Representation Surgery for Multi-Task Model Merging

Enneng Yang [1]   Li Shen [2] [3]   Zhenyi Wang [4]   Guibing Guo [1]   Xiaojun Chen [5]   Xingwei Wang [1]   Dacheng Tao [6]

Abstract

Multi-task learning (MTL) compresses the information from multiple tasks into a unified backbone to improve computational efficiency and generalization. Recent work directly merges multiple independently trained models to perform MTL instead of collecting their raw data for joint training, greatly expanding the application scenarios of MTL. However, by visualizing the representation distribution of existing model merging schemes, we find that the merged model often suffers from the dilemma of *representation bias*. That is, there is a significant discrepancy in the representation distribution between the merged and individual models, resulting in poor performance of merged MTL. In this paper, we propose a *representation surgery* solution called "`Surgery`" to reduce representation bias in the merged model. Specifically, Surgery is a lightweight task-specific module that takes the representation of the merged model as input and attempts to output the biases contained in the representation from the merged model. We then designed an unsupervised optimization objective that updates the Surgery module by minimizing the distance between the merged model's representation and the individual model's representation. Extensive experiments demonstrate significant MTL performance improvements when our Surgery module is applied to state-of-the-art (SOTA) model merging schemes. The code is available at https://github.com/EnnengYang/RepresentationSurgery.

1. Introduction

Multi-task learning (MTL) utilizes a shared backbone to accommodate knowledge from multiple tasks simultaneously (Caruana, 1997; Vandenhende et al., 2021). The MTL model is very attractive when considering model efficiency because it does not require saving a copy of the parameters for each task. Taking advantage of this, MTL have been widely used in computer vision (Sener & Koltun, 2018; Liu et al., 2019; Sun et al., 2020; Liu et al., 2021a; Chen et al., 2022; Yang et al., 2023), natural language processing (Collobert & Weston, 2008; Dong et al., 2015; Liu et al., 2016), recommendation systems (Ma et al., 2018; Hadash et al., 2018; Pan et al., 2019; Tang et al., 2020; Wang et al., 2023; Song et al., 2024), robotics (Deisenroth et al., 2014; Shridhar et al., 2023) and other fields (He & Lawrence, 2011; Ishihara et al., 2021; Dang et al., 2022; Wu et al., 2023). However, MTL relies on the paradigm of "collecting data first and then jointly training". This usually involves high data management costs and the risk of data privacy leakage. In addition, training the MTL model simultaneously also lacks flexibility because, when new tasks come, they need to be re-trained with all old tasks, which requires extra cost.

Recently, "model fusion/merging" has emerged in the machine learning community (Wortsman et al., 2022; Matena & Raffel, 2022; Jin et al., 2023; Ilharco et al., 2023; Ortiz-Jimenez et al., 2023; Yadav et al., 2023; Zhang et al., 2023; Huang et al., 2023; Tang et al., 2024; Yang et al., 2024), which attempts to directly merge multiple independently trained or fine-tuned models to perform MTL (Li et al., 2023b), as shown in Fig. 3(a) to (b). In other words, model merging only requires trained model parameters for the respective tasks, that is, it no longer requires centralized management of MTL training data and joint training. In general, these methods greatly expand the application scenarios of MTL. Unfortunately, there is still a huge performance gap between the most advanced model merging based MTL (Ilharco et al., 2023; Jin et al., 2023; Yadav et al., 2023; Yang et al., 2024) and traditional MTL or individual models. This motivates us to further explore the existing problem in model merging and further solve it to close the above gap.

In this paper, we revisit several advanced model merging schemes (Ilharco et al., 2023; Yadav et al., 2023; Yang et al., 2024) from a **representation bias** perspective. Recall that

[1]Northeastern University, China. [2]Sun Yat-sen University, China. [3]JD Explore Academy, China. [4]University of Maryland, USA. [5]Shenzhen University, China. [6]Nanyang Technological University, Singapore. Correspondence to: Guibing Guo <guogb@swc.neu.edu.cn>, Li Shen <mathshenli@gmail.com>.

the goal of model merging is to make the merged single model have the representational capabilities of multiple individual trained models. To examine the extent to which the merged model retains the representational capabilities of the original individual model, we conduct extensive experiments across eight tasks, three architectures, and four representative model merging methods. Specifically, we visualize (see Fig. 1) the distribution of representations extracted by the individual model (blue points) and the distribution of representations extracted by the merged model (red points) through t-SNE (Van der Maaten & Hinton, 2008). We observe: (i) There is a clear discrepancy between the two distributions, and this "representation bias" exists *across tasks, across architectures, and across model merging methods*. (ii) From Weight Averaging to Task Arithmetic (Ilharco et al., 2023) / Ties-Merging (Yadav et al., 2023) to AdaMerging (Yang et al., 2024), the discrepancy between the two distributions is decreasing, which also corresponds to the performance improvement of the merged model. This indicates that the "representation bias" problem is one of the biggest obstacles to model merging, and it also directly causes the poor performance of the merged MTL model.

To solve the "representation bias" problem in model merging, we propose a novel **representation surgery** solution, abbreviated as "`Surgery`", to filter out representation bias from other tasks after model merging. Specifically, `Surgery` is a task-specific lightweight module that takes the representation extracted by the merged model as input and attempts to filter the bias contained in it, so that the filtered representation output is as close as possible to the feature representation of an individual model. Since raw training data is unavailable, we design an unsupervised surrogate objective to update parameters in the `Surgery` module, which minimizes the distance between the representation after surgery and the representation of the individual model. Our `Surgery` is completely orthogonal to existing model merging solutions, and it can be incorporated into any model merging method to solve its representation bias problem. We conduct extensive experiments on eight tasks and three architectures, and the results show that when our `Surgery` module is applied to several advanced model merging schemes, the performance of the merged MTL model can be significantly improved.

The main contributions of this paper are three-fold:

- We revisit SOTA model merging methods and identify for the first time the "representation bias" problem (which exists across tasks, architectures, and merging methods) as a major cause of poor MTL performance.
- We propose a novel "representation surgery" approach, called `Surgery`, to solve "representation bias" in the merged model. Moreover, the `Surgery` module is suitable for any existing model merging algorithm.

- We conduct extensive experiments to verify that our `Surgery` scheme can effectively alleviate representation bias and significantly improve MTL performance.

2. Related Work

2.1. Model Merging for Multi-Task Learning

Model merging is to merge multiple individual models into one (Li et al., 2023b). It has two main application scenarios: First, merge models trained on the *same task* to improve the accuracy or generalization of the final model (Gupta et al., 2020; Cha et al., 2021; Wortsman et al., 2022; Lu et al., 2022; Li et al., 2023a). Second, merge multiple models trained on *different tasks* to perform MTL (Ilharco et al., 2023; Yadav et al., 2023; Ramé et al., 2023; Zhang et al., 2023; Stoica et al., 2023; Huang et al., 2023; Yang et al., 2024), which is the focus of this paper. We further divide model merging into two stages: before and during merging.

