Evolutionary Negative Module Pruning for Better LoRA Merging

Anda Cao[1], Zhuo Gou[2], Yi Wang[1], Kaixuan Chen[1,3,4], Yu Wang[1]
Can Wang[1,3,4], Mingli Song[1,3,4], Jie Song[2]*
[1]College of Computer Science and Technology, Zhejiang University
[2]School of Software Technology, Zhejiang University
[3]State Key Laboratory of Blockchain and Security, Zhejiang University
[4]Hangzhou High-Tech Zone (Binjiang) Institute of Blockchain and Data Security
{caoanda, gouzhuo, y_w, chenkx, yu.wang}@zju.edu.cn
{wcan, brooksong, sjie}@zju.edu.cn

Abstract

Merging multiple Low-Rank Adaptation (LoRA) experts into a single backbone is a promising approach for efficient multi-task deployment. While existing methods strive to alleviate interference via weight interpolation or subspace alignment, they rest upon the implicit assumption that all LoRA matrices contribute constructively to the merged model. In this paper, we uncover a critical bottleneck in current merging paradigms: the existence of *negative modules*—specific LoRA layers that inherently degrade global performance upon merging. We propose **E**volutionary **N**egative **M**odule **P**runing (**ENMP**), a plug-and-play LoRA pruning method to locate and exclude these detrimental modules prior to merging. By leveraging an evolutionary search strategy, ENMP effectively navigates the discrete, non-differentiable landscape of module selection to identify optimal pruning configurations. Extensive evaluations demonstrate that ENMP consistently boosts the performance of existing merging algorithms, achieving a new state-of-the-art across both language and vision domains. Code is available at https://github.com/CaoAnda/ENMP-LoRAMerging.

1 Introduction

Model merging has gained prominence as a scalable paradigm for integrating multiple fine-tuned models into a unified backbone without the prohibitive costs of retraining. Task Arithmetic (TA) (Ilharco et al., 2023) laid the groundwork for this field by conceptualizing parameter differences as steerable task vectors. Subsequent advancements, such as TIES-Merging (Yadav et al., 2023), have refined this approach by resolving sign conflicts and pruning redundant parameters. Beyond element-wise aggregation, recent efforts (Choi et al., 2025; Gargiulo et al., 2025; Marczak et al., 2025) have pivoted toward the spectral properties of models, utilizing Singular Value Decomposition (SVD) to harmonize parameter-space conflicts. While these methods make significant advancements for efficient multi-task deployment, they are largely designed for full-parameter fine-tuning.

However, the landscape of model adaptation is shifting alongside the rapid scaling of neural networks (OpenAI et al., 2024; Dubey et al., 2024; Yang et al., 2025). As full-rank fine-tuning becomes computationally unsustainable, Parameter-Efficient Fine-Tuning (PEFT) (Houlsby et al., 2019; Li and Liang, 2021; Lester et al., 2021; Hu et al., 2022) has emerged as the preferred alternative. Among these, LoRA (Hu et al., 2022) is particularly dominant due to its minimal parameter footprint and robust convergence properties (Dettmers et al., 2023; Zhang et al., 2023). With the widespread adoption of LoRA, a single backbone is often required to serve multiple tasks (Wei et al., 2022; Sanh et al., 2022), resulting in a rapidly growing number of task-specific adapters. Consequently, merging these diverse LoRA experts into a single, cohesive model has become highly attractive for practical deployment. Unfortunately, conventional merging algorithms, which were originally designed for full-parameter adaptations, frequently struggle to account for the unique structural and low-rank constraints inherent to LoRA-adapted models (Stoica et al., 2025).

Specifically within the LoRA context, prior works such as KnOTS (Stoica et al., 2025) and CoreSpace (Panariello et al., 2025) attribute merging failures to subspace misalignment among independently trained adapters. To mitigate this, they project disparate adapters into a shared subspace to enforce compatibility. While effective at resolving geometric misalignment, these methods rest upon the idealized assumption that every LoRA module (*i.e.*, the low-rank adaptation matrices) contributes constructively to the merging performance. In contrast, our empirical study, as shown in Figure 1,

*Corresponding author.

Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), pages 37297–37310

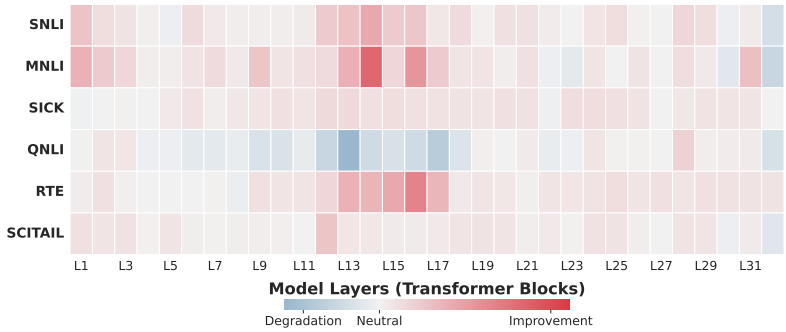

Figure 1: Not all LoRA modules are constructive for merging. In this empirical study, we perform a *leave-one-out* analysis by removing a single LoRA module at a time and merging the rest via Task Arithmetic (TA). The heatmap visualizes the change in average normalized performance of Llama-3-8B across 6 Natural Language Inference (NLI) tasks relative to the baseline (full merge). Red regions indicate a performance improvement after pruning, while blue regions indicate a performance drop.

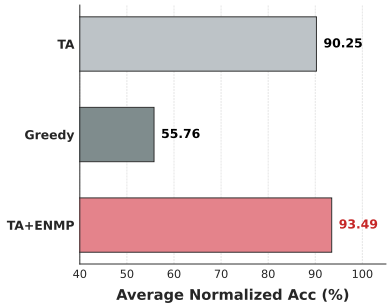

Figure 2: Limitations of Greedy Pruning. The Greedy strategy removes all modules that appear detrimental (as suggested in Fig. 1). However, this approach ignores cross-layer dependencies, resulting in a degraded accuracy of 55.76%.

uncovers a counter-intuitive phenomenon: removing the entire module in a specific layer from a task-specific LoRA can yield a performance gain in the merged model. This suggests that certain modules act as *negative modules*, which exacerbate interference rather than contributing beneficial task-specific information. Identifying and pruning these detrimental modules is therefore a critical prerequisite for effective model merging.

However, identifying the optimal subset of modules for pruning presents a formidable challenge, as the impact of a module on merging performance is interdependent: a module characterized as "negative" within the full set may become constructive once some detrimental modules are removed, and vice versa. This conditional dependency leads to a complex optimization landscape. Consequently, the greedy strategy, which evaluates and prunes modules independently, fails to capture higher-order interactions, resulting in significant performance degradation (Figure 2). Furthermore, the search space for the pruning mask suffers from a combinatorial explosion, yielding 2^N possible states for N modules, which renders exhaustive search computationally intractable. In this work, we propose an evolutionary search approach, termed Evolutionary Negative Module Pruning (ENMP), to efficiently explore the configuration space and locate the optimal pruning mask that maximizes collective performance. By enabling the precise pruning of negative modules, ENMP effectively boosts the performance of existing methods, achieving a new state-of-the-art.

To summarize, our contributions are as follows:

- We unveil the phenomenon of *negative modules* and demonstrate that pruning them effectively alleviates task interference.
- We propose ENMP, a plug-and-play LoRA pruning method to locate and exclude these detrimental modules prior to merging.
- Extensive evaluations across language and vision domains demonstrate that ENMP consistently boosts the performance of existing merging algorithms.

