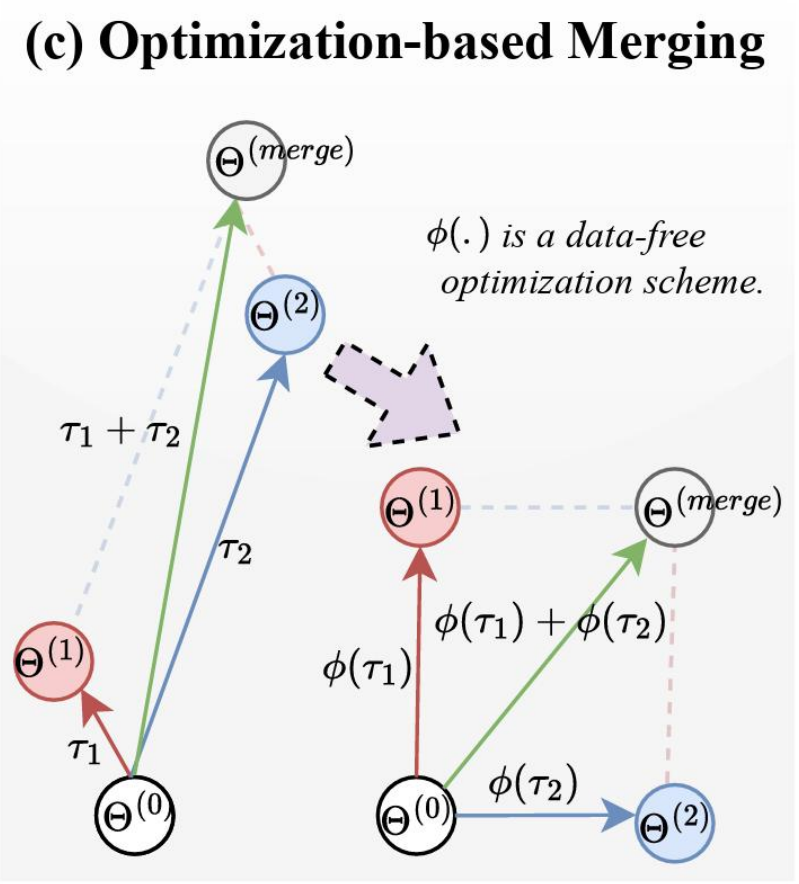
Optimization-based Merging Methods

- Optimization-based methods exploit the intrinsic properties of task vectors to formulate an optimization objective, thereby enabling data-free minimization of inter-task interference or information loss after merging.

$$\min_{\tau_{m,l}} \sum_i \frac{1}{\|\tau_{i,l}\|_F^2} \left(\|(\tau_{m,l} - \tau_{i,l})(\tau_{i,l})^\top\|_F^2 + \omega \cdot \|\tau_{m,l} - \tau_{i,l}\|_F^2 \right)$$

通过让合并后的任务向量 τ_m 在每个任务向量 τ_i 所张成的线性子空间上投影尽可能接近 τ_i 本身，从而在无数据的情况下消除干扰

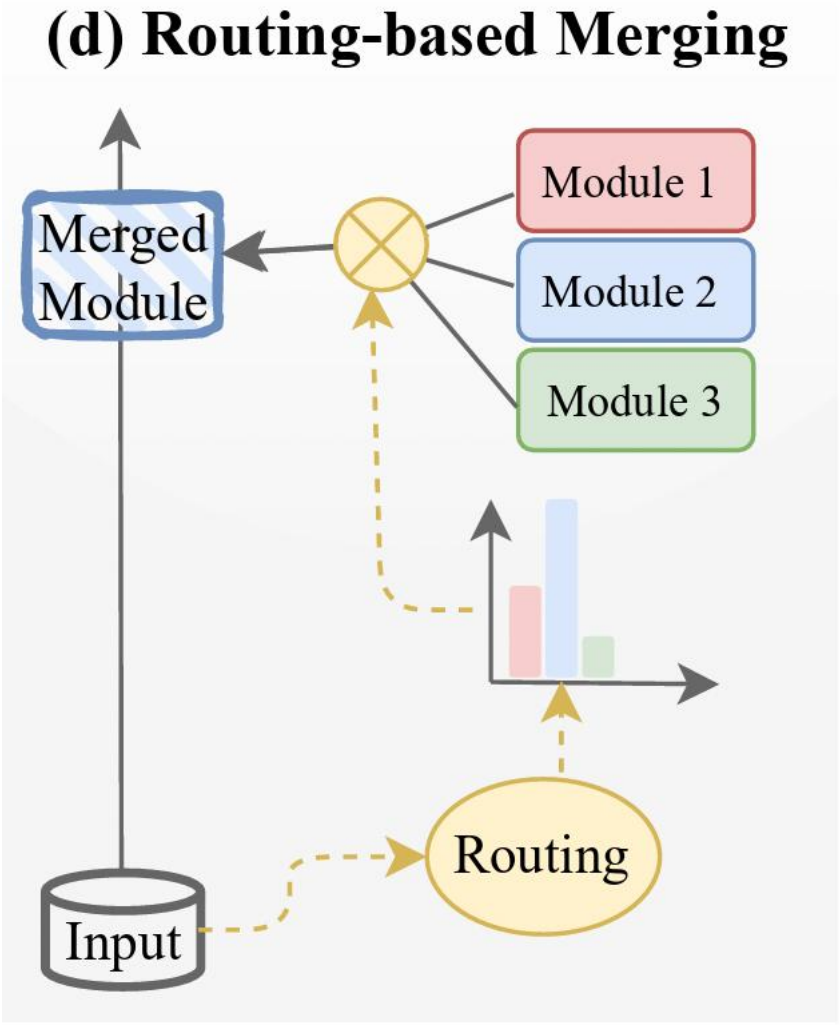
$$\boldsymbol{\tau}_{m,l} = \text{Matmul}(\textstyle\sum_i \frac{1}{\|\boldsymbol{\tau}_{i,l}\|_F^2} \boldsymbol{\tau}_{i,l}(\boldsymbol{\tau}_{i,l}^\top \boldsymbol{\tau}_{i,l} + \omega I), \\ (\textstyle\sum_i \frac{1}{\|\boldsymbol{\tau}_{i,l}\|_F^2} (\boldsymbol{\tau}_{i,l}^\top \boldsymbol{\tau}_{i,l} + \omega I))^{-1}) \quad (20)$$



