Unraveling LoRA Interference: Orthogonal Subspaces for Robust Model Merging

Haobo Zhang
University of Michigan
Ann Arbor, USA
haobozha@umich.edu

Jiayu Zhou
University of Michigan
Ann Arbor, USA
jiayuz@umich.edu

Abstract

Fine-tuning large language models (LMs) for individual tasks yields strong performance but is expensive for deployment and storage. Recent works explore model merging to combine multiple task-specific models into a single multi-task model without additional training. However, existing merging methods often fail for models fine-tuned with low-rank adaptation (LoRA), due to significant performance degradation. In this paper, we show that this issue arises from a previously overlooked interplay between model parameters and data distributions. We propose **O**rthogonal **S**ubspaces for **R**obust model **M**erging (**OSRM**) to constrain the LoRA subspace *prior* to fine-tuning, ensuring that updates relevant to one task do not adversely shift outputs for others. Our approach can seamlessly integrate with most existing merging algorithms, reducing the unintended interference among tasks. Extensive experiments on eight datasets, tested with three widely used LMs and two large LMs, demonstrate that our method not only boosts merging performance but also preserves single-task accuracy. Furthermore, our approach exhibits greater robustness to the hyperparameters of merging. These results highlight the importance of data-parameter interaction in model merging and offer a plug-and-play solution for merging LoRA models.

1 Introduction

Pre-trained language models (LMs) have achieved remarkable success across diverse tasks, with fine-tuning approaches enabling strong downstream performance (Radford et al., 2019; Touvron et al., 2023). However, maintaining a separate fine-tuned model for each task becomes prohibitively expensive in terms of both storage and deployment. While multi-task learning (Zhang and Yang, 2021) attempts to address this issue by training a unified model for multiple tasks, it demands simultaneous access to all task data and high computational overhead, thereby limiting its scalability.

An appealing alternative is model merging, which combines multiple task-specific models into a single multi-task model without further training (Ilharco et al., 2022; Huang et al., 2024; Yadav et al., 2024). Early merging techniques primarily average parameters, often guided by Fisher information (Matena and Raffel, 2022) or inner-product-based metrics (Jin et al., 2022). Another line of work employs task vectors—the difference between pre-trained and fine-tuned model weights—and manipulates them before summation (Yadav et al., 2024; Du et al., 2024; Jiang et al., 2023). Despite these promising developments, merging models that were fine-tuned with low-rank adaptation (LoRA) (Hu et al., 2021), remains challenging and can severely degrade performance (Stoica et al., 2024; Tang et al., 2023a).

We argue that this degradation stems from parameter interference and how each model's parameters interact with out-of-task data. While prior work has focused on preserving orthogonality among task vectors (Ortiz-Jimenez et al., 2023; Gao et al., 2024; Yoshida et al.) or aligning them in a shared space (Stoica et al., 2024), these data-free strategies often overlook the crucial interplay between latent features and parameter updates. Specifically, consider a pre-trained layer W_0 and two sets of learned LoRA blocks $\{B_1, A_1\}$ and $\{B_2, A_2\}$. The merged model is then formulated as $W_m = W_0 + B_1A_1 + B_2A_2$. Given a latent feature vector $\mathbf{h}_1$ from task T_1, the merged model produces $W_m\mathbf{h}_1 = W_1\mathbf{h}_1 + B_2A_2\mathbf{h}_1$, where the first term represents the intended response while the second term is the undesired shift. Notably, existing data-free approaches that focus solely on resolving parameter conflicts are insufficient to mitigate such interference.

In this paper, we propose a novel approach named **OSRM** (**O**rthogonal **S**ubspaces for **R**obust

Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), pages 26459–26472

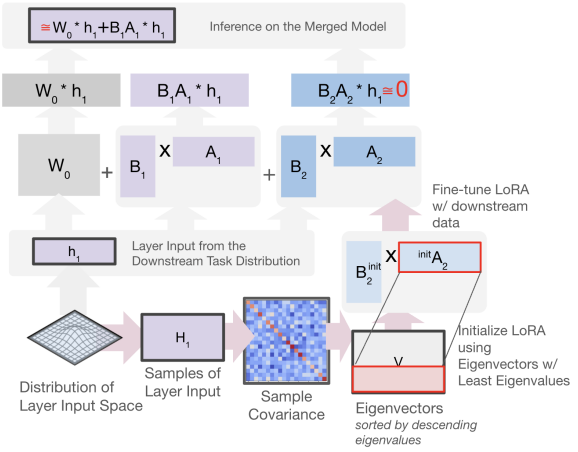


Figure 1: Overview of OSRM, which seeks a data-driven subspace to initiate LoRA fine-tuning and thereby greatly improves model performance when merging multiple LoRA models from different tasks. W_0 is the pre-trained weight. $\{B_i, A_i\}$ are LoRA fine-tuned on the i-th task. Purple: $(W_0 + B_1A_1) * h_1$ is the required output. Light blue: Decompose the sample covariance matrix to initialize A_2. Dark blue: Reduce the output shift induced by B_2A_2.

model **M**erging) that restricts the LoRA subspace *before* fine-tuning, making it largely orthogonal to irrelevant out-of-task data distributions. Concretely, we aim to reduce the interference between data and parameters by minimizing $\|A_2\mathbf{h}_1\|_F$ and derive an analytical solution under the assumption that A_2 has an orthogonal basis. Our method integrates seamlessly with existing merging algorithms and mitigates the unintentional output shifts that arise when multiple LoRA modules are combined (c.f. Fig. 1). We further present practical extensions to ensure robust performance in real-world scenarios.

Extensive experiments on eight datasets using three widely used LMs and two large LMs confirm that our approach consistently outperforms existing merging baselines on multi-task evaluations, while preserving strong single-task performance. Additionally, our empirical results show the robustness of our method against hyperparameters, such as the scaling coefficient, the sample size, the number of tasks, and the choice of learnable blocks. Our findings highlight the importance of considering data-parameter interplay in model merging and demonstrate a generalizable strategy for combining LoRA models more effectively.

2 Related Work

Model Merging. A significant body of work explores merging models that have been fine-tuned independently on different datasets to obtain a unified multi-task model. Ilharco et al. (2022) introduced Task Arithmetic (TA), which defines task vectors for models and combines them using a unified weight. Building on TA, several methods have been proposed to address weight entanglement in task-specific models by aligning them before merging (Gao et al., 2024; Ortiz-Jimenez et al., 2023; Yoshida et al.; Tam et al., 2024; Stoica et al., 2024; Hazimeh et al.; Ainsworth et al., 2022). Other approaches improve TA by designing better weighting schemes for model averaging (Matena and Raffel, 2022; Jin et al., 2022; Wang et al., 2024a; Zhou et al., 2024). A different research direction focuses on manipulating task vectors to enhance merging performance (Yadav et al., 2024; Du et al., 2024; Jiang et al., 2023; He et al., 2024; Wang et al., 2024b; Davari and Belilovsky, 2024). Alternatively, some methods perform adaptive model merging using dynamic inference, employing either a router (Lu et al., 2024; Stoica et al., 2023) or masks (Huang et al., 2024).

Merging LoRA Models. LoRA (Hu et al., 2021) has become a widely used technique for parameter-efficient fine-tuning. However, most existing model merging methods struggle to effectively transfer to LoRA models (Stoica et al., 2024; Tang et al., 2023a). KnOTS (Stoica et al., 2024) highlights the importance of merging LoRA models within a shared space, while Tang et al. (2023a) propose that linearized LoRA models reduce weight entanglement. Zhao et al. (2024) and Prabhakar et al. (2024) further improve LoRA merging by learning optimal merging weights and clustering LoRA modules, respectively.