(i) The main concern *before merging* is how to provide more favorable preconditions for model merging, such as linearization or orthogonalization. Specifically, (Ortiz-Jimenez et al., 2023) independently fine-tunes each task in the Tangent space (Jacot et al., 2018) of the pre-trained model and demonstrates that this helps decouple the weight space from the input space, leading to better model merging. Similarly, Linearization-LoRA (Tang et al., 2024) linearly fine-tunes some LoRA modules (Hu et al., 2022) in Tangent space. In addition, Task Arithmetic (Ilharco et al., 2023) pointed out that the orthogonality between task vectors is one of the conditions for successful model merging.

(ii) The main focus *during merging* is how to mitigate interference and conflicts between models (Ilharco et al., 2023; Yadav et al., 2023; Yu et al., 2023; Peña et al., 2023; Jin et al., 2023; Ramé et al., 2023; Zhang et al., 2023; Yang et al., 2024). For example, Ties-Merging (Yadav et al., 2023) eliminates the problem of parameter sign conflicts during model merging. DARE (Yu et al., 2023) removes a large number of useless neuron updates and then scales the neurons for merging. ReBasin (Ainsworth et al., 2023; Peña et al., 2023) rearranges and aligns the neurons of multiple models to establish connectivity paths between multiple models. Fisher-Merging (Matena & Raffel, 2022) performs weighted merging utilizing the importance of each parameter through the Fisher information matrix (Fisher, 1922). RegMean (Jin et al., 2023) reweights and linearly combines rows in weight matrices based on statistics from training data. Concrete (Tang et al., 2023) finds a shared subspace between multiple tasks for model merging. AdaMerging (Yang et al., 2024) leverages unlabeled test data to automatically learn a set of task- or layer-level model merging coefficients.

While existing methods predominantly concentrate on merging in weight space, they often neglect a crucial concern

stemming from weight merging–the *representation bias*. A substantial disparity emerges in the *representation space* between the merged model and individually-trained models. In contrast, our surgery method addresses this gap, aiming to minimize the representation discrepancy. Moreover, our approach operates in the *representation space*, offering a complementary and orthogonal perspective to traditional weight-space merging methods. Consequently, our method can be seamlessly integrated with them.

2.2. Traditional Multi-Task Learning

MTL usually faces negative transfer (Caruana, 1997; Vandenhende et al., 2021; Zhang et al., 2022). Existing work mainly solves the negative transfer problem from two directions: architecture and optimization. Specifically, (i) The classic SharedBottom (Caruana, 1997) backbone performs poorly when tasks are not highly correlated. Advanced *architectures* primarily alleviate the phenomenon of negative transfer through modularization (Ma et al., 2018; 2019; Tang et al., 2020), sparsification (Sun et al., 2020; Liu et al., 2019), and soft sharing (Misra et al., 2016; Gao et al., 2020) of the backbone. (ii) Other work alleviates task interference from an *optimization* perspective. For example, (Kendall et al., 2018; Sener & Koltun, 2018; Liu et al., 2019; 2022; Hu et al., 2023) try to optimize the weight for each task loss. (Chen et al., 2020; Yu et al., 2020; Liu et al., 2021a; Wang & Tsvetkov, 2021; Javaloy & Valera, 2022; Navon et al., 2022) resolve multi-task gradient direction or sign conflicts. (Chen et al., 2018; Liu et al., 2021b; He et al., 2022; Yang et al., 2023) try to eliminate the dominance of the learning rate or gradient. Distinct from these existing traditional MTL approaches that concentrate on loss weight or gradient space to tackle the negative transfer issue, our method improves MTL performance by addressing the representation bias problem within the representation space in model merging based MTL, providing a novel and orthogonal perspective.

3. Representation Bias in Model Merging

We first give the notation and definition in Sec. 3.1. Next, we revisit existing model merging schemes in Sec. 3.2 and point out their existing "representation bias" issues.

3.1. Preliminaries

Notations: Denote the neural network model $f : \mathcal{X} \times \Theta \mapsto \mathcal{Y}$, the parameter is $\theta \in \Theta \in \mathbb{R}^n$, the input is $\mathbf{x}_i \in \mathcal{X} \in \mathbb{R}^d$, and the output is $\mathbf{y}_i \in \mathcal{Y} \in \mathbb{R}^c$. Among them, n represents the number of parameters, d represents the dimension of the input data, and c represents the number of output classes. Considering T independently fine-tuned models f_{θ_t} (where $t \in \{1, 2, \ldots, T\}$), f_{θ_t} is well trained on their respective training data $\mathcal{D}_{tr}^t(\mathcal{X}, \mathcal{Y})$. Without loss of generality, it is usually assumed that $\{f_{\theta_t}\}_{t=1}^T$ are all fine-

tuned on a popular backbone f_{θ_0} (i.e., the pre-trained weight is θ_0), such as ResNet (He et al., 2016), ViT (Dosovitskiy et al., 2021) or BERT (Devlin et al., 2019).

Problem Setup: Model merging is to merge the weights $\{\theta_t\}_{t=1}^T$ to obtain a final weight θ_{mtl}^m, so that $f_{\theta_{mtl}^m}$ can simultaneously perform the tasks $\{1, 2, \ldots, T\}$, where m is a model merging method. Additionally, we are not allowed to access raw training data $\mathcal{D}_{tr}^t$ when merging models. Formally, we expect the loss of the merged model $f_{\theta_{mtl}^m}$ to be as small as possible on the test dataset $\{\mathcal{D}_{te}^t\}_{t=1}^T$ of all tasks, i.e., $\min \frac{1}{T} \sum_{t=1}^T \sum_{i=1}^{|\mathcal{D}_{te}^t|} \frac{1}{|\mathcal{D}_{te}^t|} \ell(f_{\theta_{mtl}^m}(\mathbf{x}_i), \mathbf{y}_i)$, where $\ell(\cdot)$ is the loss function, such as the cross-entropy loss.