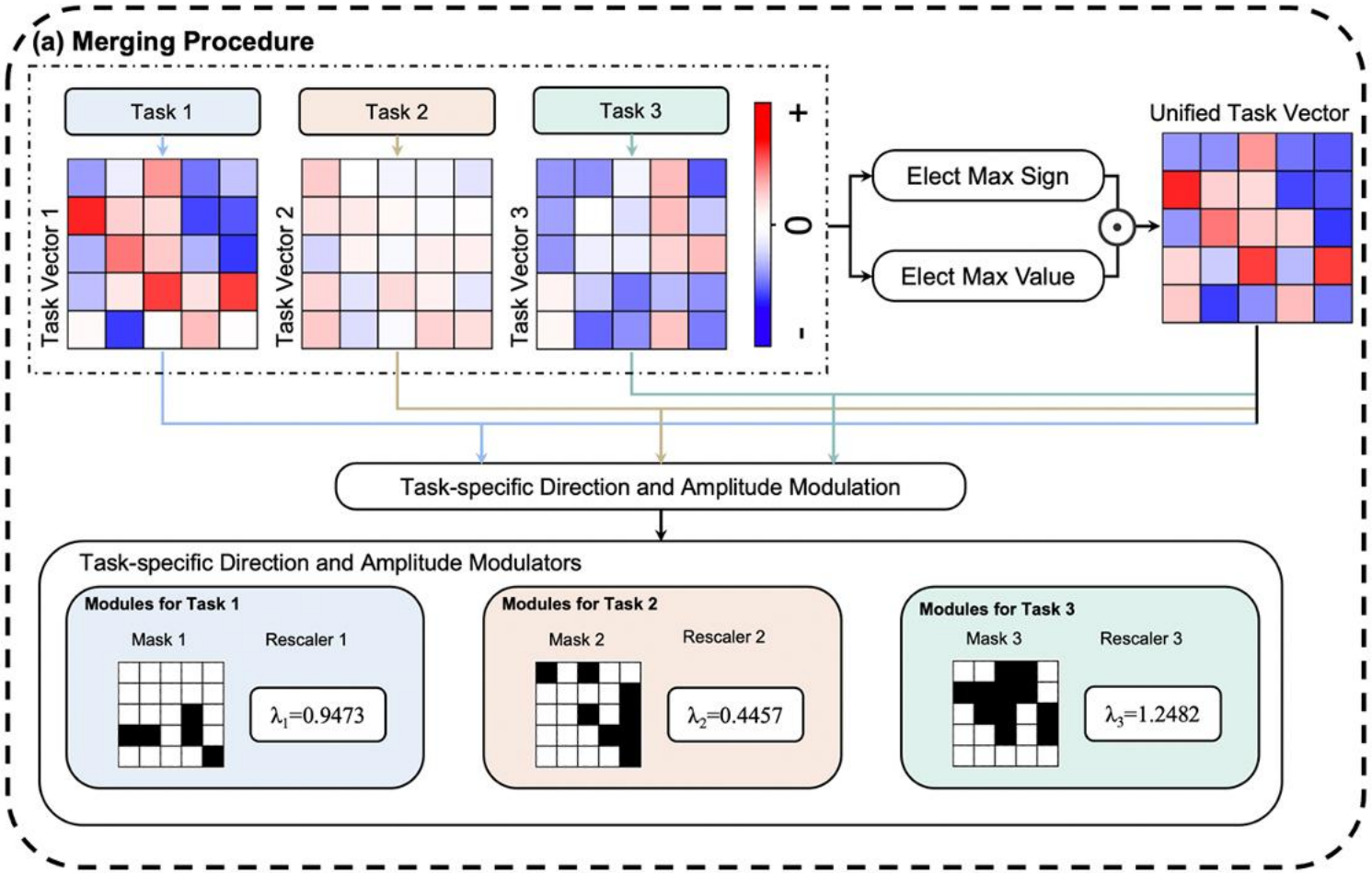
Cheng, R., Xiong, F., Wei, Y., Zhu, W., & Yuan, C. (2025). Whoever started the interference should end it: Guiding data-free model merging via task vectors. *ICML 2026*