Despite these advancements, existing methods often overlook the interaction between model parameters and data, leading to suboptimal merging performance. Moreover, prior work primarily focuses on the phases *during* and *after* fine-tuning. In contrast, our method explicitly addresses parameter-data interactions and focuses on the merging phase *before* fine-tuning.

3 Preliminaries and Background

Notation. Let f_0 be a pre-trained model with L layers where $\theta_0 = \{W_0^{(1)}, \ldots, W_0^{(L)}\}$ is the parameters of the model and $W_0^{(l)}$ is the weight matrix of the l-th layer. During adaptation, f_0 is individually fine-tuned on N downstream tasks $\{T_1, \ldots, T_N\}$

with θ_t the fine-tuned parameters on task T_t where $\theta_t = \{W_t^{(1)}, \ldots, W_t^{(L)}\}$.

Low-Rank Adaptation. Low-rank adaptation (LoRA) (Hu et al., 2021) is a popular method for efficiently fine-tuning a pre-trained model on a downstream task. It introduces low-rank subspaces to contain the parameter updates during fine-tuning. Specifically, the parameter update for a weight matrix $W \in \mathbb{R}^{m\times n}$ is represented as $\Delta W = BA$, where $B \in \mathbb{R}^{m\times r}$ and $A \in \mathbb{R}^{r\times n}$ are two learnable matrices with $r \ll \min(m, n)$. Typically, B and A are initialized as zeros and random Gaussian noise before fine-tuning, respectively, and then learned during the fine-tuning phase. We denote the latent feature at the l-th layer as:

$$\begin{aligned}\mathbf{h}^{(l)} &= W_0^{(l)}\mathbf{h}^{(l-1)} + \Delta W^{(l)}\mathbf{h}^{(l-1)} \\ &= W_0^{(l)}\mathbf{h}^{(l-1)} + B^{(l)}A^{(l)}\mathbf{h}^{(l-1)}.\end{aligned}$$

Model Merging. We focus on two categories of model merging: weighted averaging and task vector manipulations.

Task Arithmetic (TA) (Ilharco et al., 2022) merges models by linearly summing their parameters as $\theta_0 + \lambda \sum_t (\theta_t - \theta_0)$, where λ is a scaling coefficient tuned on a validation set. *Fisher Merging* (Matena and Raffel, 2022) improves upon TA by modeling each model's posterior as a Gaussian distribution, leveraging Fisher information to weight model averages instead of using a single uniform coefficient. *RegMean* (Jin et al., 2022) extends model merging by drawing inspiration from linear models, aiming to minimize transformation shifts on data before and after merging. The design leads to an analytical solution that generalizes to language models.

TIES (Yadav et al., 2024) further refines TA by operating at the level of task vectors. It first reduces redundancy by pruning low-magnitude parameters, then resolves parameter conflicts by selecting dominant signs, and finally merges only the aligned parameters. For more flexible merging, Huang et al. (2024) propose an adaptive approach *EMR* that generates a task-specific model at inference time. Their method selects a unified base model from a set of candidates and dynamically applies task-specific masks and rescaling factors.

4 Our Proposed Method

In this section, we first introduce our proposed method OSRM (**O**rthogonal **S**ubspaces for **R**obust model **M**erging) for constraining the transformation capacity of the LoRA subspace *before* fine-tuning to improve the performance of model merging *after* fine-tuning. We then propose several practical extensions to facilitate the integration of our method into real-world scenarios.

4.1 Motivation

Existing approaches often seek to eliminate interference among multiple models via weight disentanglement, such as orthogonalizing task vectors. Recently, Stoica et al. (2024) argued that task-vector orthogonality does not necessarily imply no interference and proposed a data-free method to align LoRA modules in a shared space. Despite its promising empirical results, data-free approaches ignore how parameters interact with the input features in each layer and may be suboptimal in effectively mitigating interference.

Without loss of generality, let us consider merging two tasks T_1 and T_2 via task arithmetic as an illustrative example; we omit the subscript l for the layer index. The merged weight matrix is:

$$W_m = W_0 + \Delta W_1 + \Delta W_2 = W_0 + B_1A_1 + B_2A_2,$$

where $\{B_t, A_t\}$ denote the LoRA matrices for task t. During inference, given a latent feature $\mathbf{h}_1$ that arises from a sample of task T_1, the transformation of W_m on $\mathbf{h}_1$ is:

$$\begin{aligned}W_m\mathbf{h}_1 &= (W_0 + B_1A_1 + B_2A_2)\mathbf{h}_1 \\ &= (W_0 + B_1A_1)\mathbf{h}_1 + B_2A_2\mathbf{h}_1 \\ &= W_1\mathbf{h}_1 + B_2A_2\,\mathbf{h}_1, \qquad (1)\end{aligned}$$

where $W_1 = W_0 + B_1A_1$ is the fine-tuned model for task T_1, and $B_2A_2\,\mathbf{h}_1$ can be viewed as a "perturbation" induced by the knowledge learned for task T_2. To reduce the influence of B_2A_2 on $\mathbf{h}_1$, we must consider this interaction and limit the transformation capacity of B_2A_2 when operating on $\mathbf{h}_1$. This motivated us to propose a novel approach.

4.2 Constraining the LoRA Subspace

We first generalize Eq. (1) to the situation where $k > 1$ samples (from task T_1) are available to characterize the interference. Let $H_1 \in \mathbb{R}^{k\times n}$ be the matrix whose rows are the latent features of these k samples. The merged transformation becomes:

$$W_mH_1^\top = W_1H_1^\top + B_2A_2H_1^\top.$$

To make $W_mH_1^\top$ as close as possible to $W_1H_1^\top$, ideally one wants to force $A_2H_1^\top = \mathbf{0}$. If H_1 is

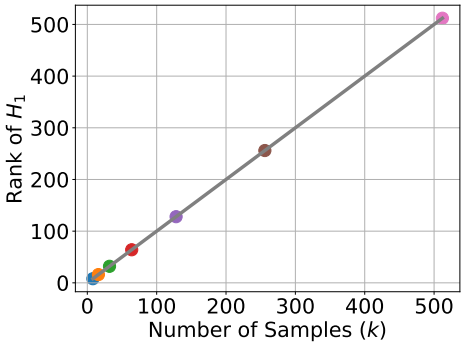


Figure 2: The rank of H_1 (y-axis) vs. the number of samples k (x-axis) with RoBERTa-large (Liu, 2019). The grey line represents $y = x$. For each dot, k samples are randomly selected to concatenate their latent features as H_1.

rank-deficient, we could simply constrain each row of A_2 to be in the null space of H_1. However, Fig. 2 shows that H_1 is typically full-rank in real scenarios due to intrinsic data variability, so we cannot generally make $A_2 H_1^\top = \mathbf{0}$. Instead, we propose to *reduce* the interference by minimizing its Frobenius norm:

$$\min_{A_2} \left\| A_2 H_1^\top \right\|_F.$$

We further opt *not* to constrain B_2 because stringent constraints on both matrices could degrade the representation power needed for task T_2. Thus, our strategy is to disentangle the LoRA matrices so that A_2 is constrained *prior to* fine-tuning to reduce interference with other tasks, while B_2 remains free to maintain downstream performance.

Directly minimizing $\|A_2 H_1^\top\|_F$ without constraints would trivially yield $A_2 = \mathbf{0}$. In practice, although $\{B, A\}$ serve as low-rank approximations of the fine-tuned weight ΔW, the learned $\{B, A\}$ are mostly full-rank. Because any full-rank matrices can be factorized using RQ decomposition, we can safely set A to be an orthogonal basis.