2 Related Work

2.1 Model Merging

Model merging aims to integrate multiple models independently fine-tuned on different tasks into a single unified model without additional training (Ilharco et al., 2023). The field has evolved from simple weight averaging to sophisticated techniques that manipulate task-specific updates. Task Vectors (Ilharco et al., 2023) established the foundation by demonstrating that weight differences support arithmetic operations for steering model behavior. Building on this, subsequent approaches focus on mitigating interference among conflicting parameters. TIES-Merging (Yadav et al., 2023) resolves sign conflicts and prunes insignificant updates, while DARE (Yu et al., 2024) and Model Breadcrumbs (Davari and Belilovsky, 2024) exploit the parameter redundancy in fine-tuned models to enable aggressive sparsification without performance loss. Beyond element-wise manipulation, spectral methods such as TSV (Gargiulo et al., 2025),

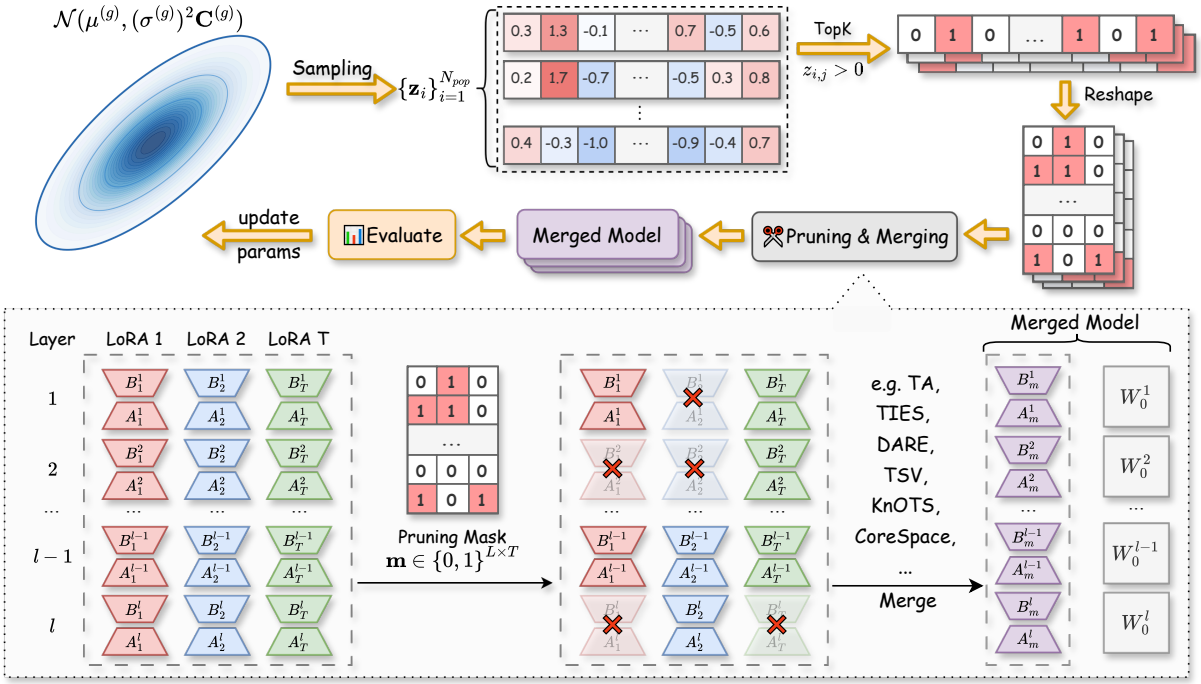

Figure 3: Overview of the proposed framework. The upper part illustrates the optimization loop based on evolutionary strategies (e.g., CMA-ES). Latent variables $\{\mathbf{z}_i\}$ are sampled from a evolving Gaussian distribution $\mathcal{N}(\mu^{(g)}, (\sigma^{(g)})^2\mathbf{C}^{(g)})$ and converted into binary pruning masks via mapping and reshaping operations. The lower part details the *Pruning & Merging* process: the pruning mask $\mathbf{m}$ is applied to the LoRA adapters to prune negative modules. The remaining adapters are aggregated using existing merging methods (e.g., TIES, DARE) to form the final merged model. The evaluation performance is used to update the distribution parameters iteratively.

Iso-C (Marczak et al., 2025), and CART (Choi et al., 2025) leverage low-rank properties and singular value alignment to further reduce interference. While effective for full-rank fine-tuned models, these techniques often exhibit instability or suboptimal performance when directly applied to LoRA-adapted models (Tang et al., 2024).

2.2 LoRA Merging

Existing approaches for merging LoRA models can be categorized based on whether they require specialized training strategies. Tang et al. (2024) attribute the merging difficulty to increased weight-entanglement and proposes a specialized fine-tuning method involving partial linearization. In contrast, KnOTS (Stoica et al., 2025) presents a gradient-free framework that performs post-hoc alignment of low-rank subspaces using SVD. Building on this, CoreSpace (Panariello et al., 2025) further optimizes the paradigm to ensure no information loss. However, these methods implicitly assume that all modules contribute constructively, neglecting the impact of negative modules. To address this, our proposed ENMP departs from the full-retention assumption by introducing an evolutionary search to prune negative modules.

3 Methodology

Figure 3 illustrates the proposed ENMP framework. Following the preliminaries (Sec. 3.1), we detail the pruning formulation (Sec. 3.2) and the optimization process via evolutionary search (Sec. 3.3).

3.1 Preliminaries

Low-Rank Adaptation (LoRA). Let $\mathbf{W}_0 \in \mathbb{R}^{d_{\text{out}} \times d_{\text{in}}}$ denote the weight matrix of a dense layer in a pre-trained backbone model. Standard full-rank fine-tuning updates the entire weight matrix, which is computationally expensive and memory-intensive. LoRA (Hu et al., 2022) hypothesizes that the weight updates have a low intrinsic rank and parameterizes the update $\Delta\mathbf{W}$ by decomposing it into two low-rank matrices $\mathbf{B} \in \mathbb{R}^{d_{\text{out}} \times r}$ and $\mathbf{A} \in \mathbb{R}^{r \times d_{\text{in}}}$, where $r \ll \min(d_{\text{in}}, d_{\text{out}})$. The forward pass is given by:

$$\mathbf{h} = \mathbf{W}_0\mathbf{x} + \Delta\mathbf{W}\mathbf{x} = \mathbf{W}_0\mathbf{x} + \mathbf{B}\mathbf{A}\mathbf{x}. \quad (1)$$

In the context of LoRA merging, we consider T diverse tasks. Each task $t \in \{1, \ldots, T\}$ is associated with a specific LoRA update $\Delta\mathbf{W}_t$. The collection of these task-specific updates is denoted as $\mathcal{T} = \{\Delta\mathbf{W}_1, \ldots, \Delta\mathbf{W}_T\}$.

Model Merging. Given the set of task-specific experts $\mathcal{T}$ defined above, the goal of model merging is to combine them into a single unified model $\mathbf{W}_{\text{merged}}$, which is capable of handling all tasks simultaneously without retraining. A widely adopted formulation, such as Task Arithmetic (Ilharco et al., 2023), computes the weighted sum of task vectors:

$$\mathbf{W}_{\text{merged}} = \mathbf{W}_0 + \lambda \sum_{t=1}^{T} \Delta\mathbf{W}_t, \tag{2}$$

where λ is a global scaling coefficient that controls the strength of the merged updates.

3.2 Merging with Negative Module Pruning

Given the set of LoRA updates $\mathcal{T}$ defined above, standard approaches typically perform a global aggregation across all parameters. We posit that this unselective full aggregation introduces negative modules—specific layers from certain task LoRAs that induce incompatibility and degrade global performance when included. Conversely, removing these modules can recover the model's capabilities.

To implement this, we propose to augment the merging process with a selective pruning mechanism. Let L denote the number of Transformer layers in the backbone. We define the fundamental pruning unit as the aggregation of LoRA updates for query, key, value, and output projections within a layer, treating them as an indivisible whole to preserve the internal semantic consistency of the attention mechanism. Consequently, the total number of pruning units is $N = L \times T$. We introduce a binary pruning mask $\mathbf{m} \in \{0, 1\}^{L \times T}$, where $m_{l,t} = 1$ indicates that the corresponding LoRA module (`q_proj`, `k_proj`, `v_proj`, `out_proj`) at layer l from task t is removed.

We formally define the merging process with module pruning using a generic aggregation function $\Phi(\cdot)$. Let $\mathcal{S}^{(l)}(\mathbf{m}) = \{\Delta\mathbf{W}_t^{(l)} \mid m_{l,t} = 0\}$ denote the set of retained LoRA modules for layer l based on the pruning mask $\mathbf{m}$ (where 1 indicates pruned). The merged weight is formulated as:

$$\mathbf{W}_{\text{ENMP}}^{(l)}(\mathbf{m}) = \mathbf{W}_0^{(l)} + \lambda \cdot \Phi\left(\mathcal{S}^{(l)}(\mathbf{m})\right). \tag{3}$$

Specifically, we define Task Arithmetic with module pruning as:

$$\Phi_{\text{TA-ENMP}}\left(\mathcal{S}^{(l)}(\mathbf{m})\right) = \sum_{\Delta\mathbf{W}_t^{(l)} \in \mathcal{S}^{(l)}(\mathbf{m})} \Delta\mathbf{W}_t^{(l)}. \tag{4}$$

Our objective is to find the optimal pruning mask $\mathbf{m}^*$ that maximizes the collective performance across all tasks. Let $\mathcal{D}_{val}$ represent the validation data and $\mathcal{M}(\cdot)$ be a comprehensive performance metric (e.g., normalized average accuracy). The optimization problem is formulated as:

$$\mathbf{m}^* = \underset{\mathbf{m} \in \{0,1\}^{L \times T}}{\arg\max}\ \mathcal{M}\left(\mathbf{W}_{\text{ENMP}}(\mathbf{m}); \mathcal{D}_{val}\right). \tag{5}$$

3.3 Evolutionary Search with CMA-ES

Directly searching for the binary pruning mask $\mathbf{m}$ in Eq. (5) is computationally intractable due to the combinatorial explosion of the discrete search space (2^N). More importantly, the decision to prune a specific module is not independent; it heavily relies on the presence or absence of other modules (i.e., cross-layer couplings). To address these issues, we propose to formulate the pruning task as a search problem within a continuous latent space.