Representative Model Merging Solutions: ① The simplest merging scheme is **Weight Averaging**, which directly averages the parameters $\{\theta_t\}_{t=1}^T$ of multiple models: $\theta_{mtl}^m = \frac{1}{T} \sum_{t=1}^T \theta_t$. However, the performance of simple weighted averaging is often unsatisfactory. ② Recently, **Task Arithmetic** (Ilharco et al., 2023) record the parameter difference ($\tau_t = \theta_t - \theta_0$) obtained by subtracting the pre-trained model f_{θ_0} from the fine-tuned model f_{θ_t} as the *task vector*. Then, it merges the task vectors $\{\tau_t\}_{t=1}^T$ of multiple tasks and combines them into the pre-trained weight θ_0: $\theta_{mtl}^m = \theta_0 + \lambda \sum_{t=1}^T \tau_t$. By simply adjusting the hyperparameter λ, Task Arithmetic can achieve better model merging performance than Weight Averaging. ③ On the basis of task vector, **Ties-Merging** (Yadav et al., 2023) further proposes three operations: TRIM, ELECT SIGN and MERGE to eliminate the symbol conflict problem in task vectors. We combine these three operations and call them $\phi(\cdot)$. Finally, the Ties-Merging merge is expressed as: $\theta_{mtl}^m = \theta_0 + \lambda \sum_{t=1}^T \phi(\tau_t)$. ④ Furthermore, **AdaMerging** (Yang et al., 2024) adaptively learns a set of task-level or layer-level merging coefficients for Task Arithmetic or Ties-Merging, which significantly improves the MTL performance of model merging. Task-wise and layer-wise AdaMerging are expressed as: $\theta_{mtl}^m = \theta_0 + \lambda_t \sum_{t=1}^T \tau_t$ and $\theta_{mtl}^m = \{\theta_0^l + \lambda_t^l \sum_{t=1}^T \tau_t^l\}_{l=1}^L$ (where L is the number of layers in model $f_{\theta_{mtl}^m}$) respectively. For other mature model merging solutions, please refer to Sec. 2.1.

3.2. Revisiting Representation Bias

This section revisits existing model merging schemes and finds that they suffer from a "representation bias" problem, that is, the feature representations extracted by the merged model differ from those of the individual models.

3.2.1. SETTINGS

Without loss of generality, we perform analysis on the following representative model merging methods, tasks/datasets, and architectures. (i) **Methods**: We analyze the four model merging methods mentioned in Sec. 3.1:

(a) Weight Averaging on ViT-B/32

(b) Task Arithmetic on ViT-B/32

(c) AdaMerging on ViT-B/32
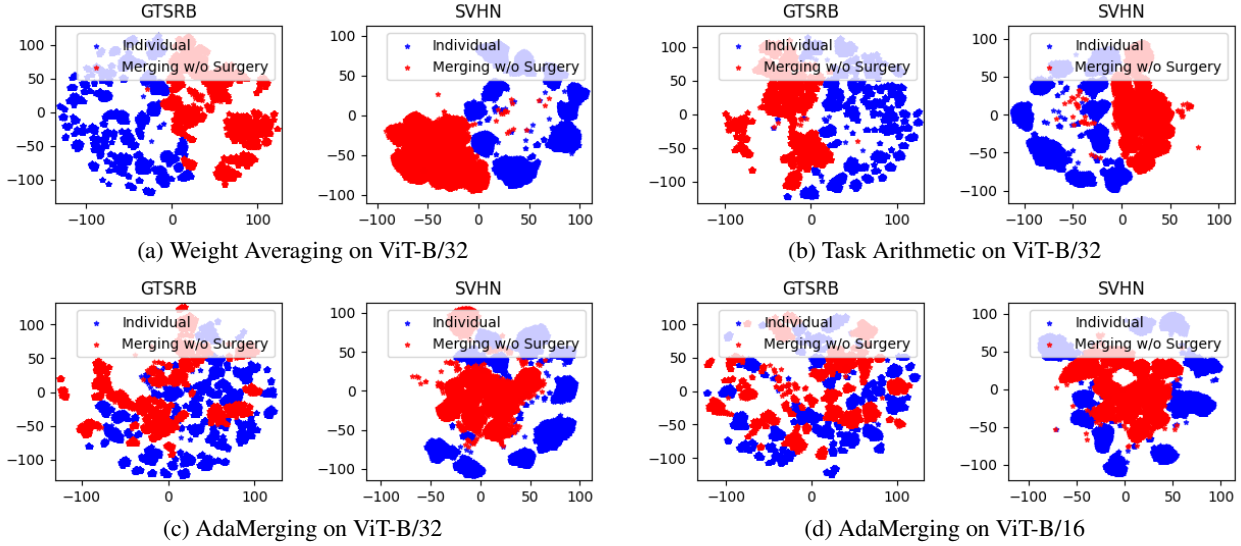
(d) AdaMerging on ViT-B/16

Figure 1. Visualization of the distribution of representations extracted by the merged model (red) for the existing model merging schemes and representations extracted by the individual model (blue). We observe that there is a clear distribution discrepancy between the two.
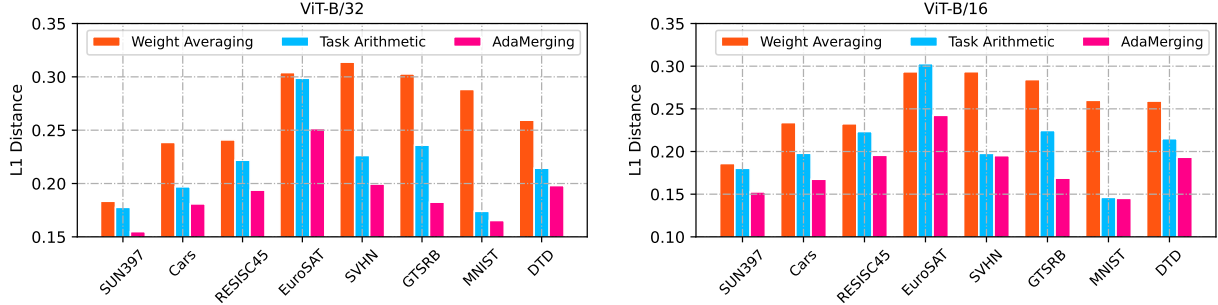

Figure 2. The L_1 distance (or "representation bias" in Eq. 1) between representations extracted by the merged model using various model merging methods (i.e., Weight Averaging, Task Arithmetic and AdaMerging) and representations extracted by individual models.

Weight Averaging, Task Arithmetic, Ties-Merging and AdaMerging. (ii) ***Tasks***: We follow Task Arithmetic (Ilharco et al., 2023) and use the following eight datasets as eight tasks for model merging: SUN397, Cars, RESISC45, EuroSAT, SVHN, GTSRB, MNIST, DTD. We provide a detailed dataset description in Appendix A. (iii) ***Architectures***: We merge eight models into one model and experiment with three ViT architectures (Dosovitskiy et al., 2021) with different parameter scales: ViT-B/32, ViT-B/16, and ViT-L/14.