To balance these considerations, we require A_2 to have orthonormal rows, i.e. $A_2 A_2^\top = I$. This enforces a full-rank condition on the row space of A_2 while removing the scale ambiguity between B_2 and A_2. Indeed, for any non-zero scalar c, $(cB_2)\,(\frac{1}{c}A_2)$ gives the same product as $B_2 A_2$. Imposing $A_2 A_2^\top = I$ fixes the scaling of A_2 and forces B_2 to handle the appropriate scaling. To this end, we arrive at the following optimization:

$$\tilde{A}_2 = \arg\min_A \left\| A\, H_1^\top \right\|_F^2, \text{ s.t. } A\, A^\top = I. \quad (2)$$

We show that it admits an analytical solution.

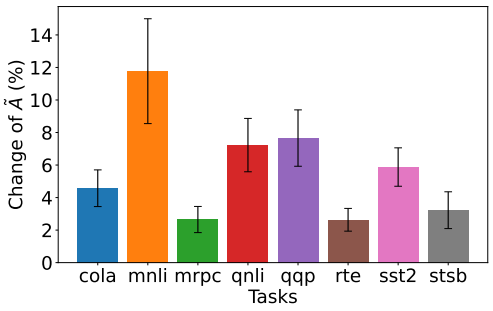


Figure 3: The change of $\tilde{A}$ (%) after fine-tuning compared to the initialization. A normalized distance is used as the metric. See Section 4.4 for details.

4.3 Analytical Solution

For brevity, we temporarily drop the subscripts (i.e. write A instead of A_2, and H instead of H_1). Let $S = \frac{1}{k-1} H^\top H$ be the sample covariance matrix of H. Since S is symmetric and positive semi-definite, we can perform the eigendecomposition:

$$S = V \Lambda V^\top,$$

where $V \in \mathbb{R}^{n \times n}$ is an orthogonal matrix and $\Lambda = \mathrm{diag}(\lambda_1, \ldots, \lambda_n)$ contains the eigenvalues. With a descending order of eigenvalues $\lambda_1 \geq \lambda_2 \geq \cdots \geq \lambda_n$, the analytical solution to Eq. (2) is:

$$\tilde{A}_2 = V_{:,n-r:n}^\top, \quad (3)$$

where $V_{:,n-r:n}$ denotes the last r eigenvectors of S corresponding to the r smallest eigenvalues. See Appendix A for detailed derivation. Although the theory constrains A_2, in practice, one might still update A_2 to prevent excessive loss of accuracy on the target task (c.f. Section 4.4).

Interpretation. Intuitively, $\|A_2 H_1^\top\|_F^2 = \mathrm{tr}(A_2 S A_2^\top)$ measures how A_2 amplifies the principal directions of H_1 in its row space. By choosing directions corresponding to the smallest eigenvalues of S, we place A_2 in the subspace where H_1 has the minimal variance, thereby reducing the interference term $B_2 A_2 H_1^\top$ most effectively. Meanwhile, the row-orthonormal constraint $A_2 A_2^\top = I$ preserves non-trivial capacity for B_2 to learn scaling factors and ensure the model can still fit task T_2.

4.4 Practical Extensions

We now discuss practical extensions that facilitate the adaptation of OSRM to real-world applications. Algorithm 1 summarizes the overall procedure to merge models with OSRM.

Algorithm 1 Model merging with OSRM

Input: a pre-trained model f_0, tasks $[T_1, \ldots, T_N]$, a merging method $\mathcal{M}$, number of layers L.

```
/* Constrain the LoRA subspace */
for t = 1, ··· , N do
    Randomly select k validation samples from T_t
    Collect and concatenate latent features {H_t}_{l=1}^L for all layers
    Average sample-wise features to get {H̄_t^(l)}_{l=1}^L based on Eq. (4)
end for
/* Fine-tune on downstream tasks */
for t = 1, ··· , N do
    Initialize {A_t^(l)}_{l=1}^L to the solution in Eq. (3)
    Initialize {B_t^(l)}_{l=1}^L to zeros
    Fine-tune {B_t^(l), A_t^(l)}_{l=1}^L on T_t to get θ_t
end for
/* Merge fine-tuned models */
Merge the fine-tuned models with an existing method θ_m = M(θ_1, ··· , θ_N)
```

Relaxing the Constraint during Fine-Tuning. To minimize interference among merged models, our analysis suggests freezing $\tilde{A}_2$ at its solution in Eq. (3) during fine-tuning. Nonetheless, our empirical results show this can significantly degrade single-task accuracy. One possible reason for the performance drop is that fixing A_2 restricts the model's adaptation capability and hurts performance on T_2. To avoid this, we propose that $\tilde{A}_2$ should only be used as the *initialization* of A_2, and allow it to be updated during fine-tuning, which we show significantly improves single-task performance.

To show that $\tilde{A}_2$ is still validly orthogonal to the latent features, we adopt the orthogonal Procrustes problem (Gower and Dijksterhuis, 2004):

$$D = \min_{\Omega} \|\Omega \tilde{A}^{\text{ft}} - \tilde{A}^{\text{init}}\|_F, \text{ s.t. } \Omega^\top \Omega = I,$$

where $\tilde{A}^{\text{ft}}$ and $\tilde{A}^{\text{init}}$ are the fine-tuned and initialized matrices of $\tilde{A}$, respectively. Thus, D measures the distance between two matrices under orthogonal transformations. We then use the normalized distance as the metric to measure the change of $\tilde{A}^{\text{init}}$ after fine-tuning, denoted as $D/\|\tilde{A}^{\text{init}}\|_F$. Results in Fig. 3 show that the change of $\tilde{A}$ is up to 14% approximately, which is marginal, implying the validity of our relaxation.

Extension to Multiple Tasks. When merging more than two tasks, say $T_1, \ldots, T_N$, one can gather latent features from all tasks *except* T_t to build $\tilde{A}_t$. Concretely, let H_i be the latent feature matrix of task T_i. For task T_t, we concatenate the features of the other tasks into

$$H_{\neg t} = \left[H_1; \cdots; H_{t-1}; H_{t+1}; \cdots; H_N\right].$$

Then the objective in Eq. (2) becomes:

$$\tilde{A}_t = \arg\min_A \left\|A\, H_{\neg t}^\top\right\|_F^2 \quad \text{s.t.} \quad A\, A^\top = I.$$

In large-scale or privacy-sensitive applications, storing or sharing *all* latent features can be memory-intensive or prohibitive. A practical fix is to *average* the sample-wise latent features in each task:

$$\bar{H}_i = \frac{1}{k} \sum\nolimits_{j=1}^{k} H_{i;j,:}, \tag{4}$$

where $H_{i;j,:}$ is the j-th row of H_i. We then replace $H_{\neg t}$ with the concatenation of $\{\bar{H}_i \mid i \neq t\}$ to reduce memory usage and mitigate privacy concerns.

5 Experiments

5.1 Experimental Settings

Datasets. We evaluate our method using eight datasets from the GLUE benchmark (Wang et al., 2019), a widely used suite of natural language understanding tasks. These tasks encompass both single-sentence and sentence-pair classification, including MRPC (Dolan and Brockett, 2005), QQP (Iyer et al., 2017), QNLI (Rajpurkar et al., 2016), MNLI (Williams et al., 2018), SST-2 (Socher et al., 2013), CoLA (Warstadt et al., 2018), STS-B (Cer et al., 2017), and RTE (Giampiccolo et al., 2007). Each dataset is split into training, validation, and test sets.

For evaluation, we use the Matthews correlation coeff. for CoLA and the average of the Pearson and Spearman correlation coeff. for STS-B, and accuracy is used for the remaining tasks. Since our method is applied *before* fine-tuning and influences the training, we report absolute metric values in all experiments rather than normalized scores.