Routing-based Merging Methods

Dynamically merge models (or subsets of layers) based on the samples/tasks during the inference phase.



EMR



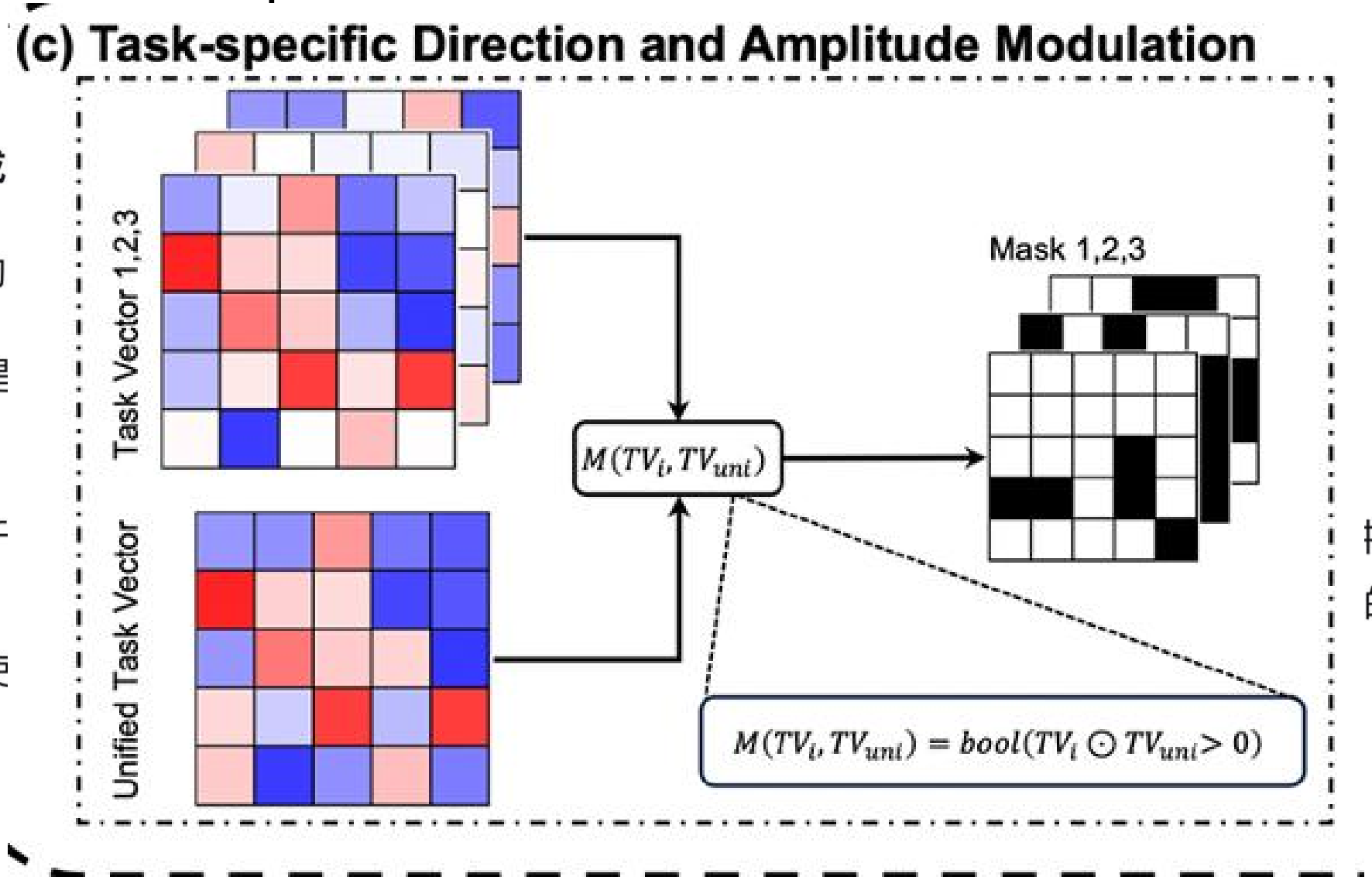
EMR-Me 个步骤。

步骤一：

这一步旨

- **选举符** 这个符号代表了在某个参
- **选举幅** 量中，选取**绝对值最大的那**
- **组合**： 上代表多数派的共识，在幅度上保留了该方向上的最大强度，旨在最大化地减少任务间的干扰。