To explore the reasons for the performance gap between the merged model $f_{\theta_{mtl}^m}$ and individual models $\{f_{\theta_t}\}_{t=1}^T$, we tried to visualize the distribution of the feature representations they extracted. Specifically, we first input all the test samples $D_{te}^t(\mathcal{X}, \mathcal{Y})$ of each task t into the models $f_{\theta_{mtl}^m}$ and f_{θ_t} respectively, and record their extracted feature representations (i.e., the ***final layer*** before task-specific head/heads of the individual/merged models) as $\mathcal{Z}_t^{mtl} \in \mathbb{R}^{N \times k}$ and $\mathcal{Z}_t^{ind} \in \mathbb{R}^{N \times k}$ respectively. Among them, $N = |D_{te}^t|$ represents the amount of data, and k represents the dimension of the feature (e.g., ViT-B/32 and ViT-B/16 are 512, and ViT-L/14 is 768). Next, we can map the high-

dimensional data $\mathcal{Z}_t^{mtl}$ and $\mathcal{Z}_t^{ind}$ to a lower-dimensional space (2-dimensional) through t-SNE tool and visualize it.

3.2.2. "REPRESENTATION BIAS" PROBLEM

We find that "representation bias" exists *across tasks, across merging methods, and across architectures* [1]. Specifically, as shown in Fig. 1, we have the following observations: (i) "Representation bias" is present across various tasks. As shown in Fig. 1(a), on the two tasks of GTSRB and SVHN, the feature representation distributions of the merged model (red points) and the individual model (blue points) are quite different. As shown in Appendix B, this representation bias problem also exists in the other six tasks. (ii) "Representation bias" persists across different merging methods. Comparing Fig. 1(a), (b), (c), we find that the phenomenon of inconsistent representation distribution between the merged

---

[1] Due to space limitations, we only show two tasks (GTRSB and SVHN), three merging methods (Weight Averaging, Task Arithmetic and AdaMerging), and two architectures (ViT-B/32 and ViT-B/16) in the main text. The comprehensive results are presented in Appendix B (A summary is given in Tab. 11).
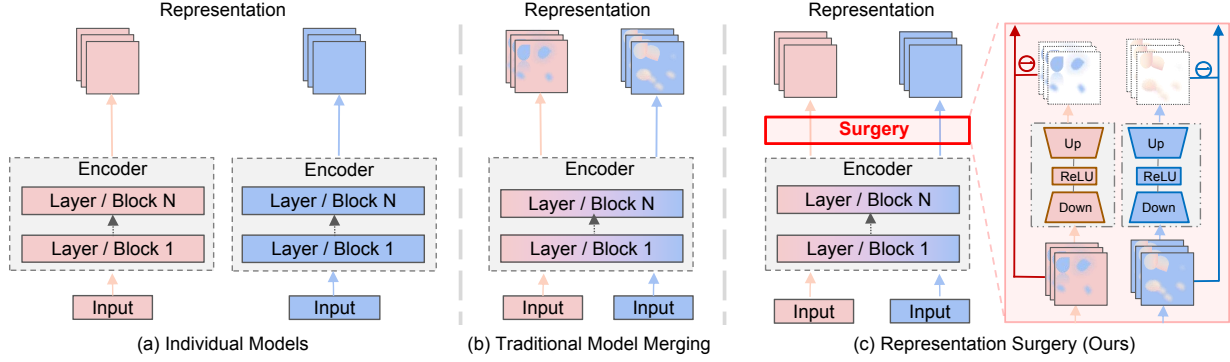
Figure 3. Representation Surgery for Multi-Task Model Merging. (a) Multiple individual trained models. (b) Traditional model merging schemes (e.g., Task Arithmetic (Ilharco et al., 2023), Ties-Merging (Yadav et al., 2023), AdaMerging (Yang et al., 2024), etc.) merge multiple individual models into one. However, they usually suffer from the "representation bias" problem. (c) "Representation surgery" solution is proposed in this paper. It is a task-specific lightweight module used to solve the representation bias problem.

model and the individual model exists in Weight Averaging, Task Arithmetic and AdaMerging. (iii) "Representation bias" exists across diverse architectures. Comparing Fig. 1(c) and (d), we find that the representation bias problem holds true in both ViT-B/32 and ViT-B/16, and the results in Appendix B show that it also holds true in ViT-L/14.

This phenomenon encourages us to think further. *Is representation bias the key factor limiting the performance improvement of model merging?* We give a "yes" answer. **First**, a large amount of previous experimental experience (Ilharco et al., 2023; Yadav et al., 2023; Yang et al., 2024) shows that in terms of model merging performance, AdaMerging > Task Arithmetic > Weight Averaging, the results are shown in Tab. 1, Tab. 2 and Tab. 3 (in Appendix B). For example, in Tab. 1, the average accuracy of AdaMerging, Task Arithmetic, and Weight Averaging on eight tasks are 80.1%, 69.1% and 65.8% respectively. **Second**, by directly observing the discrepancy between the red and blue distributions in Fig. 1(a)(b)(c), we can see the phenomenon of AdaMerging < Task Arithmetic < Weight Averaging. Furthermore, to quantitatively discuss the distance (or *representation bias*) between the representation of each merged model (e.g., by AdaMerging, Task Arithmetic, Weight Averaging) and that of individual models, we calculated the L_1 distance between feature representations $\mathcal{Z}_t^{mtl}$ and $\mathcal{Z}_t^{ind}$ in Sec. 3.2.1 on the test dataset $D_{te}^t(\mathcal{X}, \mathcal{Y})$ of each task t, i.e.,

$$\text{representation bias: } d_t = \frac{1}{k} \frac{1}{|D_{te}^t|} \|\mathcal{Z}_t^{mtl} - \mathcal{Z}_t^{ind}\|_1. \quad (1)$$

As shown in Fig. 2, we clearly observe that AdaMerging has a smaller representation bias between the merged model and the individual models compared to the other two model merging methods, i.e., Task Arithmetic and Weight Averaging. Meanwhile, Task Arithmetic also has a smaller representation bias than Weight Averaging.

The above analysis shows that representation bias poses significant challenges when completing MTL based on model

merging. The performance of the merged model improves when the representation bias decreases. This motivates us to seek a solution to alleviate "representation bias" problem in model merging based MTL methods.

4. Representation Surgery for Model Merging

In this section, we propose a simple yet effective method-agnostic representation surgery scheme to alleviate the representation bias problem pointed out in the above section.