Models. To assess the effectiveness of our approach across different architectures, we evaluate three language models: RoBERTa-large (Liu, 2019) (encoder-only), T5-large (Raffel et al., 2020) (encoder-decoder), and Llama3.2-1B (Dubey et al., 2024) (decoder-only). Additionally, we test our method on the larger Llama3.2-3B and Llama3-8B (Dubey et al., 2024). The main results for these models are presented in Section 5.2.

Table 1: Per-task performance (%) of merging fine-tuned RoBERTa-large models. "Individual" refers to the performance of each fine-tuned model on the dataset on which it was trained. **Bold** indicates a better performance.

	OSRM	CoLA	MNLI	MRPC	QNLI	QQP	RTE	SST2	STSB	Avg.
Individual	No	**68.75**	**89.05**	91.67	94.23	89.38	83.03	**96.33**	91.92	**88.05**
	Yes	66.28	88.88	91.67	**94.62**	**89.45**	**83.75**	96.1	**92.17**	87.87
TA	No	18.57	74.01	77.7	73.75	82.84	71.12	87.16	75.13	70.04
	Yes	**32.25**	**81.24**	77.7	**86.78**	**84.99**	**80.14**	**91.63**	**77.96**	**76.59**
RegMean	No	31.77	66.85	69.61	**87.42**	**83.94**	**72.56**	**94.72**	36.19	67.88
	Yes	**40.36**	**71.45**	**76.96**	72.01	76.59	63.54	92.32	**65.28**	**69.81**
Fisher	No	36.16	53.62	70.1	**65.15**	71.23	67.15	91.17	64.45	64.88
	Yes	**39.11**	**74.85**	**76.72**	62.51	**81.64**	67.15	**93.35**	**79.56**	**71.86**
TIES	No	13.06	70.23	**63.73**	**83.64**	83.36	63.18	87.61	25.75	61.32
	Yes	**29.88**	**77**	58.24	83.52	**84.39**	**72.92**	**90.48**	**36.75**	**66.65**
EMR	No	54.69	83.31	**79.66**	89.35	85.04	**81.95**	92.78	82.63	81.18
	Yes	**56.22**	**88.51**	77.21	**93.3**	**87.54**	79.78	**95.41**	**88.56**	**83.32**

Table 2: Per-task performance (%) of T5-large. "Individual" refers to the metrics of each fine-tuned model on the dataset on which it was trained. **Bold** indicates a better performance.

	OSRM	CoLA	MNLI	MRPC	QNLI	QQP	RTE	SST2	STSB	Avg.
Individual	No	60.86	88.17	88.73	94.25	90.42	81.59	95.76	91.25	86.38
	Yes	**62.58**	**88.31**	**90.69**	**94.62**	**91.04**	81.59	**95.87**	**91.36**	**87.01**
TA	No	**28.04**	68.10	**44.85**	56.07	**35.35**	58.48	56.08	76.74	52.96
	Yes	25.24	**70.83**	28.92	**79.21**	34.50	**65.70**	**62.16**	**79.79**	**55.79**
RegMean	No	41.22	**67.92**	53.68	84.77	87.02	68.23	**93.00**	63.51	69.92
	Yes	**43.67**	66.19	**69.36**	**87.24**	**88.32**	**70.76**	92.89	**70.98**	**73.68**
Fisher	No	13.72	**35.45**	31.62	49.46	63.18	**52.71**	49.08	**90.60**	48.23
	Yes	**38.91**	34.68	**68.38**	**51.22**	63.18	49.10	**53.56**	90.02	**56.13**
TIES	No	36.67	**73.13**	29.90	62.46	39.78	64.26	63.65	**74.07**	55.49
	Yes	**40.16**	70.27	**33.09**	**82.34**	**48.51**	**64.62**	**65.37**	73.57	**59.74**
EMR	No	37.16	87.27	79.66	93.30	88.32	**80.14**	94.27	88.08	81.02
	Yes	**39.54**	**88.40**	**85.78**	**94.07**	**89.52**	79.90	**94.95**	**88.72**	**82.61**

Baselines. We evaluate the effectiveness of our method against five widely used model merging techniques. Ilharco et al. (2022) introduced Task Arithmetic (TA) to merge models with a unified weight. Fisher merging (Matena and Raffel, 2022) and RegMean (Jin et al., 2022) improve upon TA by incorporating anisotropic weighting, utilizing Fisher information and the inner product of data matrices, respectively. TIES (Yadav et al., 2024) performs merging at the level of task vectors, while EMR (Huang et al., 2024) represents the state-of-the-art in adaptive model merging. For additional background, refer to Section 3.

Implementation Details. For training, we follow (Hu et al., 2021) and use the AdamW optimizer (Loshchilov, 2017) with a warmup ratio of 0.06 and a linear learning rate schedule. Following (Hu et al., 2021), LoRA is configured with a rank of $r = 8$, a scaling factor of $\alpha = 16$, and is only applied to the query and value blocks. We study the effect of different learnable blocks in Section 5.3. Other hyperparameters are selected via grid search as in (Liu, 2019) (c.f. Appendix D).

For model merging, we adopt hyperparameter settings from prior work (Yadav et al., 2024; Jin et al., 2022; Matena and Raffel, 2022). Specifically, we set the scaling coefficient to 0.3 for TA and 1 for TIES. The non-diagonal multiplier in RegMean is set to 0.9, except for T5-large, where it is 0.1. For Fisher-based merging, we use a uniform scaling factor of $\frac{1}{8}$ across all models. For methods requiring validation data, we use up to 1000 samples from the validation set, following (Jin et al., 2022). We use 100 samples per task in our method to compute H_t. The effect of hyperparameter variations is analyzed in Section 5.3. Our code implementation is adapted from (Huang et al., 2024) and available at `https://github.com/illidanlab/OSRM`.

5.2 Main Results

Merging Encoder-Only Models. The performance of merging RoBERTa-large models is presented in Table 1. Our proposed method consistently outperforms all the baselines across all merging techniques on average.

Table 3: Per-task performance (%) of Llama3.2-1B. "Individual" refers to the metrics of each fine-tuned model on the dataset on which it was trained. **Bold** indicates a better performance.

	OSRM	CoLA	MNLI	MRPC	QNLI	QQP	RTE	SST2	STSB	Avg.
Individual	No	59.97	85.44	**87.25**	91.10	88.34	78.34	94.84	**90.10**	84.42
	Yes	**61.44**	**86.91**	86.52	**92.26**	**89.07**	**82.67**	**95.76**	89.56	**85.52**
TA	No	26.05	**72.63**	66.91	60.70	**82.61**	57.04	**92.32**	67.15	65.68
	Yes	**28.20**	66.69	66.91	**73.48**	81.66	**68.23**	92.20	**71.24**	**68.58**
RegMean	No	30.55	**34.08**	41.42	48.89	52.48	50.54	48.28	23.31	41.19
	Yes	**34.96**	32.97	**43.14**	**49.99**	**60.39**	**53.07**	**48.62**	**25.45**	**43.57**
Fisher	No	**19.91**	60.47	**69.85**	**60.28**	**82.18**	70.40	92.09	**33.29**	**61.06**
	Yes	19.40	**62.42**	69.61	59.93	74.86	**71.99**	**92.34**	26.58	59.64
TIES	No	41.18	81.77	76.47	**87.35**	85.08	**75.45**	92.43	83.04	77.85
	Yes	**42.02**	**83.03**	**78.43**	87.04	**87.35**	74.73	**94.95**	**82.62**	**78.77**
EMR	No	37.16	87.27	79.66	93.30	88.32	**80.14**	94.27	88.08	81.02
	Yes	**39.54**	**88.40**	**85.78**	**94.07**	**89.52**	79.90	**94.95**	**88.72**	**82.61**

Table 4: Per-task performance (%) of Llama3.2-3B. "Individual" refers to the metrics of each fine-tuned model on the dataset on which it was trained. **Bold** indicates a better performance.