Second Step



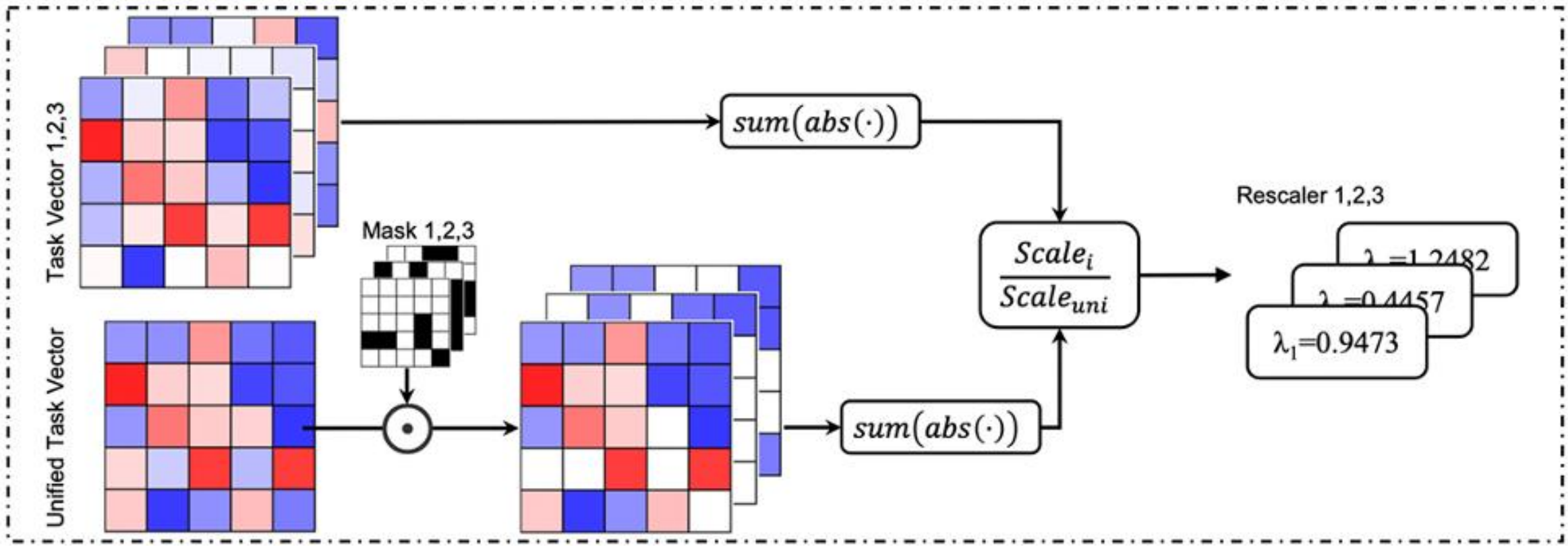
步骤二：生成

这一步的目的

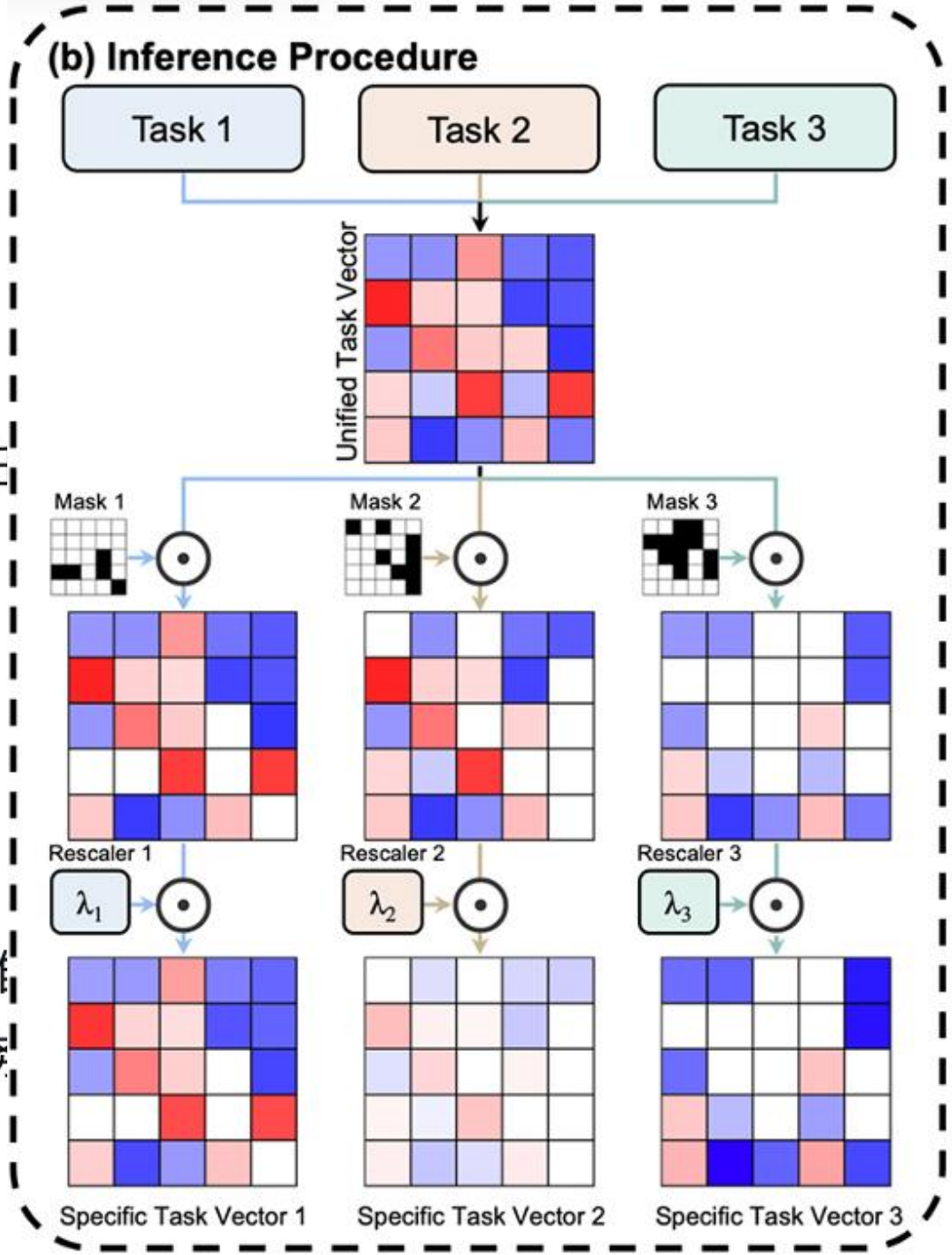
- 掩码 M_i 是
- 定义：M_i
- **作用**：对于 0（丢弃）。新，从而使

中突，则为的参数更

Third Step


Task Vector 1,2,3
Unified Task Vector
Mask 1,2,3
sum(abs(·))
sum(abs(·))
Scale_i / Scale_uni
Rescaler 1,2,3
λ_1=0.9473