We employ the Covariance Matrix Adaptation Evolution Strategy (CMA-ES) (Hansen, 2016) as our optimizer. We select CMA-ES specifically for its capability to model the dependencies between decision variables via a covariance matrix, which allows it to capture the complex interactions between LoRA modules that greedy methods overlook. For clarity and reproducibility, the detailed pseudocode is provided in Appendix A.

Latent Variable & Mask Mapping. Since CMA-ES operates on continuous one-dimensional vectors, we first represent the discrete pruning mask as a flattened vector $\mathbf{m}^{\text{flat}} \in \{0, 1\}^N$. To bridge the gap between the continuous search space and our discrete objective, we introduce a latent vector $\mathbf{z} \in \mathbb{R}^N$, where each scalar entry z_j serves as a learnable *negativity score* for the j-th module. In this formulation, a higher z_j signifies a stronger tendency for the corresponding module to be pruned.

To translate this continuous score $\mathbf{z}$ into the binary pruning mask, we apply a dynamic thresholding strategy controlled by a maximum pruning ratio $k \in [0, 1)$. Let $N_{\text{prune}} = \lfloor k \cdot N \rfloor$ denote the budget for allowed removals. We define $\mathcal{K}$ as the set of indices corresponding to the N_{prune} largest elements in $\mathbf{z}$. The pruning mask vector $\mathbf{m}^{\text{flat}}$ is

derived by:

$$m_j^{\text{flat}} = \begin{cases} 1 & \text{if } j \in \mathcal{K} \text{ and } z_j > 0 \\ 0 & \text{otherwise} \end{cases}. \quad (6)$$

This mapping ensures that the optimization algorithm can explore the continuous landscape of module interference while producing valid discrete masks, which are subsequently reshaped back to the $L \times T$ grid for the physical pruning process described in Section 3.2.

Conservative Initialization. We initialize the mean vector of the CMA-ES population to a uniform value of -1 (i.e., $\boldsymbol{\mu}^{(0)} = -\mathbf{1}$). Since our mapping requires $z_j > 0$ to trigger pruning, this negative initialization ensures that the search begins from the *fully merging* state (where all $m_{l,t} = 0$).

Search Process. The optimization proceeds iteratively over G generations. In each generation g, the algorithm executes three key steps:

- **Sampling:** We generate a population of N_{pop} candidate latent vectors $\{\mathbf{z}_1, \ldots, \mathbf{z}_{N_{pop}}\}$ from a multivariate normal distribution:

$$\mathbf{z}_i \sim \mathcal{N}(\boldsymbol{\mu}^{(g)}, (\sigma^{(g)})^2 \mathbf{C}^{(g)}), i = 1, \ldots, N_{pop}, \quad (7)$$

  where $\boldsymbol{\mu}^{(g)}$ represents the current estimate of the optimal negativity scores, $\sigma^{(g)}$ is the step size, and $\mathbf{C}^{(g)} \in \mathbb{R}^{N \times N}$ is the covariance matrix determining the geometric shape and variable dependencies of the search distribution.

- **Evaluation:** For each candidate $\mathbf{z}_i$, we generate the binary mask $\mathbf{m}_i$ via Eq. (6). We then construct the merged model $\mathbf{W}_{\text{ENMP}}(\mathbf{m}_i)$ and evaluate its fitness (e.g., validation normalized accuracy) to obtain a score $\mathcal{M}_i$.

- **Update:** The candidates are sorted by their fitness scores. The mean $\boldsymbol{\mu}^{(g+1)}$ is updated via a weighted average of the top-performing candidates. Simultaneously, the step-size $\sigma^{(g+1)}$ and covariance matrix $\mathbf{C}^{(g+1)}$ are adapted to control the exploration magnitude and capture the dependencies between modules.

This ability to model variable correlations allows ENMP to navigate the complex, non-separable landscape of module interference.

Final Model Construction & Complexity. We track the candidate latent vector $\mathbf{z}_{\text{best}}$ that achieves the highest fitness score throughout the evolutionary process. Upon convergence or exhaustion of the generation budget, the optimal binary mask is derived as $\mathbf{m}^* = \text{Mapping}(\mathbf{z}_{\text{best}})$. The final unified model is constructed by applying this mask to the merging formulation in Eq. (3):

$$\mathbf{W}_{\text{final}} = \mathbf{W}_{\text{ENMP}}(\mathbf{m}^*) = \mathbf{W}_0 + \lambda \cdot \Phi\left(\mathcal{S}(\mathbf{m}^*)\right). \quad (8)$$

Notably, this optimization process represents a *one-time offline cost*. The merged model retains the identical architectural footprint as the original pre-trained model, incurring zero additional inference overhead in terms of latency or memory usage.

4 Experiments

4.1 Experimental Setup

Benchmarks and Metrics. To ensure fair comparison and reproducibility, we follow the experimental protocols established in prior studies (Stoica et al., 2025; Panariello et al., 2025), conducting evaluations across two domains: Natural Language Processing (NLP) and Computer Vision (CV).

For the NLP benchmark, we focus on 6 Natural Language Inference (NLI) tasks: `SNLI` (Bowman et al., 2015), `MNLI` (Williams et al., 2018), `SICK` (Marelli et al., 2014), `QNLI` (Wang et al., 2018), `RTE` (Wang et al., 2018), and `SciTail` (Khot et al., 2018). For the CV benchmark, we use 8 image classification datasets: `Cars` (Krause et al., 2013), `DTD` (Cimpoi et al., 2014), `EuroSAT` (Helber et al., 2019), `GTSRB` (Stallkamp et al., 2011), `MNIST` (LeCun and Cortes, 2010), `RESISC45` (Cheng et al., 2017), `SUN397` (Xiao et al., 2016), and `SVHN` (Netzer et al., 2011).

To account for the varying difficulty levels across diverse tasks, we adopt *Normalized Accuracy* as our primary evaluation metric. This metric calibrates the merged model's performance relative to the single-task experts, ensuring a balanced comparison. For completeness, the detailed absolute accuracy scores for all individual tasks are reported in Appendix D. Formally, for a given task t, let $\text{Acc}_{\text{merged}}^{(t)}$ denote the accuracy of the merged model and $\text{Acc}_{\text{expert}}^{(t)}$ be the accuracy of the single-task expert. The normalized accuracy is defined as:

$$\text{NormAcc}^{(t)} = \frac{\text{Acc}_{\text{merged}}^{(t)}}{\text{Acc}_{\text{expert}}^{(t)}}. \quad (9)$$

This metric reflects how well the merged model retains task-specific capabilities relative to the ded-

Method	SNLI	MNLI	SICK	QNLI	RTE	SciTail	Avg.	Δ
TA	93.60	**95.29**	88.00	68.73	99.19	96.69	90.25	-
TA + ENMP	**95.93**$_{\pm 0.28}$	94.32$_{\pm 0.43}$	**91.59**$_{\pm 1.78}$	**80.40**$_{\pm 0.60}$	**101.34**$_{\pm 0.46}$	**97.37**$_{\pm 0.13}$	**93.49**$_{\pm 0.26}$	**+3.24**
TIES	94.86	96.71	80.79	71.54	**100.00**	96.00	89.99	-
TIES + ENMP	**96.57**$_{\pm 0.27}$	**98.23**$_{\pm 0.63}$	**92.73**$_{\pm 0.81}$	**94.95**$_{\pm 0.67}$	99.46$_{\pm 0.47}$	**96.39**$_{\pm 0.42}$	**96.39**$_{\pm 0.45}$	**+6.40**
DARE	94.50	96.87	77.68	72.44	97.58	96.15	89.20	-
DARE + ENMP	**96.09**$_{\pm 0.58}$	**97.77**$_{\pm 0.71}$	**92.34**$_{\pm 0.66}$	**94.94**$_{\pm 0.74}$	**98.66**$_{\pm 1.23}$	**97.22**$_{\pm 0.26}$	**96.17**$_{\pm 0.14}$	**+6.97**
TSV	95.37	**95.13**	88.85	76.82	**101.61**	97.56	92.56	-
TSV + ENMP	**96.45**$_{\pm 0.75}$	94.49$_{\pm 0.70}$	**94.40**$_{\pm 2.21}$	**92.19**$_{\pm 0.87}$	100.00$_{\pm 1.40}$	**97.58**$_{\pm 0.28}$	**95.85**$_{\pm 0.37}$	**+3.29**
KnOTS	89.29	94.08	89.67	83.63	100.81	**97.37**	92.47	-
KnOTS + ENMP	**95.11**$_{\pm 0.37}$	**97.48**$_{\pm 1.22}$	**96.60**$_{\pm 0.89}$	**95.97**$_{\pm 1.36}$	**101.34**$_{\pm 1.23}$	97.24$_{\pm 0.44}$	**97.29**$_{\pm 0.29}$	**+4.82**
CoreSpace	95.84	**95.74**	89.25	83.97	**102.42**	97.86	94.18	-
CoreSpace + ENMP	**97.56**$_{\pm 0.02}$	93.37$_{\pm 0.79}$	**96.00**$_{\pm 0.71}$	**93.69**$_{\pm 1.25}$	101.34$_{\pm 0.46}$	**98.39**$_{\pm 0.46}$	**96.73**$_{\pm 0.13}$	**+2.55**

Table 1: Performance comparison on NLP benchmark. We report normalized accuracy (%) as mean ± std over 3 runs. See Appendix D.1 for absolute accuracy. Δ denotes the improvement in average accuracy. **Bold** marks the best result per group. ENMP consistently outperforms all baselines, including advanced alignment methods.