	OSRM	CoLA	MNLI	MRPC	QNLI	QQP	RTE	SST2	STSB	Avg.
Individual	No	68.61	89.31	**87.01**	93.26	89.81	85.92	96.10	**90.25**	87.54
	Yes	**69.25**	**89.57**	86.52	**94.51**	**90.15**	**89.89**	**96.90**	89.82	**88.33**
TA	No	34.80	77.87	72.30	80.03	84.64	**68.23**	**93.46**	74.47	73.22
	Yes	**35.50**	**83.79**	**72.79**	**84.50**	**84.79**	63.54	92.78	**77.20**	**74.36**
RegMean	No	40.61	33.46	**50.74**	50.03	50.84	46.21	50.46	**41.90**	45.53
	Yes	**44.04**	**33.51**	42.16	**50.72**	**53.72**	**49.82**	**51.11**	41.10	**45.77**
TIES	No	**28.21**	34.14	62.50	66.17	73.50	**54.15**	89.91	29.24	54.73
	Yes	22.75	**41.06**	**65.20**	**71.41**	**75.88**	49.82	**90.02**	**42.12**	**57.28**
EMR	No	54.74	87.92	**84.31**	91.34	87.52	**84.12**	95.64	81.86	83.43
	Yes	**55.57**	**89.43**	80.15	**93.61**	**88.72**	79.42	**96.56**	**84.32**	**83.47**

Specifically, for TA merging, our method achieves performance at least on par with those baselines. It surpasses the baselines in seven out of eight tasks, with a notable improvement of over 13% on CoLA. In the Fisher merging setting, our method slightly underperforms on the QNLI dataset, with a marginal gap of less than 3%, but outperforms the baseline across all other tasks, achieving up to a 21% improvement on MNLI. For TIES and EMR, while the baseline slightly surpasses our method on two datasets, our approach yields significantly better overall performance across the eight datasets.

Moreover, our method minimally impacts downstream task performance, with an average performance gap of less than 1%, and even surpasses the baseline on four datasets.

Merging Encoder-Decoder Models. The results for T5-large are reported in Table 2. Our method substantially outperforms the baseline on RegMean, Fisher, and TIES. On average, it improves RegMean and Fisher by 3.76% and 7.9%, respectively. Notably, our approach enhances performance on MRPC by approximately 16% under RegMean and on QQP by around 9% under TIES.

Although the baseline slightly outperforms our method on Fisher merging, the difference is minimal at just 0.07%. Additionally, our method consistently improves downstream task performance across all eight datasets.

Merging Decoder-Only Models. The results for Llama3.2-1B are shown in Table 3. On average, our method surpasses the baseline on TA, RegMean, TIES, and EMR. Per-task improvements reach up to 12.78%, 7.91%, and 6.12% for TA, RegMean, and EMR, respectively.

Importantly, our method consistently enhances downstream fine-tuning performance on Llama3.2-1B, demonstrating its effectiveness in decoder-only model merging.

Merging Large Language Models. Following the setting of baselines, we further evaluate the effectiveness of our method on the large language model Llama3.2-3B, as shown in Table 4. Due to the large scale of Llama3.2-3B, we focus solely on gradient-free merging methods and exclude Fisher merging. On average, our method outperforms the baseline across all merging techniques, including downstream task performance. Notably, the per-task improvement reaches up to 5.92% on TA and

Table 5: Per-task performance (%) of Llama3-8B. "Individual" refers to the metrics of each fine-tuned model on the dataset on which it was trained. **Bold** indicates a better performance.

	OSRM	CoLA	MNLI	MRPC	QNLI	QQP	RTE	SST2	STSB	Avg.
Individual	No	67.57	88.74	**89.95**	94.65	**90.26**	84.48	96.44	**91.40**	87.94
	Yes	**69.20**	**90.15**	89.46	**95.44**	90.17	**88.09**	**97.02**	90.65	**88.77**
TA	No	10.00	64.74	31.62	64.74	**81.22**	80.87	**92.89**	**80.14**	63.28
	Yes	**52.89**	**82.53**	31.62	**89.58**	72.56	**87.00**	88.76	77.08	**72.75**
TIES	No	**28.28**	**35.68**	**41.18**	65.62	66.97	**61.73**	**78.67**	**45.97**	**53.01**
	Yes	27.75	33.54	36.52	**71.19**	**77.10**	59.46	74.56	41.36	52.69

12.88% on TIES. Furthermore, compared to previous results, we observe that although larger models generally achieve higher average performance than smaller ones, the performance gain from our method diminishes. A possible explanation is that as model size increases, its inherent knowledge also expands, which naturally enhances merging performance, thereby reducing the relative impact of our approach.

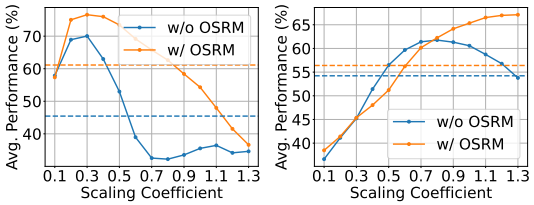
Moreover, we present the results of merging LLaMA3-8B models in Table 5. Given the large scale of the model and resource constraints, we evaluate our proposed OSRM using two lightweight merging techniques: TA and TIES. Our method consistently improves both downstream task performance and the merging effectiveness of TA. Notably, it outperforms the baseline on seven out of eight datasets when combined with TA, demonstrating its clear advantage.

Discussion. First, we observe that the choice of the optimal non-adaptive merging method (i.e., excluding EMR) varies depending on the model. For example, TA achieves the best performance on RoBERTa-large, while RegMean outperforms other methods on T5-large. Even between two decoder-based models, the most effective merging strategy can differ. Second, EMR consistently achieves the highest performance among all methods, performing close to that of individual models. It implies the superiority of adaptive merging methods.

5.3 Robustness Analysis

In this section, we evaluate the robustness of our method to different hyperparameters, including the scaling coefficient λ, the number of samples k, the number of tasks N, and the choice of learnable blocks, using RoBERTa-large.

Impact of Scaling Coefficient λ. We analyze the effect of different scaling coefficients λ on merging performance in Fig. 4. We focus on TA and TIES, both of which require tuning λ before merging. The values of λ are sampled from the range $[0.1, 1.3]$ with a step size of 0.1. The results indicate that our method consistently outperforms the baseline and is more robust to variations in λ. Specifically, for TA, our approach achieves superior performance across almost the entire range and demonstrates a significantly higher average performance. Although for $\lambda \in [0.3, 0.7]$, our method is slightly worse than the baseline on TIES, it still yields substantial improvements for other values of λ and achieves a higher average overall. The robustness of our method suggests its practical efficiency, as it does not require fine-grained tuning of the scaling coefficient.



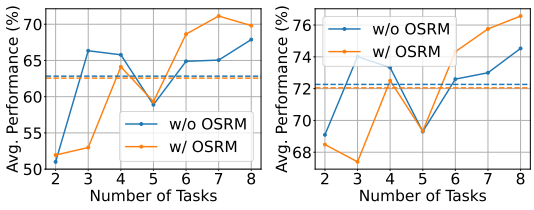
(a) TA (b) TIES

Figure 4: Effect of scaling coefficients on the performance of TA and TIES merging. Results are averaged across eight datasets. The solid line is the merging performance for each scaling coefficient. The dashed line is the average performance for each method.