Inference

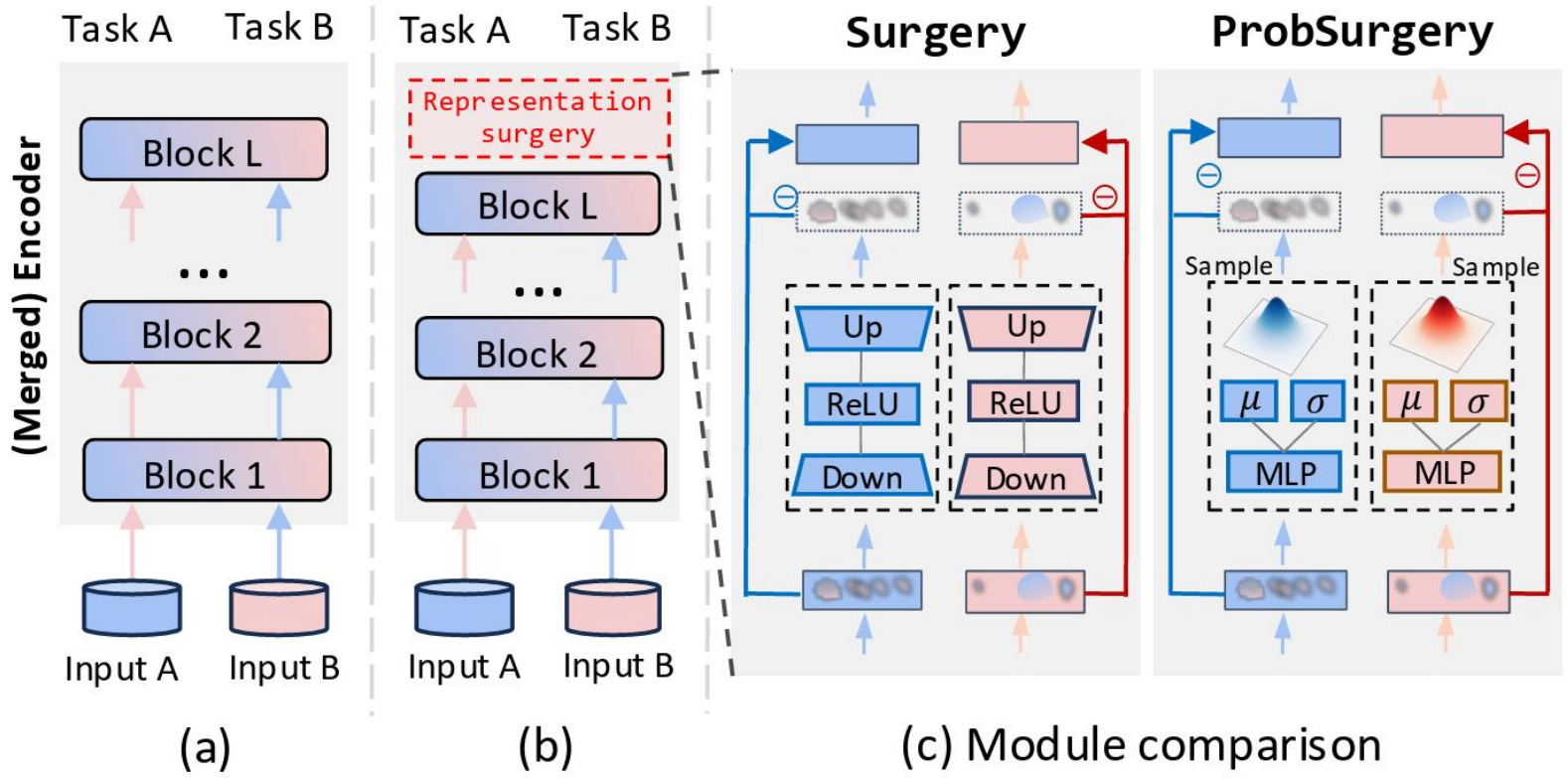


在推理时，针对特定的任 器）应用于统一任务向量，然后再加回预训练权重：

这样，最终用于任务 t 推 通用知识（τ_{uni}），又通过极轻量的调制器恢复了针对该任务

Post-calibration-based Methods

- Merged models suffer from representation bias, meaning the representations extracted by the independent and merged models are very different, leading to performance degradation in the merged model.



Surgery 方法

1. 动机

模型合并后，合并模型产生的特征表示与各任务专家模型的特征之间存在**表示偏差**，导致性能下降。Surgery 旨在通过后处理模块修正这种偏差。

2. 做法

- **模块结构**：Surgery 是一个轻量级的**三层全连接 MLP**，以合并模型输出的特征 $\mathbf{z}^{\theta_{\text{unif}}}$ 为输入，直接输出一个偏差向量 ξ。
- **任务特定性**：为每个任务 t 独立训练一个 Surgery 模块 g_{ω_t}，需要预先知道任务 ID。
- **训练目标**：利用无标签验证集，以自监督方式对齐修正后的特征与专家模型特征：

$$\underset{\{\omega_1,\ldots,\omega_T\}}{\arg\min} \sum_{t=1}^{T} \sum_{i=1}^{|D_t^{te}|} \psi\left(\mathbf{z}_{i,\text{unif}}^{\theta_{\text{unif}}} - \xi_{i,t},\ \mathbf{z}_i^{\theta_t}\right), \quad \xi_{i,t} = g_{\omega_t}(\mathbf{z}_{i,\text{unif}}^{\theta_{\text{unif}}})$$