Method	Cars	DTD	EuroSAT	GTSRB	MNIST	RESISC	SUN397	SVHN	Avg.	Δ
TA	**81.99**	73.72	49.31	**42.17**	53.01	71.48	95.39	41.22	63.54	-
TA + ENMP	81.60$_{\pm 0.65}$	**75.18**$_{\pm 0.75}$	**55.02**$_{\pm 0.47}$	41.19$_{\pm 1.49}$	**58.08**$_{\pm 1.96}$	**71.85**$_{\pm 0.23}$	**96.09**$_{\pm 0.20}$	**42.63**$_{\pm 0.96}$	**65.21**$_{\pm 0.34}$	**+1.67**
TIES	**82.60**	73.17	50.77	36.68	57.48	69.67	95.35	42.97	63.59	-
TIES + ENMP	81.53$_{\pm 0.40}$	**74.24**$_{\pm 0.92}$	**58.16**$_{\pm 2.22}$	**37.93**$_{\pm 0.53}$	**65.58**$_{\pm 0.73}$	**70.48**$_{\pm 0.22}$	**95.93**$_{\pm 0.53}$	**46.82**$_{\pm 2.40}$	**66.33**$_{\pm 0.34}$	**+2.74**
DARE	**82.43**	72.99	49.91	37.65	56.71	69.74	95.10	44.50	63.63	-
DARE + ENMP	82.04$_{\pm 0.22}$	**75.54**$_{\pm 0.55}$	**58.89**$_{\pm 2.01}$	**39.29**$_{\pm 0.42}$	**66.90**$_{\pm 0.32}$	**70.92**$_{\pm 0.47}$	**96.11**$_{\pm 0.12}$	**47.58**$_{\pm 1.50}$	**67.16**$_{\pm 0.04}$	**+3.53**
TSV	**83.59**	75.45	52.60	44.85	59.53	73.39	95.25	49.29	66.74	-
TSV + ENMP	82.75$_{\pm 0.46}$	**75.48**$_{\pm 1.83}$	**62.56**$_{\pm 3.81}$	**46.46**$_{\pm 0.86}$	**69.49**$_{\pm 1.76}$	**73.90**$_{\pm 0.90}$	**96.22**$_{\pm 0.42}$	**53.12**$_{\pm 0.55}$	**70.00**$_{\pm 0.21}$	**+3.26**
KnOTS	82.74	72.63	47.06	44.14	61.59	71.25	93.58	49.22	65.28	-
KnOTS + ENMP	**82.98**$_{\pm 0.35}$	**75.64**$_{\pm 0.16}$	**55.06**$_{\pm 0.72}$	**44.76**$_{\pm 1.79}$	**80.46**$_{\pm 1.53}$	**71.38**$_{\pm 1.62}$	**95.41**$_{\pm 0.74}$	**60.91**$_{\pm 2.81}$	**70.82**$_{\pm 0.31}$	**+5.54**
CoreSpace	82.89	**85.03**	53.16	**84.30**	71.00	84.34	97.51	53.55	76.47	-
CoreSpace + ENMP	**84.14**$_{\pm 0.12}$	82.54$_{\pm 2.79}$	**67.15**$_{\pm 1.58}$	81.40$_{\pm 1.55}$	**76.28**$_{\pm 1.26}$	**84.98**$_{\pm 0.64}$	**97.80**$_{\pm 0.46}$	**56.06**$_{\pm 2.11}$	**78.79**$_{\pm 0.22}$	**+2.32**

Table 2: Results on Vision benchmarks (ViT-B/32). We report normalized accuracy (%) as mean ± std over 3 runs. See Appendix D.2 for absolute accuracy. Δ denotes the improvement in average accuracy. **Bold** marks the best result per group. ENMP demonstrates robust improvements across diverse visual recognition tasks.

icated experts. We report both the individual scores and the benchmark average to highlight task-specific variances and global trends.

Models and Architectures. We conduct experiments using two distinct architectures to demonstrate the generality of our approach across modalities: *(1) Large Language Models*: We employ `Llama-3-8B` (Dubey et al., 2024), a representative decoder-only model, to evaluate performance on NLI tasks. *(2) Vision Encoders*: For cross-modal validation, we use the `ViT-B/32` variant of the CLIP vision encoder (Radford et al., 2021).

Baselines. We validate the efficacy of ENMP by integrating it with representative state-of-the-art parameter merging methods, including Task Arithmetic (TA) (Ilharco et al., 2023), TIES-Merging (Yadav et al., 2023), DARE (Yu et al., 2024), TSV (Gargiulo et al., 2025), KnOTS (Stoica et al., 2025), and CoreSpace (Panariello et al., 2025). Detailed formulations of these baselines are provided in Appendix C.1. By treating these methods as foundational baselines, we demonstrate that ENMP serves as a versatile plug-and-play module that consistently enhances their performance by effectively locating and removing negative modules.

Implementation Details. To ensure a rigorous comparison, we use the pre-trained LoRA checkpoints provided by Stoica et al. (2025). Consistent with existing studies, we apply LoRA adapters to all weight matrices in the attention mechanism (`q_proj`, `k_proj`, `v_proj`, `out_proj`), setting the rank $r = 16$ and scaling factor $\alpha = 16$. For the optimization phase, we use the same held-out validation sets as Panariello et al. (2025). The pruning mask $\mathbf{m}$ is optimized via CMA-ES with a population size $N_{pop} = 16$ for 60 generations. Unless otherwise stated, we configure the search with an initial step size $\sigma = 0.5$ and a maximum pruning ratio $k = 0.2$. Detailed hyperparameter configurations are provided in Appendix C.2. All experiments are

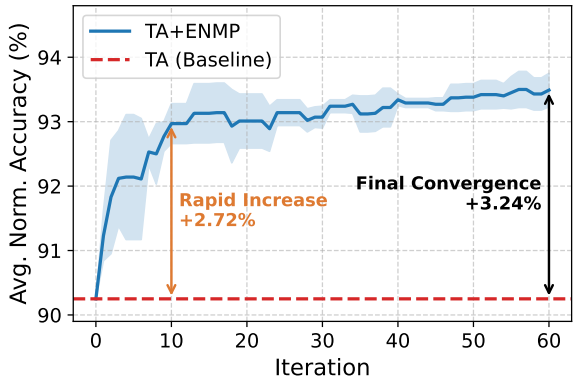

Figure 4: Optimization trajectory of ENMP. We track the test set average normalized accuracy of the pruning mask selected based on validation performance at each generation. The shaded region represents the standard deviation across three independent runs.

conducted on 8 NVIDIA RTX 4090 GPUs, where the candidate evaluations per CMA-ES generation are distributed across GPUs in parallel.

4.2 Main Results

We evaluate the proposed framework on both NLP and Vision benchmarks to verify its effectiveness and generalizability.

Performance on NLP Benchmark. Table 1 presents the comprehensive evaluation results on the NLP benchmark. The empirical evidence strongly supports our hypothesis that selectively removing negative modules enhances the performance of merged language models. Detailed observations are discussed below.

(1) Universal enhancement. ENMP consistently improves performance across all baselines, regardless of the underlying merging strategy. It provides notable gains for simple methods (e.g., +3.24% for TA) while propelling advanced baselines like KnOTS to a new state-of-the-art accuracy of 97.29%. This universality indicates that negative modules are a pervasive bottleneck in LoRA merging, spanning from element-wise aggregation to spectral alignment methods.

(2) Synergy with sparsification methods. We observe substantial improvements when integrating ENMP with parameter-level sparsification methods. Specifically, it boosts TIES by +6.40% and DARE by +6.97%. These gains confirm that while fine-grained pruning reduces intra-module redundancy, it neglects inter-module interference. ENMP effectively alleviates interference at the modular level.