Table 6: Effect of the number of samples used to compute Eq. (4). Results are averaged across eight datasets. **Bold** indicates the best performance for each merging method.

	2	10	100	1000	5000	w/o OSRM
RegMean	47.66	**70.42**	69.81	69.21	70.16	67.88
Fisher	66.19	65.22	**71.86**	62.71	61.71	64.88
EMR	81.73	82.84	**83.32**	81.47	78.59	81.18

Impact of Sample Size k. We evaluate the influence of different values of k in Eq. (4) on merging performance. To minimize the impact of other hyperparameters, such as the scaling coefficient, we focus on RegMean, Fisher, and EMR. Following (Jin et al., 2022), we consider



(a) RegMean (b) EMR

Figure 5: Performance of merging different numbers of tasks with RegMean and EMR. Results are averaged across eight datasets. The solid line is the merging performance for each number of tasks. The dashed line is the average performance for each method.

$k \in \{2, 10, 100, 1000, 5000\}$. Intuitively, increasing k should result in an initialization of $\tilde{A}$ that is more orthogonal to out-of-task samples, potentially leading to better performance. However, as shown in Table 6, the optimal values of k are 10 and 100 for RegMean, Fisher, and EMR, respectively.

This phenomenon can be attributed to two key factors. First, when k is close to 1, we observe an ill-conditioned latent feature matrix H, making the computation of $\tilde{A}$ more challenging. Second, as k increases, the knowledge overlap between in-task and out-of-task samples also grows. In this case, enforcing $\tilde{A}$ to be orthogonal to these samples may degrade performance by discarding useful shared knowledge. Due to the complex interplay between fine-tuning procedures, model parameters, and data characteristics, it is challenging to analytically determine the cause of the observed counter-intuitive results. We believe the exploration of this situation is interesting. Despite this, Table 6 shows that our method consistently outperforms the baseline with as few as 100 samples, highlighting its practical applicability.

Impact of the Number of Tasks N. We investigate the effect of the number of tasks N on merging performance in Fig. 5. Although the average performance of our method and the baseline remains comparable for small N, our approach shows clear advantages when N increases. Specifically, when $N > 5$, our method begins to outperform the baseline, with improvements becoming more significant as N grows. This observation suggests that our method enhances merging performance in large-scale settings, demonstrating its scalability.

Impact of Learnable Blocks. We evaluate the effectiveness of our method with various learnable blocks in Table 7. While the main experiments in Section 5.2 focus on fine-tuning the query (Q) and value (V) blocks, we further investigate the impact of fine-tuning all three blocks—query (Q), key (K), and value (V)—as well as fine-tuning the query blocks alone. Although our method introduces a slight degradation in downstream task performance, it consistently outperforms baseline approaches when integrated with TA, RegMean, and Fisher merging strategies. Additionally, we observe that fine-tuning Q, K, and V yields better performance than fine-tuning Q alone, suggesting that increasing the number of learnable blocks contributes to more effective model merging.

Table 7: Effect of different learnable blocks. Results are averaged across eight datasets. **Bold** indicates better performance.

	OSRM	Individual	TA	RegMean	Fisher	TIES	EMR
Q, K, V	No	**87.81**	71.24	67.89	65.60	**51.61**	78.56
	Yes	87.76	**75.85**	**67.95**	**67.81**	49.90	**82.16**
Q	No	**86.67**	71.95	43.90	58.63	**41.63**	**79.64**
	Yes	86.26	**73.83**	**52.55**	**60.04**	40.78	78.80

5.4 Extension to Merging Existing LoRA Modules

While there are some cases where the LoRA modules are obtained externally, such as HuggingFace checkpoints, we extend our method to merging existing LoRA modules by decomposing the learned weight with the OSRM-based solution $\tilde{A}$. Specifically, given a set of learned weights $\{\Delta W_t\}_{t=1}^N$, we use the analytically-derived $\tilde{A}_t$ to decompose and recover ΔW_t by using $\hat{B}_t = \min_B \|B\tilde{A}_t - \Delta W_t\|_F$ and $\Delta\hat{W}_t = \hat{B}_t\tilde{A}_t$. Then we merge the recovered $\{\Delta\hat{W}_t\}_{t=1}^N$. While this decomposition may suffer from degradation of the downstream task performance, the recovered $\Delta\hat{W}_t$ can still maintain the property as shown in Eq. (2). See Appendix E.2 for results.

6 Conclusion

We present a novel approach to address performance degradation when merging LoRA-based models. By constraining the LoRA subspace before fine-tuning, our method decreases harmful output shifts arising from data and parameter interference. Empirical results on eight datasets show that this approach substantially improves upon existing merging strategies across multiple LMs.

7 Limitations

While our approach shows significant performance improvement in model merging, some limitations should be discussed. First, similar to previous works, our method relies on the identical model architecture, which limits its applicability across different model types. Second, since we only focus on LoRA models, our method cannot be applied to merging fully fine-tuned models. Future works could further investigate the potential application of our method on models with different architectures or fully fine-tuned.

Acknowledgments

We thank anonymous reviewers for their helpful comments. This material is based in part upon work supported by the National Science Foundation under Grant IIS-2212174, National Institute of Aging (NIA) 1RF1AG072449, National Institute of General Medical Sciences (NIGMS) 1R01GM145700.

References

Samuel K Ainsworth, Jonathan Hayase, and Siddhartha Srinivasa. 2022. Git re-basin: Merging models modulo permutation symmetries. *arXiv preprint arXiv:2209.04836*.

Daniel Cer, Mona Diab, Eneko Agirre, Inigo Lopez-Gazpio, and Lucia Specia. 2017. Semeval-2017 task 1: Semantic textual similarity-multilingual and cross-lingual focused evaluation. *arXiv preprint arXiv:1708.00055*.

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.

William B Dolan and Chris Brockett. 2005. Automatically constructing a corpus of sentential paraphrases. In *Proceedings of the International Workshop on Paraphrasing*.

Guodong Du, Junlin Lee, Jing Li, Runhua Jiang, Yifei Guo, Shuyang Yu, Hanting Liu, Sim Kuan Goh, Ho-Kin Tang, Daojing He, et al. 2024. Parameter competition balancing for model merging. *arXiv preprint arXiv:2410.02396*.

Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Amy Yang, Angela Fan, et al. 2024. The llama 3 herd of models. *arXiv preprint arXiv:2407.21783*.

Yue Gang, Jianhong Shun, and Mu Qing. 2025. Smarter fine-tuning: How lora enhances large language models.

Lei Gao, Yue Niu, Tingting Tang, Salman Avestimehr, and Murali Annavaram. 2024. Ethos: Rectifying language models in orthogonal parameter space. *arXiv preprint arXiv:2403.08994*.

Danilo Giampiccolo, Bernardo Magnini, Ido Dagan, and Bill Dolan. 2007. The third PASCAL recognizing textual entailment challenge. In *Proceedings of the ACL-PASCAL workshop on textual entailment and paraphrasing*, pages 1–9. Association for Computational Linguistics.

John C Gower and Garmt B Dijksterhuis. 2004. *Procrustes problems*, volume 30. OUP Oxford.

Adam Hazimeh, Alessandro Favero, and Pascal Frossard. Task addition and weight disentanglement in closed-vocabulary models. In *Workshop on Efficient Systems for Foundation Models II@ ICML2024*.

Yifei He, Yuzheng Hu, Yong Lin, Tong Zhang, and Han Zhao. 2024. Localize-and-stitch: Efficient model merging via sparse task arithmetic. *arXiv preprint arXiv:2408.13656*.

Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. 2021. Lora: Low-rank adaptation of large language models. *arXiv preprint arXiv:2106.09685*.