4.1. Optimization Objective

We directly take the representation bias (i.e., Eq. 1) as the optimization goal of representation surgery, that is, minimizing the distance between the representation extracted by the merged model and the representation extracted by the individual model. In other words, as shown in Fig. 3(c), our representation surgery is to filter out the representation bias $\Phi_t(\mathcal{Z}_t^{mtl})$ of the representation $\mathcal{Z}_t^{mtl}$ extracted by the merged model $f_{\theta_{mtl}^m}$, thereby making it (i.e., $\mathcal{Z}_t^{mtl} - \Phi_t(\mathcal{Z}_t^{mtl})$) closer to the representation $\mathcal{Z}_t^{ind}$ of the individual model f_{θ_t}. Formally, the optimization problem of representation surgery is as follows:

$$\underset{\{\theta_{\Phi_1}, \theta_{\Phi_2}, \dots, \theta_{\Phi_T}\}}{\arg\min} \sum_{t=1}^{T} \frac{1}{|D_{te}^t|} \|\hat{\mathcal{Z}}_t^{mtl} - \mathcal{Z}_t^{ind}\|_1 \quad (2)$$
$$\text{s.t. } \hat{\mathcal{Z}}_t^{mtl} = \mathcal{Z}_t^{mtl} - \Phi_t(\mathcal{Z}_t^{mtl}),$$

where $\Phi_t(\cdot)$ is a task-private lightweight module, which can be an arbitrary implementation (such as multiple fully connected layers, etc.). Without loss of generality, in this paper, we implement it as an Adapter (Houlsby et al., 2019)-like structure, that is,

$$\Phi_t(\mathcal{Z}_t^{mtl}) = \mathbf{W}_{up} \cdot \text{ReLU}(\mathbf{W}_{down} \cdot \mathcal{Z}_t^{mtl\top}) \quad (3)$$

where $\mathbf{W}_{up} \in \mathbb{R}^{k \times r}$, $\mathbf{W}_{down} \in \mathbb{R}^{r \times k}$ represent two learnable matrices, i.e., $\theta_{\Phi_t} = \{\mathbf{W}_{up}, \mathbf{W}_{down}\}$, ReLU($\cdot$) is a

(a) Weight Averaging on ViT-B/32

(b) Task Arithmetic on ViT-B/32

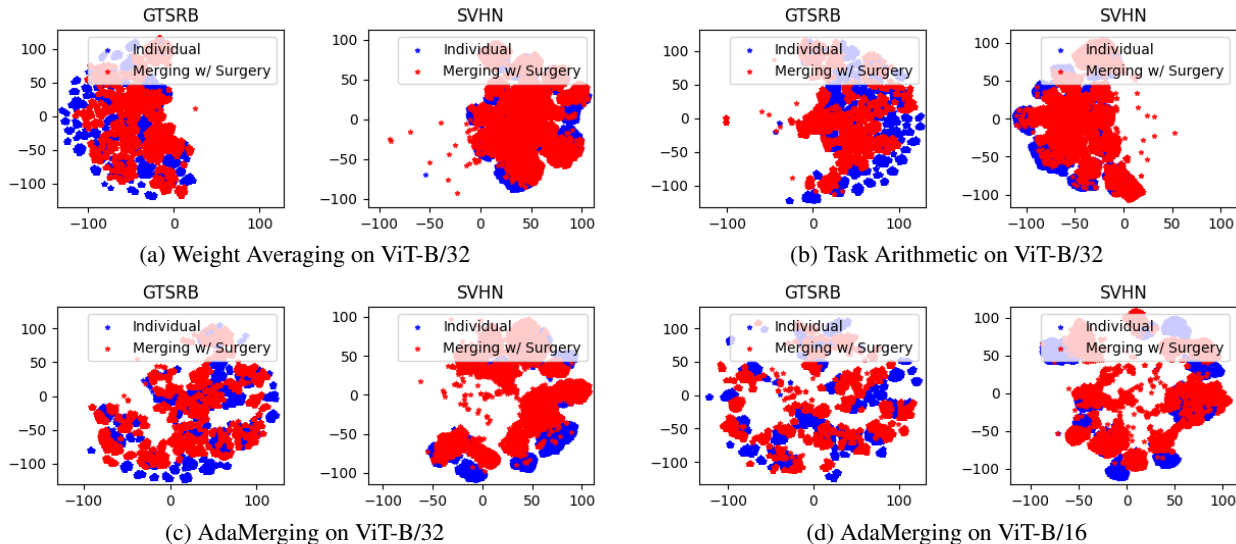(c) AdaMerging on ViT-B/32

(d) AdaMerging on ViT-B/16

Figure 4. Visualization of the distribution of features extracted by the merged model **after performing the representation surgery** (red) and features extracted by the individual model (blue). We observe that the distributions of the two are relatively close.
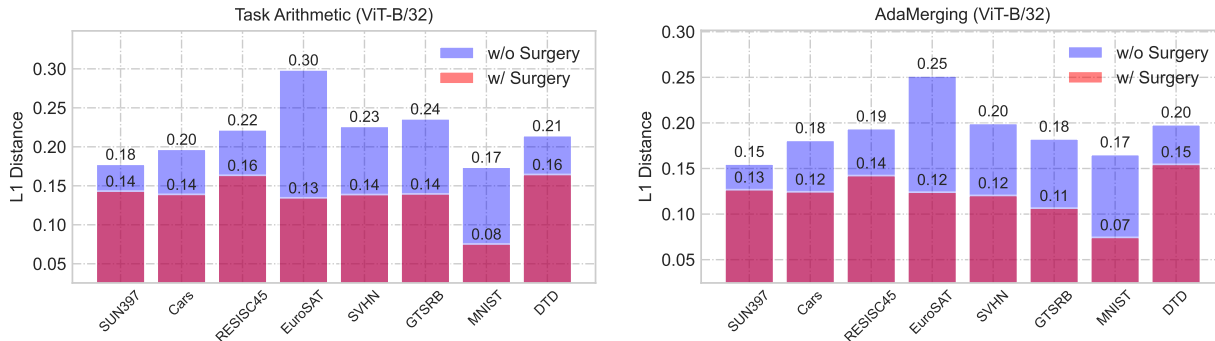

Figure 5. Visualization of the L_1 distance (or "representation bias" in Eq. 1) of the representation of the merged model **with** (red) and **without** (blue) representation surgery versus the individual model.

nonlinear activation function, k is the dimension of representation as mentioned in Sec. 3.1, and r is a hyperparameter, also called rank, which we set to 16 by default.

Overall, our `Surgery` scheme does not rely on any labeled training data, but utilizes unlabeled test data $\{\mathcal{D}_{te}^t(\mathcal{X}, \mathcal{Y})\}_{t=1}^T$ and individual models $\{f_{\theta_t}\}_{t=1}^T$ as a self-supervised signal to train the `Surgery` module's parameters $\{\theta_{\Phi_1}, \theta_{\Phi_2}, \ldots, \theta_{\Phi_T}\}$. In addition, as mentioned in Appendix B.3, the number of parameters increased by our surgery module is very minor (0.01%) compared to the number of parameters that need to be merged into the model.

4.2. Discussion

What is the positioning of "representation surgery" within the domain of model merging? The representation surgery proposed in this paper is orthogonal to existing model merging techniques in the following two aspects:

• "Seek common v.s. reserve differences": All previous

model merging methods focus on how to merge information shared by multiple tasks, that is, a process of *"seeking common"*. On the contrary, the representation surgery proposed in this paper adds a task-specific module after the merged model to store task-private information, so it achieves ***seeking common while reserving differences***.