(3) Critical recovery on sensitive tasks. Notably, the performance recovery on QNLI is particularly striking. While baseline methods like TIES and DARE suffer from severe performance degradation on this dataset, ENMP achieves a remarkable gain of over +20%, restoring the normalized accuracy to approximately 95%. This drastic improvement suggests that task interference is not uniformly distributed but can be catastrophic for specific sensitive tasks. It demonstrates that structural conflicts can be effectively resolved by physically removing the specific modules that induce negative transfer.

Cross-Modal Generalizability. To further verify that our method addresses the fundamental problem of module interference rather than overfitting to specific language architectures, we extend our evaluation to the vision domain (Table 2). Consistent with the NLP findings, ENMP yields robust improvements across diverse image recognition tasks (e.g., +5.54% for KnOTS). These results indicate that the phenomenon of negative modules is pervasive and our framework is modality-agnostic.

4.3 Search Efficiency

The search efficiency of the proposed framework directly determines its practical viability. Using TA as the representative baseline, we investigate the convergence behavior of ENMP to evaluate how quickly it finds effective pruning masks. In Figure 4, we plot the test set performance of the candidate mask that achieves the highest accuracy on the validation set at each generation. As illustrated, the optimization trajectory exhibits a steep accuracy ascent during the early phase—the method achieves a rapid performance boost of +2.72% within the first 10 iterations (approximately 23 minutes on our experimental setup), indicating that the evolutionary algorithm efficiently identifies the most significant negative modules. The complete 60-generation search converges to a final gain of +3.24% in approximately 2.3 hours, suggesting that the majority of the gain is captured early in the search. Notably, the test set accuracy improves in tandem with the optimization on the validation set, indicating that ENMP navigates the combinatorial search space without overfitting to the validation data.

4.4 Synergy with Subspace Alignment

We investigate the interaction between ENMP and subspace alignment methods (KnOTS (Stoica et al., 2025), CoreSpace (Panariello et al., 2025)) by comparing two execution orders: *Align-then-Prune* and

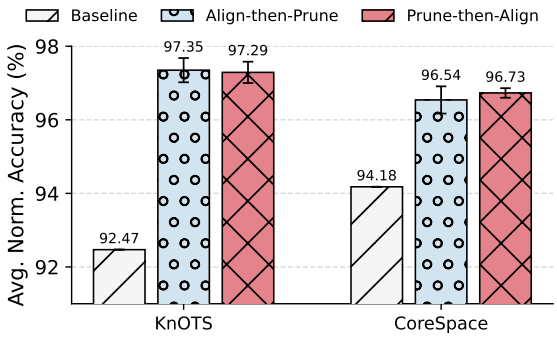

Figure 5: Synergy between ENMP and Subspace Alignment. *Prune-then-Align* consistently outperforms or matches the reverse order by preventing negative modules from polluting the shared subspace.

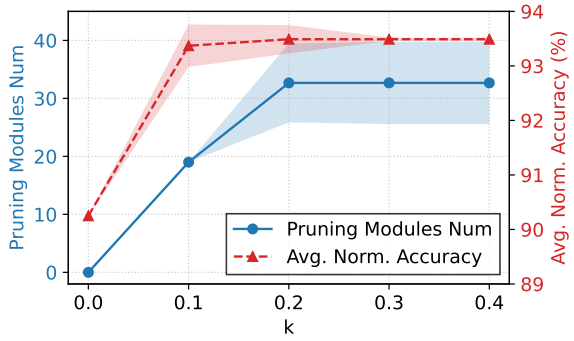

Figure 6: Sensitivity to Maximum Pruning Ratio (k). Results (mean ± std over three runs) show increasing k leads to saturated module removal (left) and stable accuracy (right), confirming ENMP's *adaptive sparsity*.

Method	**Avg. Norm. Acc. (%)**	**Δ**
Task Arithmetic (TA)	90.25	-
TA + Random Pruning	89.10 ± 1.19	-1.15
TA + ENMP (Ours)	**93.49** ± 0.26	**+3.24**

Table 3: Comparison with Random Pruning on the NLP benchmark. Δ denotes the change in average normalized accuracy relative to the Task Arithmetic baseline.

Samples / Task	**64**	**128**	**256**	**Full Val**
TA + ENMP (Ours)	91.17	91.86	92.24	93.49
Δ (vs. TA)	+0.92	+1.61	+1.99	+3.24

Table 4: Sensitivity to Validation Set Size on NLP Benchmark. Average normalized accuracy (%) with varying numbers of samples per task. Δ denotes improvement over the TA baseline (90.25%).

Prune-then-Align. In Figure 5, *Prune-then-Align* generally yields superior performance, boosting CoreSpace by +0.19%. We attribute this to the sensitivity of subspace construction. In *Align-then-Prune*, negative modules participate in the basis construction via SVD, "polluting" the shared subspace with interference directions. Conversely, *Prune-then-Align* derives the subspace solely from constructive modules, ensuring a cleaner reference basis for alignment. While KnOTS shows robustness to ordering (around 97.3%), early pruning remains theoretically preferable for reducing the computational overhead of the SVD step.

5 Ablation Study

We validate the necessity of evolutionary search and the robustness to pruning constraints, employing Task Arithmetic (TA) as the representative baseline to analyze the impact of ENMP.

5.1 Necessity of Evolutionary Search

To investigate whether performance gains stem from precise pruning or mere sparsity, we compare ENMP with a *Random Pruning* baseline (matching approximately 16.7% sparsity resulting from ENMP's search) on the NLP benchmark. As shown in Table 3, random pruning degrades accuracy to 89.10% representing a 1.15% drop from the TA baseline, with high variance (±1.19%), indicating that interference is non-uniformly distributed and blind removal risks discarding essential knowledge. In contrast, ENMP achieves 93.49% with negligible variance (±0.26%), confirming that the advantage derives from the precise localization of interfering components rather than sparsity alone.

5.2 Robustness to Pruning Constraints

We empirically analyze ENMP's sensitivity to the maximum pruning ratio k (0.0 to 0.4) in Figure 6. Even a modest relaxation from k=0.0 to k=0.1 triggers a sharp accuracy boost (from 90.25% to 93.37%), indicating that removing even a few negative modules yields substantial benefits. For looser constraints ($k \geq 0.2$), ENMP exhibits *adaptive sparsity*: the number of removed modules plateaus at approximately 32 rather than blindly filling the allowed budget. This stability suggests that k serves merely as a flexible upper bound; providing a sufficient margin allows the algorithm to autonomously converge to the optimal sparsity level without fine-grained tuning.

5.3 Sensitivity to Validation Set Size

To evaluate how much validation data ENMP requires for reliable fitness evaluation, we vary the number of samples per task from 64 to the full

validation set on the NLP benchmark. As shown in Table 4, ENMP exhibits strong data efficiency. With as few as 64 samples per task, it already achieves 91.17% accuracy, surpassing the TA baseline (90.25%) by +0.92%. Performance further improves consistently as more data becomes available. This result suggests that the interference signal captured by ENMP is structural rather than statistical. Even under high sampling variance, ENMP can consistently identify negative modules, reducing the reliance on large-scale external validation sets.

6 Conclusion

In this work, we challenge the idealized assumption in LoRA merging that all task-specific parameters contribute constructively to the merged model. We uncover the existence of *negative modules*, specific LoRA layers that introduce interference and degrade multi-task performance. To address this, we propose ENMP, a novel framework that utilizes the evolutionary search algorithm to solve negative module pruning as a combinatorial optimization problem. Extensive experiments across NLP and vision benchmarks demonstrate that ENMP serves as a versatile, plug-and-play enhancement, consistently boosting the performance of existing state-of-the-art merging algorithms (including TIES, KnOTS, and CoreSpace). Our findings highlight that in the era of massive model composability, subtractive mechanisms are just as critical as additive ones for achieving optimal model merging.

Limitations

While ENMP effectively enhances model merging performance, we identify two primary limitations to be addressed in future work.

Offline Computational Overhead. Unlike instant merging methods such as Task Arithmetic, ENMP involves an iterative evolutionary search. Although this incurs a one-time computational cost during the merging phase, it is important to note that the merged model retains the exact architecture of the backbone, incurring *zero additional overhead during inference*. Our experiments show efficient convergence within 60 generations (Figure 4). However, applying this search to extremely large-scale settings (e.g., 70B models with hundreds of tasks) remains a challenge that may necessitate more sample-efficient optimization strategies.

Requirement for Validation Data. ENMP utilizes a validation set ($\mathcal{D}_{val}$) to compute fitness scores for the evolutionary algorithm. While this dependency allows for precise localization of negative modules, it assumes the availability of representative labeled data. In scenarios strictly requiring data-free merging, this prerequisite constitutes a constraint. Nevertheless, given that many state-of-the-art baselines also benefit from validation data for hyperparameter tuning (e.g., scaling factors), we consider this a justifiable trade-off for the significant performance gains observed.