  其中 ψ 为距离度量（如 L1 损失）。
- **推理**：直接用训练好的模块输出偏差并减去，得到修正后特征。

3. 局限性

- **表示能力有限**：确定性建模无法捕捉合并过程中由参数干扰带来的不确定性。
- **适用性差**：每个任务需独立模块，无法共享，参数随任务数线性增长，且推理时需任务 ID。

ProbSurgery 方法

1. 核心思想

将表示偏差建模为**概率分布**，通过采样得到偏差值，从而自然处理不确定性，提升表示能力和泛化性。

2. 做法

- **模块结构**：同样是三层 MLP，但输出变为**均值 μ 和对数方差** $\log \sigma^2$，定义偏差分布 $p(\xi|Q) \sim \mathcal{N}(\mu, \sigma^2)$。
- **重参数化采样**：训练时使用重参数化技巧 $\hat{\xi} = \mu + \epsilon \cdot \sigma,\ \epsilon \sim \mathcal{N}(0,1)$ 获得可微的偏差样本；推理时直接使用 μ 作为偏差。
- **训练目标**：在原有对齐损失基础上增加 **KL 散度正则项**，约束分布接近标准正态分布：

$$\underset{\{\omega_1,\ldots,\omega_T\}}{\arg\min} \sum_{t=1}^{T} \sum_{i} \psi(\mathbf{z}_{i,\text{unif}}^{\theta_{\text{unif}}} - \hat{\xi}_{i,t}, \mathbf{z}_i^{\theta_t}) + \lambda\, \mathrm{KL}\big(p(\xi|Q)\,\|\,\mathcal{N}(0,1)\big)$$