• "Before-merging v.s. during-merging v.s. post-merging": All previous model merging methods focus on how to create better merging conditions *before merging* or mitigate conflicts *during merging*, as discussed in Sec. 2. In contrast, our representation surgery focuses on how to solve the "representation bias" problem ***post-merging***.

In summary, the representation surgery proposed in this paper is complementary to existing model merging schemes.

Why does "representation surgery" work? As mentioned in Sec. 4.1, the goal of the representation surgery is to reduce the discrepancy in the representation distributions between the merged and individual models. Comparing the discrepancy in representation distributions in Fig. 1 (w/o

Table 1. Multi-task performance when merging ViT-B/32 models on eight tasks.

Method	SUN397	Cars	RESISC45	EuroSAT	SVHN	GTSRB	MNIST	DTD	Avg.
Pretrained	62.3	59.7	60.7	45.5	31.4	32.6	48.5	43.8	48.0
Individual	75.3	77.7	96.1	99.7	97.5	98.7	99.7	79.4	90.5
Traditional MTL	73.9	74.4	93.9	98.2	95.8	98.9	99.5	77.9	88.9
Weight Averaging	65.3	63.4	71.4	71.7	64.2	52.8	87.5	50.1	65.8
Fisher Merging (Matena & Raffel, 2022)	68.6	69.2	70.7	66.4	72.9	51.1	87.9	59.9	68.3
RegMean (Jin et al., 2023)	65.3	63.5	75.6	78.6	78.1	67.4	93.7	52.0	71.8
Task Arithmetic (Ilharco et al., 2023)	55.2	54.9	66.7	78.9	80.2	69.7	97.3	50.4	69.1
Ties-Merging (Yadav et al., 2023)	65.0	64.4	74.8	77.4	81.2	69.3	96.5	54.5	72.9
Concrete TA (Tang et al., 2023)	62.5	61.1	76.0	95.7	91.0	81.9	98.5	51.9	77.3
Concrete AM (Tang et al., 2023)	67.8	70.0	87.5	96.0	91.6	96.7	98.7	63.8	84.0
TW AdaMerging (Yang et al., 2024)	58.0	53.2	68.8	85.7	81.1	84.4	92.4	44.8	71.1
AdaMerging (Yang et al., 2024)	64.5	68.1	79.2	93.8	87.0	91.9	97.5	59.1	80.1
Weight Averaging w/ Surgery (Ours)	67.6	64.6	85.8	96.8	76.9	82.9	97.8	67.3	80.0
Task Arithmetic w/ Surgery (Ours)	63.8	59.9	83.3	97.9	87.0	87.0	98.6	69.4	80.9
Ties-Merging w/ Surgery (Ours)	69.8	66.1	87.3	97.5	86.7	87.6	98.5	71.6	83.1
TW AdaMerging w/ Surgery (Ours)	63.9	57.6	84.2	98.2	87.6	92.7	98.0	66.8	81.1
AdaMerging w/ Surgery (Ours)	69.8	71.0	88.9	98.1	91.7	96.5	98.8	73.6	86.1
AdaMerging w/ Surgery[†] (Ours)	71.2	72.0	92.3	99.0	92.2	97.9	99.0	76.1	87.5

[†] means that the rank (i.e., Eq. 3) of the surgery module is set to 64, and the other defaults are 16.

surgery) and Fig. 4 (w/ surgery), we can observe that the distributions of the merged and individual models in Fig. 4 are closer (i.e., higher overlap).

Going one step further, we quantify the distance between the two distributions. Specifically, similar to the setting in Sec. 3.2.2, we measure the L_1 distance between two sets (i.e., $\mathcal{Z}_t^{mtl}$ and $\mathcal{Z}_t^{ind}$) of feature representations by Eq. 1. As shown in Fig. 5, we observe a significant reduction in the L_1 distance for the merged model with the proposed representation surgery (red column) compared to the merged model without representation surgery (blue column). For example, on the EuroSAT dataset, Task Arithmetic based model merging reduces the L_1 distance from 0.30 to 0.13 after representation surgery, a relative reduction of 56%. These pieces of evidence suggest that the proposed representation surgery helps alleviate the representation bias problem.

5. Experiment

In this section, we describe our experimental setup and show performance comparisons. Due to the page limit, additional experimental setups and results are shown in the **Appendix**.

5.1. Experimental Setup

Datasets. Following the setup of Task Arithmetic (Ilharco et al., 2023), Ties-Merging (Yadav et al., 2023) and AdaMerging (Yang et al., 2024), we treat the following eight datasets as eight tasks to perform model merging: SUN397 (Xiao et al., 2016), Cars (Krause et al., 2013), RESISC45 (Cheng et al., 2017), EuroSAT (Helber et al., 2019), SVHN (Yuval, 2011), GTSRB (Stallkamp et al.,

2011), MNIST (LeCun, 1998), DTD (Cimpoi et al., 2014).

Baselines. We use the following *non-model merging methods* as reference baselines: Pretrained, Individual, Traditional MTL. Additionally, we compared the following *model fusion methods*: Weight Averaging, Fisher Merging (Matena & Raffel, 2022), RegMean (Jin et al., 2023), Task Arithmetic (Ilharco et al., 2023), Ties-Merging (Yadav et al., 2023), Concrete TA (Tang et al., 2023), Concrete AM (Tang et al., 2023), AdaMerging (Yang et al., 2024).

Architectures. Following Task Arithmetic (Ilharco et al., 2023) and AdaMerging (Yang et al., 2024), we merge ViT-type architectures (Dosovitskiy et al., 2021), including three different scale architectures of ViT-B/32, ViT-B/16 and ViT-L/14 from CLIP's (Radford et al., 2021) visual encoder.

5.2. Performance

Due to page limitations, we leave the results of ViT-B/16 in the Appendix B. The results on the ViT-B/32 and ViT-L/14 architectures are shown in Tab. 1 and Tab. 2. From these two tables, we have the following observations: (i) For the ***non-model merging baselines***, the Pretrained model performed the worst, and the Individual model and Traditional MTL achieved optimal and suboptimal performance, respectively. The reason why a pre-trained model is bad is that it does not utilize any task-relevant information. In contrast, Traditional MTL uses data from all tasks to train the model together, significantly improving efficiency. However, Traditional MTL may suffer from negative transfer (Zhang et al., 2022) problems and, therefore, is not as effective as individual models. (ii) For ***model merging baselines***, Weight Averaging is the simplest method. It directly averages multi-

Table 2. Multi-task performance when merging ViT-L/14 models on eight tasks.