Acknowledgments

This work was supported by the National Natural Science Foundation of China (62576305, 62506330), the Ningbo Project of Leading Youth Talents for S&T Innovation (2025QL052), and the Zhejiang Provincial Natural Science Foundation of China (LQN26F020007).

References

Samuel Bowman, Gabor Angeli, Christopher Potts, and Christopher D Manning. 2015. A large annotated corpus for learning natural language inference. In *Proceedings of the 2015 conference on empirical methods in natural language processing*, pages 632–642.

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

Jiho Choi, Donggyun Kim, Chanhyuk Lee, and Seunghoon Hong. 2025. Revisiting weight averaging for model merging. *Preprint*, arXiv:2412.12153.

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

MohammadReza Davari and Eugene Belilovsky. 2024. Model breadcrumbs: Scaling multi-task model merging with sparse masks. In *European Conference on Computer Vision*, pages 270–287. Springer.

Tim Dettmers, Artidoro Pagnoni, Ari Holtzman, and Luke Zettlemoyer. 2023. Qlora: Efficient finetuning of quantized llms. *Advances in neural information processing systems*, 36:10088–10115.

Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, A. Schelten, Amy Yang, Angela Fan, Anirudh Goyal, Anthony S. Hartshorn, Aobo Yang,

Archi Mitra, A. Sravankumar, A. Korenev, Arthur Hinsvark, Arun Rao, Aston Zhang, and 510 others. 2024. The llama 3 herd of models.

Antonio Andrea Gargiulo, Donato Crisostomi, Maria Sofia Bucarelli, Simone Scardapane, Fabrizio Silvestri, and Emanuele Rodola. 2025. Task singular vectors: Reducing task interference in model merging. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pages 18695–18705.

Nikolaus Hansen. 2016. The cma evolution strategy: A tutorial. *arXiv preprint arXiv:1604.00772*.

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

Neil Houlsby, Andrei Giurgiu, Stanislaw Jastrzebski, Bruna Morrone, Quentin De Laroussilhe, Andrea Gesmundo, Mona Attariyan, and Sylvain Gelly. 2019. Parameter-efficient transfer learning for nlp. In *International conference on machine learning*, pages 2790–2799. PMLR.

Edward J Hu, yelong shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. 2022. LoRA: Low-rank adaptation of large language models. In *International Conference on Learning Representations*.

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

Tushar Khot, Ashish Sabharwal, and Peter Clark. 2018. Scitail: A textual entailment dataset from science question answering. In *Proceedings of the AAAI conference on artificial intelligence*, volume 32.

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

Yann LeCun and Corinna Cortes. 2010. MNIST handwritten digit database.

Brian Lester, Rami Al-Rfou, and Noah Constant. 2021. The power of scale for parameter-efficient prompt tuning. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics.

Xiang Lisa Li and Percy Liang. 2021. Prefix-tuning: Optimizing continuous prompts for generation. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 4582–4597.

Daniel Marczak, Simone Magistri, Sebastian Cygert, Bartłomiej Twardowski, Andrew D. Bagdanov, and Joost van de Weijer. 2025. No task left behind: Isotropic model merging with common and task-specific subspaces. In *Forty-second International Conference on Machine Learning*.

Marco Marelli, Stefano Menini, Marco Baroni, Luisa Bentivogli, Raffaella Bernardi, and Roberto Zamparelli. 2014. A SICK cure for the evaluation of compositional distributional semantic models. In *Proceedings of the Ninth International Conference on Language Resources and Evaluation (LREC'14)*, pages 216–223, Reykjavik, Iceland. European Language Resources Association (ELRA).

Yuval Netzer, Tao Wang, Adam Coates, A. Bissacco, Bo Wu, and A. Ng. 2011. Reading digits in natural images with unsupervised feature learning.

OpenAI, Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, Red Avila, Igor Babuschkin, Suchir Balaji, Valerie Balcom, Paul Baltescu, Haiming Bao, Mohammad Bavarian, Jeff Belgum, and 262 others. 2024. GPT-4 technical report. *Preprint*, arXiv:2303.08774.

Aniello Panariello, Daniel Marczak, Simone Magistri, Angelo Porrello, Bartłomiej Twardowski, Andrew D. Bagdanov, Simone Calderara, and Joost van de Weijer. 2025. Accurate and efficient low-rank model merging in core space. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*.

Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, Gretchen Krueger, and Ilya Sutskever. 2021. Learning transferable visual models from natural language supervision. In *Proceedings of the 38th International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, pages 8748–8763. PMLR.

Victor Sanh, Albert Webson, Colin Raffel, Stephen Bach, Lintang Sutawika, Zaid Alyafeai, Antoine Chaffin, Arnaud Stiegler, Arun Raja, Manan Dey, M Saiful Bari, Canwen Xu, Urmish Thakker, Shanya Sharma Sharma, Eliza Szczechla, Taewoon Kim, Gunjan Chhablani, Nihal Nayak, Debajyoti Datta, and 21 others. 2022. Multitask prompted training enables zero-shot task generalization. In *International Conference on Learning Representations*.

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

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

Anke Tang, Li Shen, Yong Luo, Yibing Zhan, Han Hu, Bo Du, Yixin Chen, and Dacheng Tao. 2024. Parameter-efficient multi-task model fusion with partial linearization. In *The Twelfth International Conference on Learning Representations*.

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

Jason Wei, Maarten Bosma, Vincent Zhao, Kelvin Guu, Adams Wei Yu, Brian Lester, Nan Du, Andrew M. Dai, and Quoc V Le. 2022. Finetuned language models are zero-shot learners. In *International Conference on Learning Representations*.

Adina Williams, Nikita Nangia, and Samuel Bowman. 2018. A broad-coverage challenge corpus for sentence understanding through inference. In *Proceedings of the 2018 conference of the North American chapter of the association for computational linguistics: human language technologies, volume 1 (long papers)*, pages 1112–1122.

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

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

An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, Chujie Zheng, Dayiheng Liu, Fan Zhou, Fei Huang, Feng Hu, Hao Ge, Haoran Wei, Huan Lin, Jialong Tang, and 41 others. 2025. Qwen3 technical report. *Preprint*, arXiv:2505.09388.

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

Qingru Zhang, Minshuo Chen, Alexander Bukharin, Pengcheng He, Yu Cheng, Weizhu Chen, and Tuo Zhao. 2023. Adaptive budget allocation for parameter-efficient fine-tuning. In *The Eleventh International Conference on Learning Representations*.

A Evolutionary Search Details

In this section, we provide the implementation details of the evolutionary search strategy employed in our framework. Algorithm 1 outlines the complete optimization process, including the initialization of the CMA-ES strategy, the generation of candidate pruning masks via latent variable mapping, and the iterative update mechanism driven by validation performance.

Algorithm 1 Evolutionary Search with CMA-ES

Require: Pre-trained LoRA experts $\mathcal{T} = \{\Delta\mathbf{W}_1, \ldots, \Delta\mathbf{W}_T\}$, Validation Set $\mathcal{D}_{val}$, Generations G, Pop. Size N_{pop}.

Initialize: Mean $\boldsymbol{\mu}^{(0)} \leftarrow -\mathbf{1}$, Covariance $\mathbf{C}^{(0)} \leftarrow \mathbf{I}$, Step size σ.
for generation $g = 0$ **to** $G - 1$ **do**
  Sample N_{pop} latent vectors:
    $\mathbf{z}_i \sim \mathcal{N}(\boldsymbol{\mu}^{(g)}, (\sigma^{(g)})^2\mathbf{C}^{(g)})$
  for $i = 1$ **to** N_{pop} **do**
    Masking: $\mathbf{m}_i^{\text{flat}} \leftarrow \text{TopK-Map}(\mathbf{z}_i)$ *(Eq. 6)*
    Reshape: $\mathbf{m}_i \leftarrow \mathbf{m}_i^{\text{flat}}$
    Pruning: $\mathcal{S}_i \leftarrow \{\Delta\mathbf{W}_t^{(l)} \mid m_{i,l,t} = 0\}$
    Merging: $\mathbf{W}_{\text{merged}} \leftarrow \Phi(\mathcal{S}_i)$ *(Eq. 3)*
    Eval: $\mathcal{F}_i \leftarrow \mathcal{M}(\mathbf{W}_{\text{merged}}; \mathcal{D}_{val})$
    Update best mask $\mathbf{m}^*$ if $\mathcal{F}_i > \mathcal{F}_{best}$.
  end for
  Update $\boldsymbol{\mu}^{(g+1)}, \mathbf{C}^{(g+1)}$ via CMA-ES rule using $\{\mathcal{F}_i\}$.
end for
Output: Optimal merged model $\mathbf{W}_{\text{final}}$ using $\mathbf{m}^*$.