- **扩展性**：可进一步升级为**一对多（one-to-all）设置**，即用一个 ProbSurgery 模块处理所有任务，通过 DFA（直接特征对齐）或 PDA（代理分布对齐）实现。

3. 优势

- **更强表示能力**：通过分布建模不确定性，更好应对合并干扰。
- **更优泛化**：PAC-Bayes 理论证明其泛化误差界比确定性方法更紧。
- **更高效率**：单个模块可服务所有任务，参数不随任务数增加；收敛速度更快（300步即超 Surgery 最佳）。
- **更好鲁棒性**：在领域迁移、分布外检测等场景下表现更稳定。

Table 2. A brief overview of representative model merging methods. *Category*: The methodological category each approach belongs to. *Parameters*: The number of parameters in the merged model, where 1× denotes the parameter size of a single-task model and N denotes the total number of tasks. *Tuning-Free*: Whether the merging process is tuning-free. *Required Data*: What additional data (if any) are required for merging. *Pattern*: Whether the merging pattern is static or dynamic. *Weighted Level*: The granularity at which multiple models are weighted during merging. Here, ✓=yes and ✗=no.

Method	Category	Parameters	Tuning-Free	Required Data	Pattern	Weighted Level
Traditional MTL	-	= 1×	✗	Labeled Training Dataset	Static	-
Individual Models	-	= N×	✗	Labeled Training Dataset	Static	-
Weighted Average [241]	Basic	= 1×	✓	✗	Static	Global
Task Arithmetic [79]	Basic	= 1×	✓	✗	Static	Global
SLERP [59]	Weighted	= 1×	✓	✗	Static	Task
Evolutionary-model-merge [6]	Weighted	= 1×	✓	Labeled Validation Dataset	Static	Layer
AdaMerging [261]	Weighted	= 1×	✗	Unlabeled Test Dataset	Static	Task/Layer
MetaGPT [287]	Weighted	= 1×	✓	✗	Static	Task
Fisher-Merging [140]	Weighted	= 1×	✗	Labeled Validation Dataset	Static	Parameter
RegMean [96]	Weighted	= 1×	✗	Labeled Validation Dataset	Static	Parameter
Ties-Merging [252]	Sparse Subspace	= 1×	✓	✗	Static	Global
DARE [269]	Sparse Subspace	= 1×	✓	✗	Static	Global
Model Breadcrumbs [37]	Sparse Subspace	= 1×	✓	✗	Static	Global
Model Tailor [290]	Sparse Subspace	= 1×	✗	Labeled Training Dataset	Static	Global
Localize-and-Stitch [67]	Sparse Subspace	= 1×	✓	Labeled Validation Dataset	Static	Global
Consensus Merging [229]	Sparse Subspace	= 1×	✓	✗	Static	Task
DELLA-Merging [38]	Sparse Subspace	= 1×	✗	✗	Static	Global
PCB-Merging [46]	Sparse Subspace	= 1×	✓	✗	Static	Task
KnOTS [189]	Low-Rank Subspace	= 1×	✓	✗	Static	Global
HO-GSVD [185]	Low-Rank Subspace	= 1×	✓	✗	Static	Global
TSV-M [56]	Low-Rank Subspace	= 1×	✓	✗	Static	Global
ISO-Merging [138]	Low-Rank Subspace	= 1×	✓	✗	Static	Global
OPCM [203]	Low-Rank Subspace	= 1×	✓	✗	Static	Task
AWD [249]	Optimization	= 1×	✗	✗	Static	Task
DOGE [238]	Optimization	= 1×	✗	✗	Static	Task
WUDI-Merging [26]	Optimization	= 1×	✗	✗	Static	Global
DOP [260]	Optimization	= 1×	✗	✗	Static	Task
Surgery [258]	Calibration	> 1×	✗	Unlabeled Test Dataset	Static	Global
ProbSurgery [236]	Calibration	> 1×	✗	Unlabeled Test Dataset	Static	Global
EMR Merging [77]	Routing	> 1×	✓	✗	Dynamic	Task
Twin Merging [134]	Routing	> 1×	✗	Labeled Validation Dataset	Dynamic	Sample
WEMoE [201]	Routing	≫ 1×	✗	Unlabeled Test Dataset	Dynamic	Sample

Summary

- Training-free methods and training based methods
- data-dependent approaches and data-independent approaches
- Static scheme and dynamic
- Weighting granularity: global, task-wise, layer-wise, and parameter-wise weighting.
- Overall, there is no single “perfect” merging method that fits all scenarios; the most appropriate approach should be chosen based on performance requirements, implementation complexity, and the availability of data and computational resources.