Method	SUN397	Cars	RESISC45	EuroSAT	SVHN	GTSRB	MNIST	DTD	**Avg.**
Pretrained	66.8	77.7	71.0	59.9	58.4	50.5	76.3	55.3	64.5
Individual	82.3	92.4	97.4	100	98.1	99.2	99.7	84.1	94.2
Traditional MTL	80.8	90.6	96.3	96.3	97.6	99.1	99.6	84.4	93.5
Weight Averaging	72.1	81.6	82.6	91.9	78.2	70.7	97.1	62.8	79.6
Fisher Merging (Matena & Raffel, 2022)	69.2	88.6	87.5	93.5	80.6	74.8	93.3	70.0	82.2
RegMean (Jin et al., 2023)	73.3	81.8	86.1	97.0	88.0	84.2	98.5	60.8	83.7
Task Arithmetic (Ilharco et al., 2023)	73.9	82.1	86.6	94.1	87.9	86.7	98.9	65.6	84.5
Ties-Merging (Yadav et al., 2023)	76.5	85.0	89.3	95.7	90.3	83.3	99.0	68.8	86.0
Concrete TA (Tang et al., 2023)	74.6	86.2	89.0	96.7	93.6	93.4	99.1	66.9	87.4
Concrete AM (Tang et al., 2023)	77.8	91.2	92.1	97.0	94.4	97.9	99.0	79.5	91.1
AdaMerging (Yang et al., 2024)	79.0	90.3	90.8	96.2	93.4	98.0	99.0	79.9	90.8
Weight Averaging w/ Surgery (Ours)	73.7	83.9	92.0	98.4	82.4	86.3	98.7	71.9	85.9
Task Arithmetic w/ Surgery (Ours)	75.7	84.4	93.1	98.8	91.3	93.4	99.1	76.1	89.0
Ties-Merging w/ Surgery (Ours)	76.5	85.9	93.7	99.2	89.7	92.0	99.1	78.1	89.3
AdaMerging w/ Surgery (Ours)	80.3	90.8	94.3	98.2	94.1	98.7	99.2	82.5	92.3

ple model parameters. Naturally, its performance is also the worst. Fisher Merging and RegMean calculate the importance of parameters/models during merging, and there are obvious hints in terms of performance compared to Weight Averaging. In addition, Concrete TA, Concrete AM and Ties-Merging remove some neurons when merging models, thereby effectively alleviating the parameter conflict problem during merging, and the final performance is better than baselines such as Task Arithmetic. TW AdaMerging and AdaMerging automatically learn task-wise/layer-wise merging coefficients on the test set in an unsupervised manner and achieve good results. However, as our analysis in Sec. 5.3 shows, these model merging methods still suffer from the problem of "representation bias". (iii) Our proposed *representation surgery* helps alleviate the representation bias problem, and it is orthogonal to existing model merging schemes. When the proposed representation surgery scheme is used on Weight Averaging, Task Arithmetic, Ties-Merging, and AdaMerging, their performance has been greatly improved. For example, on ViT-B/32, the accuracy of Task Arithmetic without and with the proposed representation surgery is 69.1% and 80.9%, respectively. On the more advanced AdaMerging, the accuracy has also increased from 80.1% to 87.5% , which is very close to the 88.9% of Traditional MTL. On ViT-L/14, AdaMerging achieved an accuracy of 92.3% after representation surgery, which is also very close to the 93.5% of Traditional MTL.

5.3. Additional Results and Analysis

Rank in Surgery Module. As shown in Sec. 4.2, the representation surgery module can be regarded as used to save the task-private information of each task. Therefore, the capacity of this module has an impact on the accuracy of model merging. In this section, we try different ranks (i.e., $r \in \{4, 8, 16, 32, 64\}$ in Eq. 3) and observe the accuracy
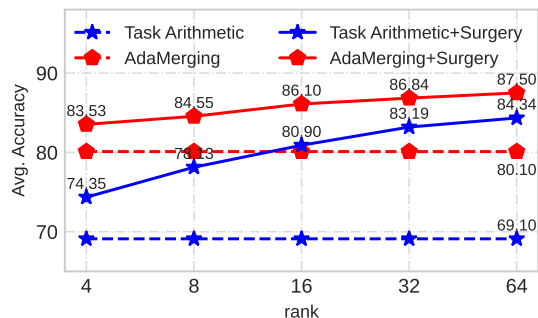

Figure 6. The average accuracy changes corresponding to different ranks in the surgery module under ViT-B/32 architecture.
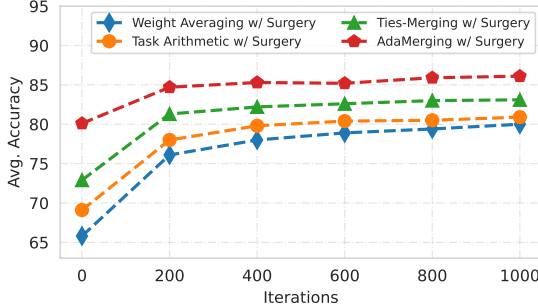

Figure 7. The average accuracy of model merging changes with the number of iterations on ViT-B/32.

changes. As shown in Fig. 6 and Tab. 8 in Appendix B, we consistently observed that as the rank increases, the accuracy of model merging also improves. For example, on AdaMerging, when the rank increases from 16 to 64, the average accuracy improves from 86.1% to 87.5%.

Training Step v.s. Avg. Accuracy. In Fig. 7, we show how the average accuracy of the merged model changes with the number of iterations of the representation surgery on the ViT-B/32. We observe that in the early stages of training (e.g., the first 200 iterations), the accuracy of the merged model improves rapidly. As the iterations continue, the accuracy improvement gradually flattens out.

6. Conclusion and Future Work

This paper first studies that the "representation bias" problem exists widely in model merging, and it exists across tasks, across model merging methods, and across architectures. Next, we propose a "representation surgery" scheme to alleviate the representation bias problem by reducing the difference between the merged model and the individual models. Finally, we have verified through a large number of experiments that existing model merging methods can effectively improve the performance of model merging using the proposed representation surgery. In the future, we plan to apply the proposed representation surgery to more model merging schemes and explore model merging from different architectures or initializations.

Acknowledgments

Impact Statement

Model merging based MTL provides an orthogonal perspective to perform multi-task learning. This paper discusses the common problem of representation bias in model merging based MTL methods and proposes representation surgery to alleviate the representation bias problem. This work has no ethical aspects as well as negative social consequences.

References

Ainsworth, S., Hayase, J., and Srinivasa, S. Git re-basin: Merging models modulo permutation symmetries. In *ICLR*, 2023.

Caruana, R. Multitask learning. *Machine learning*, 28: 41–75, 1997.

Cha, J., Chun, S., Lee, K., Cho, H.-C., Park, S., Lee, Y., and Park, S. Swad: Domain generalization by seeking flat minima. *NeurIPS*, 34:22405–22418, 2021.

Chen, X., Liu, T., Zhao, H., Zhou, G., and Zhang, Y.-Q. Cerberus transformer: Joint semantic, affordance and attribute parsing. In *CVPR*, pp. 19649–19658, 2022.