Chenyu Huang, Peng Ye, Tao Chen, Tong He, Xiangyu Yue, and Wanli Ouyang. 2024. Emr-merging: Tuning-free high-performance model merging. *arXiv preprint arXiv:2405.17461*.

Gabriel Ilharco, Marco Tulio Ribeiro, Mitchell Wortsman, Suchin Gururangan, Ludwig Schmidt, Hannaneh Hajishirzi, and Ali Farhadi. 2022. Editing models with task arithmetic. *arXiv preprint arXiv:2212.04089*.

Shankar Iyer, Nikhil Dandekar, Kornél Csernai, et al. 2017. First quora dataset release: Question pairs. *data. quora. com.*

Weisen Jiang, Baijiong Lin, Han Shi, Yu Zhang, Zhenguo Li, and James Kwok. 2023. Byom: Building your own multi-task model for free. *Preprint*.

Xisen Jin, Xiang Ren, Daniel Preotiuc-Pietro, and Pengxiang Cheng. 2022. Dataless knowledge fusion by merging weights of language models. *arXiv preprint arXiv:2212.09849*.

So Kuroki, Taishi Nakamura, Takuya Akiba, and Yujin Tang. 2024. Agent skill acquisition for large language models via cycleqd. *arXiv preprint arXiv:2410.14735*.

Yinhan Liu. 2019. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*, 364.

I Loshchilov. 2017. Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101*.

Zhenyi Lu, Chenghao Fan, Wei Wei, Xiaoye Qu, Dangyang Chen, and Yu Cheng. 2024. Twin-merging: Dynamic integration of modular expertise in model merging. *arXiv preprint arXiv:2406.15479*.

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

Guillermo Ortiz-Jimenez, Alessandro Favero, and Pascal Frossard. 2023. Task arithmetic in the tangent space: Improved editing of pre-trained models. *Advances in Neural Information Processing Systems*, 36:66727–66754.

Akshara Prabhakar, Yuanzhi Li, Karthik Narasimhan, Sham Kakade, Eran Malach, and Samy Jelassi. 2024. Lora soups: Merging loras for practical skill composition tasks. *arXiv preprint arXiv:2410.13025*.

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

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of machine learning research*, 21(140):1–67.

Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. 2016. SQuAD: 100,000+ questions for machine comprehension of text. In *Proceedings of EMNLP*, pages 2383–2392. Association for Computational Linguistics.

Ozan Sener and Vladlen Koltun. 2018. Multi-task learning as multi-objective optimization. *Advances in neural information processing systems*, 31.

Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D Manning, Andrew Ng, and Christopher Potts. 2013. Recursive deep models for semantic compositionality over a sentiment treebank. In *Proceedings of EMNLP*, pages 1631–1642.

George Stoica, Daniel Bolya, Jakob Bjorner, Pratik Ramesh, Taylor Hearn, and Judy Hoffman. 2023. Zipit! merging models from different tasks without training. *arXiv preprint arXiv:2305.03053*.

George Stoica, Pratik Ramesh, Boglarka Ecsedi, Leshem Choshen, and Judy Hoffman. 2024. Model merging with svd to tie the knots. *arXiv preprint arXiv:2410.19735*.

Derek Tam, Mohit Bansal, and Colin Raffel. 2024. Merging by matching models in task parameter subspaces. *Transactions on Machine Learning Research*.

Anke Tang, Li Shen, Yong Luo, Yibing Zhan, Han Hu, Bo Du, Yixin Chen, and Dacheng Tao. 2023a. Parameter efficient multi-task model fusion with partial linearization. *arXiv preprint arXiv:2310.04742*.

Anke Tang, Li Shen, Yong Luo, Yibing Zhan, Han Hu, Bo Du, Yixin Chen, and Dacheng Tao. 2023b. Parameter efficient multi-task model fusion with partial linearization. *arXiv preprint arXiv:2310.04742*.

Zhixu Tao, Ian Mason, Sanjeev Kulkarni, and Xavier Boix. 2024. Task arithmetic through the lens of one-shot federated learning. *arXiv preprint arXiv:2411.18607*.

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

Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R. Bowman. 2019. GLUE: A multi-task benchmark and analysis platform for natural language understanding. In the Proceedings of ICLR.

Ke Wang, Nikolaos Dimitriadis, Alessandro Favero, Guillermo Ortiz-Jimenez, Francois Fleuret, and Pascal Frossard. 2024a. Lines: Post-training layer scaling prevents forgetting and enhances model merging. *arXiv preprint arXiv:2410.17146*.

Ke Wang, Nikolaos Dimitriadis, Guillermo Ortiz-Jimenez, François Fleuret, and Pascal Frossard. 2024b. Localizing task information for improved model merging and compression. *arXiv preprint arXiv:2405.07813*.

Alex Warstadt, Amanpreet Singh, and Samuel R. Bowman. 2018. Neural network acceptability judgments. *arXiv preprint 1805.12471*.

Adina Williams, Nikita Nangia, and Samuel R. Bowman. 2018. A broad-coverage challenge corpus for sentence understanding through inference. In *Proceedings of NAACL-HLT*.

Prateek Yadav, Derek Tam, Leshem Choshen, Colin A Raffel, and Mohit Bansal. 2024. Ties-merging: Resolving interference when merging models. *Advances in Neural Information Processing Systems*, 36.

Kotaro Yoshida, Yuji Naraki, Takafumi Horie, Ryosuke Yamaki, Ryotaro Shimizu, Yuki Saito, Julian McAuley, and Hiroki Naganuma. Mastering task arithmetic: τ jp as a key indicator for weight disentanglement. In *The Thirteenth International Conference on Learning Representations*.

Yu Zhang and Qiang Yang. 2021. A survey on multi-task learning. *IEEE transactions on knowledge and data engineering*, 34(12):5586–5609.

Ziyu Zhao, Tao Shen, Didi Zhu, Zexi Li, Jing Su, Xuwu Wang, Kun Kuang, and Fei Wu. 2024. Merging loras like playing lego: Pushing the modularity of lora to extremes through rank-wise clustering. *arXiv preprint arXiv:2409.16167*.

Yuyan Zhou, Liang Song, Bingning Wang, and Weipeng Chen. 2024. Metagpt: Merging large language models using model exclusive task arithmetic. *arXiv preprint arXiv:2406.11385*.

A Proof.

We prove that Eq. (3) is an analytical solution to the problem Eq. (2).

Proof. Let $S = \frac{1}{k-1} H_1^\top H_1$ be the covariance matrix of H_1, which is symmetric and positive semi-definite. Then

$$\|AH_1^\top\|_F^2 = \mathrm{tr}(AH_1^\top H_1 A^\top) = (k-1)\mathrm{tr}(ASA^\top).$$

The eigendecomposition on S yields

$$S = V\Lambda V^\top,$$

where $V \in \mathbb{R}^{n\times n}$ is orthogonal and $\Lambda = \mathrm{diag}(\lambda_1, ..., \lambda_n)$ is a diagonal matrix with non-negative entities in a descending order. Thus,

$$\mathrm{tr}(ASA^\top) = \mathrm{tr}(AV\Lambda V^\top A^\top).$$

Let $M = AV \in \mathbb{R}^{m\times n}$. Since $AA^\top = I$ and $VV^\top = I$, it follows that $MM^\top = I$ and

$$\mathrm{tr}(ASA^\top) = \mathrm{tr}(M\Lambda M^\top).$$

Considering $M = AV$, the solution to the problem $\arg\min_A \mathrm{tr}(M\Lambda M^\top)$ is spanned by the eigenvectors of S associated with the smallest eigenvalues, i.e., $\tilde{A}_2 = V_{:,n-r:n}^\top$, where $V_{:,n-r:n}$ is the last r eigenvectors associated with the r smallest eigenvalues. □

B Dataset Details

The GLUE benchmark [1] is widely used for general language understanding evaluation. It consists of eight English datasets. We show the details of the datasets we use in Table 8.