Application of Model Merging in Foundation Models

Table 4. A summary of the application of model merging techniques in foundation models.

Scenarios	The Main Purpose of Model Merging
Large Language Models (§4.1)	Enhancing the domain-specific capabilities of pre-trained LLMs or editing old knowledge
Multimodal Large Language Models (§4.2)	Understanding content across multiple modalities using a single model
Visual Generative Models (§4.3)	Generate images with multiple styles or achieve image-style transformation

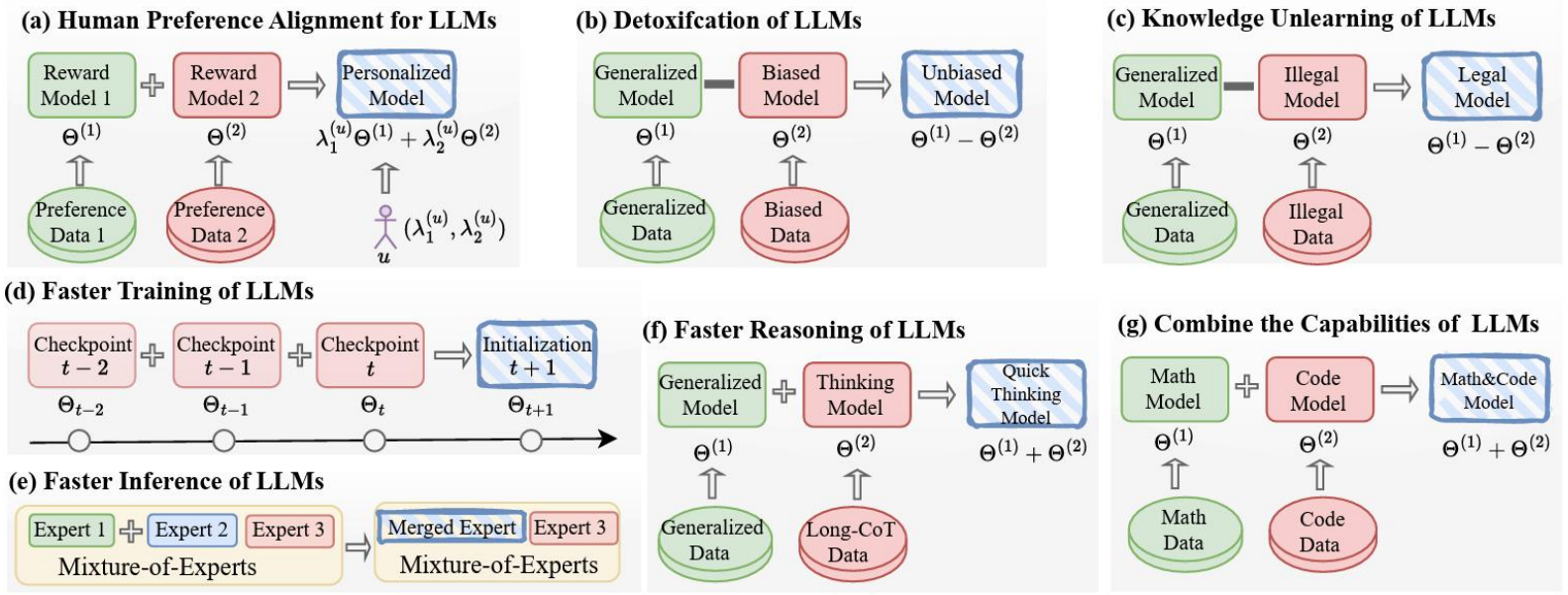


Fig. 7. Example application scenarios of model merging in large language models.

Application of Model Merging in Different Machine Learning Subfields

Table 5. A summary of the application of model merging techniques in different machine learning subfields.

Scenarios	The Main Purpose of Model Merging
Continual Learning (§5.1)	Avoiding catastrophic forgetting with respect to old tasks
Multi-Task / Multi-Domain / Multi-Objective / Auxiliary Learning (§5.2)	Performing multiple tasks / domains / objectives via one model
Domain / Out-of-Distribution Generalization (§5.3)	Achieving generalization to unknown target domains or distributions
Federated Learning (§5.4)	Merging local models provided by different clients
Zero-shot / Few-shot Learning (§5.5)	Multiple related models are merged to improve the zero-shot / few-shot learning ability on new tasks
Adversarial Learning (§5.6)	Implementing model poisoning attack, defense, and copyright protection