Layer	SNLI	MNLI	SICK	QNLI	RTE	SCITAIL
0	+0.5025	+0.6965	-0.0165	+0.0249	+0.0759	+0.1954
1	+0.2244	+0.4301	+0.0106	+0.1585	+0.1909	+0.1493
2	+0.1694	+0.2833	+0.0237	+0.1357	+0.0424	+0.1808
3	+0.0374	+0.0464	+0.0007	-0.0441	+0.0121	+0.0233
4	-0.0602	+0.0643	+0.1127	-0.0450	+0.0152	+0.1436
5	+0.2331	+0.1672	+0.1728	-0.1481	+0.0042	+0.0642
6	+0.1073	+0.2377	+0.0580	-0.1480	+0.0220	+0.0203
7	+0.0433	+0.1108	+0.1214	-0.1064	-0.0652	+0.0577
8	+0.0660	+0.4925	+0.1411	-0.2709	+0.1860	+0.0645
9	+0.0596	+0.1534	+0.1944	-0.2721	+0.1591	+0.0412
10	+0.0809	+0.1962	+0.1464	-0.1141	+0.1425	+0.0013
11	+0.4212	+0.2474	+0.2655	-0.4792	+0.2823	+0.4918
12	+0.5209	+0.7198	+0.2745	-1.2861	+0.6948	+0.1298
13	+0.7976	+1.5032	+0.2018	-0.3989	+0.6677	+0.1148
14	+0.4281	+0.2933	+0.2083	-0.2945	+0.7995	+0.0798
15	+0.4847	+0.9975	+0.1951	-0.4015	+1.1883	+0.0847
16	+0.1145	+0.4264	+0.1804	-0.6437	+0.6445	+0.0982
17	+0.2521	+0.1381	+0.1607	-0.2474	+0.0949	+0.1409
18	+0.0197	+0.1548	+0.1434	+0.0318	+0.1374	+0.1658
19	+0.1997	+0.0495	+0.2051	+0.0114	+0.1211	+0.1487
20	+0.1941	+0.2056	+0.1798	+0.0813	+0.0346	+0.0439
21	+0.0687	-0.0392	-0.0301	-0.0865	+0.1521	+0.1051
22	+0.0138	-0.1376	+0.2211	-0.0590	+0.1302	+0.0217
23	+0.1581	+0.1541	+0.2085	+0.1425	+0.1657	+0.1765
24	+0.2243	-0.0065	+0.2044	+0.0209	+0.2088	+0.1654
25	+0.0222	+0.1643	+0.1607	+0.0249	+0.1457	+0.0544
26	+0.0082	+0.0137	-0.0043	+0.0005	+0.2046	-0.0121
27	+0.2817	+0.2123	+0.1193	+0.3553	+0.1595	+0.1639
28	+0.2290	+0.0965	+0.1664	+0.0615	+0.1908	+0.1506
29	-0.0474	-0.1692	+0.1511	+0.0747	+0.1752	-0.0420
30	+0.0793	+0.5465	+0.1400	+0.0873	+0.1694	+0.0895
31	-0.3368	-0.4654	+0.0018	-0.3027	+0.1696	-0.2022

Table 5: Detailed performance impact of *leave-one-out* pruning. Background color intensity indicates the magnitude of impact (red for increase, blue for decrease).

B Quantitative Analysis of Pilot Study

In Figure 1, we visualized the impact of pruning individual LoRA modules via a heatmap. In this section, we provide the specific numerical values corresponding to that analysis. Table 5 lists the change in performance (Δ Acc) when the specific module (layer l, task t) is removed from the aggregation.

C Implementation Details

C.1 Baseline Formulations

In this section, we formulate the baseline methods using a unified notation. Let $\{\Delta\mathbf{W}_t\}_{t=1}^{T}$ denote the set of task-specific LoRA updates. The merged update $\Delta\mathbf{W}_{\text{merged}}$ is computed as follows:

Task Arithmetic (TA) TA (Ilharco et al., 2023) assumes that task vectors are independent and constructive. It computes the merged update as a simple weighted sum:

$$\Delta\mathbf{W}_{\text{TA}} = \lambda \sum_{t=1}^{T} \Delta\mathbf{W}_t, \tag{10}$$

where λ is a scalar hyperparameter scaling the aggregate strength.

TIES-Merging TIES (Yadav et al., 2023) mitigates interference via a "Trim, Elect, and Merge" pipeline. First, it *trims* the task updates to retain only the top-$k\%$ largest magnitude parameters, yielding sparse updates $\Delta\hat{\mathbf{W}}_t$. Second, it resolves conflicts by *electing* a unified sign vector $\mathbf{s}^*$ based on the total magnitude of parameters. Finally, it computes a *disjoint mean* by averaging only the values that align with the elected sign:

$$\Delta\mathbf{W}_{\text{TIES}} = \lambda \cdot \text{Mean}\left(\{\Delta\hat{\mathbf{W}}_t \mid \text{sgn}(\Delta\hat{\mathbf{W}}_t) = \mathbf{s}^*\}_{t=1}^{T}\right), \tag{11}$$

where the mean ignores zero entries pruned in the trimming step.

DARE DARE (Yu et al., 2024) exploits the extreme redundancy in delta parameters. Operating on the premise that most fine-tuned updates can be removed without performance loss, it employs a stochastic "Drop And REscale" strategy. For each task t, DARE generates a binary mask $\mathbf{M}_t \sim \text{Bernoulli}(1-p)$ to randomly drop a fraction p of the elements, and rescales the surviving parameters to preserve the expected value of the updates:

$$\Delta\tilde{\mathbf{W}}_t = \frac{1}{1-p}(\Delta\mathbf{W}_t \odot \mathbf{M}_t). \tag{12}$$

These sparsified updates are then typically aggregated via summation or combined with other techniques (e.g., TIES).

Task Singular Vectors (TSV) TSV (Gargiulo et al., 2025) approaches model merging by analyzing the geometric structure of layer-wise weight updates. Recognizing that task matrices are inherently low-rank, TSV decomposes each update via Singular Value Decomposition (SVD) and retains only the most significant singular components, $\Delta\mathbf{W}_t \approx \mathbf{U}_t\mathbf{\Sigma}_t\mathbf{V}_t^\top$. To resolve task interference, it concatenates the singular vectors across all tasks and applies a whitening transformation (formulated as an orthogonal Procrustes problem) to decorrelate them. The merged update is reconstructed from these orthogonalized bases without requiring additional scaling coefficients:

$$\Delta\mathbf{W}_{\text{TSV}} = \mathbf{U}_\perp\mathbf{\Sigma}_{\text{block}}\mathbf{V}_\perp^\top, \tag{13}$$

where $\mathbf{\Sigma}_{\text{block}} = \text{diag}(\mathbf{\Sigma}_1, \ldots, \mathbf{\Sigma}_T)$, and $\mathbf{U}_\perp$ (similarly for $\mathbf{V}_\perp$) is the orthogonal matrix that minimizes the projection error $\|\mathbf{U}_{\text{cat}} - \mathbf{U}_\perp\|_F$ relative to the concatenated singular vectors $\mathbf{U}_{\text{cat}}$.

KnOTS KnOTS (Stoica et al., 2025) addresses the misalignment issue in LoRA-finetuned models by projecting task-specific updates into a shared geometric subspace. By attributing the poor mergeability of LoRA models to their updates residing in disparate subspaces, KnOTS concatenates the weight updates from all tasks layer-wise and applies Singular Value Decomposition (SVD) to extract a common orthonormal basis $\mathbf{U}$ and a scaling matrix $\mathbf{\Sigma}$. This decomposition isolates task-specific variations into the right singular vectors $\mathbf{V}_t$, which are aligned to the shared basis. Standard merging algorithms (such as TIES or Task Arithmetic) are then applied exclusively to these aligned vectors to produce a unified component $\mathbf{V}_{\text{merged}}$. The final merged update is reconstructed by projecting back via the shared basis:

$$\Delta\mathbf{W}_{\text{merged}} = \mathbf{U}\mathbf{\Sigma}\mathbf{V}_{\text{merged}}^\top, \tag{14}$$

where $\mathbf{V}_{\text{merged}} = \text{Merge}(\{\mathbf{V}_1, \ldots, \mathbf{V}_T\})$. This procedure effectively aligns the representation spaces of disjoint LoRA models without requiring additional data or gradient-based optimization. For the main experiments, we follow the best practices from the original paper, utilizing TIES for the merging stage.

Method	NLP (Llama-3)			Vision (ViT-B/32)		
	λ	Top-k (%)	p	λ	Top-k (%)	p
Task Arithmetic (TA)	0.3	-	-	0.1	-	-
TIES-Merging	1.2	80	-	0.3	40	-
DARE-TIES	1.1	80	0.1	0.3	20	0.1
TSV	0.6	-	-	0.3	-	-
KnOTS (w/ TIES)	1.1	90	-	0.6	90	-
CoreSpace	0.5	-	-	0.9	-	-

Table 6: Detailed hyperparameter configurations for baselines across NLP and CV benchmarks. λ denotes the scaling factor (consistent with Eq. 2), Top-k indicates the percentage of parameters retained (density), and p represents the drop rate for DARE.