C Model Details

We use three language models, including RoBERTa-large [2], T5-large [3], and Llama3.2-1B [4], and one large language model Llama3.2-3B [5] in our experiments. Table 9 shows the details of used models.

[1] https://huggingface.co/datasets/nyu-mll/glue
[2] https://huggingface.co/FacebookAI/roberta-large
[3] https://huggingface.co/google-t5/t5-large
[4] https://huggingface.co/meta-llama/Llama-3.2-1B
[5] https://huggingface.co/meta-llama/Llama-3.2-3B

D Hyper-Parameters

We show the hyper-parameters used to fine-tune language models in Table 10. Similar to (Liu, 2019), we use a grid search for the optimal hyper-parameters. All the experiments are conducted on eight NVIDIA RTX A6000 GPUs.

E More Experimental Results

E.1 Averaged Results

We show the averaged performance of each model across all datasets in Tables 11 to 15, respectively. Results show that our method outperforms the baseline in almost all settings on average.

E.2 Results of Extension to Merging Existing LoRA Modules

In Section 5.4, we extend OSRM to support the merging of existing LoRA modules, such as externally obtained checkpoints from HuggingFace. The corresponding results are reported in Table 16. When OSRM is applied after fine-tuning, the original LoRA weight matrices cannot be perfectly recovered, resulting in a significant drop in the individual performance of the models. Nevertheless, the merged performance remains largely preserved, demonstrating the effectiveness of our method. Preserving individual performance in the post-fine-tuning setting poses an interesting yet non-trivial challenge, which we identify as a promising direction for future work.

F Discussion

F.1 Elaborate Discussion on Limitations

As we have briefly discussed limitations in the main text, we further give elaborate discussion on limitations in this section. First, as LoRA has been widely used for fine-tuning LLMs in many areas such as finance, healthcare, and code generation (Gang et al., 2025), there is still a performance gap in merging LoRA-fine-tuned models (Stoica et al., 2024). Thus, many methods have been proposed to improve the merging performance of models specifically fine-tuned with LoRA (Stoica et al., 2024; Tang et al., 2023b; Prabhakar et al., 2024). Second, the identical model architecture is a common assumption in the area of model merging, such as our baselines, and has many realistic applications, such as one-shot FL (Tao et al., 2024) and LLM agents (Kuroki et al., 2024). We also believe

Table 8: Dataset details in the GLUE benchmark. Acc. and cc. mean accuracy and correlation coefficient, respectively.

Dataset	#Train (K)	#Val (K)	#Test (K)	Metric
CoLA	8.55	1.04	1.06	Matthews cc.
MNLI	393	9.82	9.8	Acc.
MRPC	3.67	0.408	1.73	Acc.
QNLI	105	5.46	5.46	Acc.
QQP	364	40.4	391	Acc.
RTE	2.49	0.277	3	Acc.
SST2	67.3	0.872	1.82	Acc.
STSB	5.75	1.5	1.38	Avg. of Pearson and Spearman cc.

Table 9: Details of used models.

Model	#Params	Architecture
RoBERTa-large	355M	Encoder-only
T5-large	738M	Encoder-decoder
Llama3.2-1B	1.24B	Decoder-only
Llama3.2-3B	3.21B	Decoder-only

model merging under various architectures is an interesting problem, but out of this paper's scope.

F.2 Comparison with Multi-task Learning

There are two main differences between OSRM-based training followed by merging and standard multi-task learning (MTL) (Sener and Koltun, 2018; Zhang and Yang, 2021). First, standard MTL is a data-collecting paradigm while merging with OSRM is a model-collecting paradigm. Specifically, standard MTL requires the samples from all datasets to be collected together to train a multi-task model. Instead, OSRM allows each data source to train its own model and merges the models together. Second, while standard MTL requires the collection of all the original samples, OSRM only requires a small number of latent features from each task (100 samples per task in our experiments).

Compared to MTL, our method can be applied to cases where full data access has memory issues or privacy concerns, such as model merging and one-shot FL (Tao et al., 2024), or other paradigms such as training LLM agents (Kuroki et al., 2024).

Table 10: Hyper-parameters for fine-tuning language models.

Hyper-param	RoBERTa-large	T5-large	Llama3.2-1B	Llama3.2-3B
Learning rate	{3e-5, 2e-4, 4e-4}	{3e-5, 2e-4, 4e-4}	{1e-4, 2e-4, 4e-4}	{2e-4, 4e-4}
Batch size	{32, 64}	{32, 64}	{16, 32}	{16, 32}
Weight decay	{0, 0.01, 0.1}	{0, 0.01, 0.1}	{0, 0.1}	{0, 0.1}
Max #epochs	10	10	10	5

Table 11: Averaged performance (%) across eight tasks of RoBERTa-large. "Individual" refers to the metrics of each fine-tuned model on the dataset on which it was trained. **Bold** indicates a higher accuracy.

OSRM	Individual	TA	RegMean	Fisher	TIES	EMR	Avg.
No	**88.05**	70.04	67.88	64.88	61.32	81.18	72.22
Yes	87.87	**76.59**	**69.81**	**71.86**	**66.65**	**83.32**	**76.02**

Table 12: Averaged performance (%) across eight tasks of T5-large. "Individual" refers to the metrics of each fine-tuned model on the dataset on which it was trained. **Bold** indicates a higher accuracy.

OSRM	Individual	TA	RegMean	Fisher	TIES	EMR	Avg.
No	86.38	52.96	69.92	48.23	55.49	81.02	65.67
Yes	**87.01**	**55.79**	**73.68**	**56.13**	**59.74**	**82.61**	**69.16**

Table 13: Averaged performance (%) across eight tasks of Llama3.2-1B. "Individual" refers to the metrics of each fine-tuned model on the dataset on which it was trained. **Bold** indicates a higher accuracy.

OSRM	Individual	TA	RegMean	TIES	EMR	Avg.
No	84.42	65.68	41.19	**61.06**	77.85	66.04
Yes	**85.52**	**68.58**	**43.57**	59.64	**78.77**	**67.22**

Table 14: Averaged performance (%) across eight tasks of Llama3.2-3B. "Individual" refers to the metrics of each fine-tuned model on the dataset on which it was trained. **Bold** indicates a higher accuracy.

OSRM	Individual	TA	RegMean	TIES	EMR	Avg.
No	87.54	73.22	45.53	54.73	83.43	68.89
Yes	**88.33**	**74.36**	**45.77**	**57.28**	**83.47**	**69.84**

Table 15: Averaged performance (%) across eight tasks of Llama3-8B. "Individual" refers to the metrics of each fine-tuned model on the dataset on which it was trained. **Bold** indicates a higher accuracy.

OSRM	Individual	TA	TIES	Avg.
No	87.94	63.28	**53.01**	68.08
Yes	**88.77**	**72.75**	52.69	**71.4**

Table 16: Extension to merging existing LoRA modules. "Post" indicates applying OSRM *after* fine-tuning (c.f. Section 5.4). Performance (%) is averaged across eight tasks of RoBERTa-large. "Individual" refers to the metrics of each fine-tuned model on the dataset on which it was trained.

OSRM	Individual	TA	RegMean	Fisher	TIES	EMR	Avg.
No	88.05	70.04	67.88	64.88	61.32	81.18	72.22
Post	35.07	35.26	35.46	35.11	35.17	35.14	35.2