Method	SNLI	MNLI	SICK	QNLI	RTE	SciTail	Avg.	Δ
Individual Task	92.50	90.31	91.58	94.49	89.86	96.52	92.54	-
TA	86.57	**86.06**	80.60	64.94	89.13	93.32	83.44	-
TA + ENMP	**88.74** ± 0.26	85.18 ± 0.39	**83.88** ± 1.63	**75.97** ± 0.57	**91.06** ± 0.42	**93.98** ± 0.12	**86.47** ± 0.25	**+3.03**
TIES	87.74	87.34	73.99	67.60	**89.86**	92.66	83.20	-
TIES + ENMP	**89.33** ± 0.25	**88.71** ± 0.56	**84.92** ± 0.75	**89.72** ± 0.63	89.37 ± 0.42	**93.04** ± 0.40	**89.18** ± 0.42	**+5.98**
DARE	87.41	87.48	71.14	68.44	87.68	92.80	82.49	-
DARE + ENMP	**88.88** ± 0.54	**88.30** ± 0.64	**84.56** ± 0.61	**89.71** ± 0.70	**88.65** ± 1.11	**93.84** ± 0.25	**88.99** ± 0.12	**+6.50**
TSV	88.21	**85.91**	81.37	72.59	**91.30**	94.17	85.59	-
TSV + ENMP	**89.22** ± 0.69	85.33 ± 0.63	**86.45** ± 2.03	**87.11** ± 0.82	89.86 ± 1.25	**94.18** ± 0.26	**88.69** ± 0.33	**+3.10**
KnOTS	82.59	84.96	82.12	79.02	90.58	**93.98**	85.54	-
KnOTS + ENMP	**87.98** ± 0.35	**88.03** ± 1.10	**88.47** ± 0.81	**90.68** ± 1.28	**91.06** ± 1.10	93.85 ± 0.43	**90.01** ± 0.27	**+4.47**
CoreSpace	88.65	**86.46**	81.74	79.34	**92.03**	94.45	87.11	-
CoreSpace + ENMP	**90.24** ± 0.02	84.32 ± 0.71	**87.92** ± 0.64	**88.53** ± 1.19	91.06 ± 0.42	**94.97** ± 0.45	**89.51** ± 0.13	**+2.40**

Table 7: Absolute Accuracy (%) on NLP Benchmarks. We report the mean accuracy ± standard deviation across 3 seeds. Δ denotes the average improvement over the corresponding baseline. The *Individual Task* row represents the performance of models fine-tuned on single tasks.

Method	Cars	DTD	EuroSAT	GTSRB	MNIST	RESISC45	SUN397	SVHN	Avg.	Δ
Individual Task	74.00	58.30	99.00	92.70	99.30	88.40	64.50	96.20	84.05	-
TA	**60.67**	42.98	48.81	**39.09**	52.64	63.19	61.53	39.65	51.07	-
TA + ENMP	60.39 ± 0.48	**43.83** ± 0.44	**54.47** ± 0.46	38.18 ± 1.38	**57.68** ± 1.95	**63.51** ± 0.19	**61.98** ± 0.13	**41.02** ± 0.92	**52.63** ± 0.31	**+1.56**
TIES	**61.12**	42.66	50.26	34.01	57.08	61.59	61.50	41.34	51.19	-
TIES + ENMP	60.33 ± 0.30	**43.28** ± 0.54	**57.58** ± 2.20	**35.16** ± 0.49	**65.12** ± 0.72	**62.31** ± 0.20	**61.88** ± 0.34	**45.04** ± 2.31	**53.84** ± 0.26	**+2.65**
DARE	**61.00**	42.55	49.41	34.90	56.31	61.65	61.34	42.81	51.25	-
DARE + ENMP	60.71 ± 0.16	**44.04** ± 0.32	**58.30** ± 1.99	**36.42** ± 0.39	**66.43** ± 0.31	**62.69** ± 0.42	**61.99** ± 0.08	**45.78** ± 1.44	**54.54** ± 0.04	**+3.29**
TSV	**61.85**	44.00	52.07	41.58	59.11	64.87	61.44	47.42	54.04	-
TSV + ENMP	61.23 ± 0.34	**44.01** ± 1.06	**61.94** ± 3.77	**43.07** ± 0.79	**69.01** ± 1.74	**65.33** ± 0.80	**62.06** ± 0.27	**51.10** ± 0.53	**57.22** ± 0.20	**+3.18**
KnOTS	61.23	42.34	46.59	40.91	61.16	62.98	60.36	47.35	52.87	-
KnOTS + ENMP	**61.40** ± 0.25	**44.10** ± 0.09	**54.50** ± 0.72	**41.49** ± 1.66	**79.90** ± 1.52	**63.09** ± 1.43	**61.54** ± 0.48	**58.60** ± 2.71	**58.08** ± 0.33	**+5.21**
CoreSpace	61.34	49.57	52.63	**78.15**	70.50	74.56	62.90	51.51	62.64	-
CoreSpace + ENMP	**62.26** ± 0.09	48.12 ± 1.62	**66.48** ± 1.56	75.45 ± 1.44	**75.75** ± 1.24	**75.13** ± 0.56	**63.09** ± 0.30	**53.93** ± 2.03	**65.03** ± 0.11	**+2.39**

Table 8: Absolute Accuracy (%) on Vision Benchmarks. The improvement (Δ) indicates the gain of ENMP over the baseline method. The *Individual Task* row represents the performance of models fine-tuned on single tasks.

CoreSpace CoreSpace (Panariello et al., 2025) proposes a computationally efficient framework for merging LoRA-adapted models by operating within a compact, shared geometric subspace. Unlike methods that merge in the high-dimensional parameter space or require costly SVD on full weight updates (e.g., KnOTS), CoreSpace constructs a common alignment basis $(\mathbf{U}_B^{\text{ref}}, \mathbf{V}_A^{\text{ref}})$ by decomposing the concatenated low-rank factors $\mathbf{B}_t$ and $\mathbf{A}_t$ across all tasks. Each task's update is projected into this space to obtain a dense *Core Matrix* $\tilde{\mathbf{M}}_t$, capturing the task-specific transformation without information loss. Merging is performed on these low-dimensional matrices, and the final update is reconstructed via the reference bases:

$$\Delta \mathbf{W}_{\text{Core}} = \mathbf{U}_B^{\text{ref}} \text{Merge}\left(\{\tilde{\mathbf{M}}_t\}_{t=1}^{T}\right) (\mathbf{V}_A^{\text{ref}})^{\top}, \tag{15}$$

where $\tilde{\mathbf{M}}_t = (\mathbf{U}_B^{\text{ref}})^\top \mathbf{B}_t \mathbf{A}_t \mathbf{V}_A^{\text{ref}}$. This approach decouples merging complexity from the model dimension, ensuring scalability while theoretically guaranteeing zero reconstruction error relative to full-space concatenation. For the main experiments, we follow the best practices from the original paper in merging stage, employing TSV for NLP and TSV + Iso-C for the Vision benchmark.

C.2 Hyperparameter Configurations

To ensure fair comparison and reproducibility, we align our baseline settings with prior works. We observe that optimal hyperparameters vary significantly between modalities due to differences in backbone architectures (Llama-3 vs. ViT) and task characteristics. Table 6 details the specific configurations for both NLP and Vision benchmarks.

D Additional Quantitative Results

In Table 1 and Table 2, we reported the Normalized Accuracy to provide a balanced view across tasks with varying difficulty. To ensure transparency and facilitate comparison with future works, we present the **Absolute Accuracy** for all benchmarks in this section.

D.1 NLP Benchmark Results

Table 7 reports the raw accuracy scores for the NLP tasks. As demonstrated in Table 7, our method (ENMP) consistently improves performance across all baselines. It is particularly encouraging to see that even for strong baselines such as TSV, KnOTS, and CoreSpace, which already achieve high base accuracy, ENMP still provides significant further improvements of +3.10%, +4.47%, and +2.40%, respectively.

D.2 Vision Benchmark Results

Table 8 presents the absolute accuracy for the CV tasks using the ViT-B/32 backbone. Similar to the NLP results, ENMP provides a universal boost across all CV baselines. It is particularly effective when combined with subspace-based methods like KnOTS, achieving a remarkable +5.21% improvement. Even for the state-of-the-art method CoreSpace, which already operates in a highly optimized subspace, ENMP still squeezes out an additional +2.39% accuracy, demonstrating its complementary nature to existing subspace-based merging techniques.