Chen, Z., Badrinarayanan, V., Lee, C.-Y., and Rabinovich, A. Gradnorm: Gradient normalization for adaptive loss balancing in deep multitask networks. In *ICML*, pp. 794–803. PMLR, 2018.

Chen, Z., Ngiam, J., Huang, Y., Luong, T., Kretzschmar, H., Chai, Y., and Anguelov, D. Just pick a sign: Optimizing deep multitask models with gradient sign dropout. In *NeurIPS*, 2020.

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

Cimpoi, M., Maji, S., Kokkinos, I., Mohamed, S., and Vedaldi, A. Describing textures in the wild. In *CVPR*, pp. 3606–3613, 2014.

Collobert, R. and Weston, J. A unified architecture for natural language processing: Deep neural networks with multitask learning. In *ICML*, pp. 160–167, 2008.

Dang, Q.-L., Xu, W., and Yuan, Y.-F. A dynamic resource allocation strategy with reinforcement learning for multimodal multi-objective optimization. *Machine Intelligence Research*, 19(2):138–152, 2022. ISSN 2731-538X.

Deisenroth, M. P., Englert, P., Peters, J., and Fox, D. Multitask policy search for robotics. In *ICRA*, pp. 3876–3881. IEEE, 2014.

Devlin, J., Chang, M., Lee, K., and Toutanova, K. BERT: pre-training of deep bidirectional transformers for language understanding. In *NAACL*, pp. 4171–4186. Association for Computational Linguistics, 2019.

Dong, D., Wu, H., He, W., Yu, D., and Wang, H. Multi-task learning for multiple language translation. In *ACL*, pp. 1723–1732, 2015.

Dosovitskiy, A., Beyer, L., Kolesnikov, A., Weissenborn, D., Zhai, X., Unterthiner, T., Dehghani, M., Minderer, M., Heigold, G., Gelly, S., Uszkoreit, J., and Houlsby, N. An image is worth 16x16 words: Transformers for image recognition at scale. *ICLR*, 2021.

Fisher, R. A. On the mathematical foundations of theoretical statistics. *Philosophical transactions of the Royal Society of London. Series A, containing papers of a mathematical or physical character*, 222(594-604):309–368, 1922.

Gao, Y., Bai, H., Jie, Z., Ma, J., Jia, K., and Liu, W. Mtl-nas: Task-agnostic neural architecture search towards general-purpose multi-task learning. In *CVPR*, pp. 11543–11552, 2020.

Gupta, V., Serrano, S. A., and DeCoste, D. Stochastic weight averaging in parallel: Large-batch training that generalizes well. In *ICLR*. OpenReview.net, 2020.

Hadash, G., Shalom, O. S., and Osadchy, R. Rank and rate: multi-task learning for recommender systems. In *RecSys*, pp. 451–454, 2018.

He, J. and Lawrence, R. A graphbased framework for multi-task multi-view learning. In *ICML*, pp. 25–32, 2011.

He, K., Zhang, X., Ren, S., and Sun, J. Deep residual learning for image recognition. In *CVPR*, pp. 770–778, 2016.

He, Y., Feng, X., Cheng, C., Ji, G., Guo, Y., and Caverlee, J. Metabalance: Improving multi-task recommendations via adapting gradient magnitudes of auxiliary tasks. *WWW*, pp. 2205–2215, 2022.

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

Houlsby, N., Giurgiu, A., Jastrzebski, S., Morrone, B., De Laroussilhe, Q., Gesmundo, A., Attariyan, M., and Gelly, S. Parameter-efficient transfer learning for nlp. In *ICML*, pp. 2790–2799. PMLR, 2019.

Hu, E. J., Shen, Y., Wallis, P., Allen-Zhu, Z., Li, Y., Wang, S., Wang, L., and Chen, W. LoRA: Low-rank adaptation of large language models. In *ICLR*, 2022.

Hu, Y., Xian, R., Wu, Q., Fan, Q., Yin, L., and Zhao, H. Revisiting scalarization in multi-task learning: A theoretical perspective. In *NeurIPS*, 2023.

Huang, C., Liu, Q., Lin, B. Y., Pang, T., Du, C., and Lin, M. Lorahub: Efficient cross-task generalization via dynamic lora composition. *arXiv preprint arXiv:2307.13269*, 2023.

Ilharco, G., Ribeiro, M. T., Wortsman, M., Schmidt, L., Hajishirzi, H., and Farhadi, A. Editing models with task arithmetic. In *ICLR*, 2023.

Ishihara, K., Kanervisto, A., Miura, J., and Hautamaki, V. Multi-task learning with attention for end-to-end autonomous driving. In *CVPR*, pp. 2902–2911, 2021.

Jacot, A., Gabriel, F., and Hongler, C. Neural tangent kernel: Convergence and generalization in neural networks. *NeurIPS*, 31, 2018.

Javaloy, A. and Valera, I. Rotograd: Gradient homogenization in multitask learning. In *ICLR*, 2022.

Jin, X., Ren, X., Preotiuc-Pietro, D., and Cheng, P. Dataless knowledge fusion by merging weights of language models. In *ICLR*, 2023.

Kendall, A., Gal, Y., and Cipolla, R. Multi-task learning using uncertainty to weigh losses for scene geometry and semantics. In *CVPR*, pp. 7482–7491. IEEE Computer Society, 2018.

Kingma, D. P. and Ba, J. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

Krause, J., Stark, M., Deng, J., and Fei-Fei, L. 3d object representations for fine-grained categorization. In *ICCV workshops*, pp. 554–561, 2013.

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

Li, T., Huang, Z., Tao, Q., Wu, Y., and Huang, X. Trainable weight averaging: Efficient training by optimizing historical solutions. In *ICLR*, 2023a.

Li, W., Peng, Y., Zhang, M., Ding, L., Hu, H., and Shen, L. Deep model fusion: A survey. *arXiv preprint arXiv:2309.15698*, 2023b.

Li, W.-H. and Bilen, H. Knowledge distillation for multi-task learning. In *Computer Vision–ECCV 2020 Workshops: Glasgow, UK, August 23–28, 2020, Proceedings, Part VI 16*, pp. 163–176. Springer, 2020.

Liu, B., Liu, X., Jin, X., Stone, P., and Liu, Q. Conflict-averse gradient descent for multi-task learning. *NeurIPS*, 34:18878–18890, 2021a.

Liu, L., Li, Y., Kuang, Z., Xue, J.-H., Chen, Y., Yang, W., Liao, Q., and Zhang, W. Towards impartial multi-task learning. In *ICLR*, 2021b.

Liu, P., Qiu, X., and Huang, X. Recurrent neural network for text classification with multi-task learning. In *IJCAI*, pp. 2873–2879, 2016.

Liu, S., Johns, E., and Davison, A. J. End-to-end multi-task learning with attention. In *CVPR*, pp. 1871–1880. Computer Vision Foundation / IEEE, 2019.

Liu, S., James, S., Davison, A. J., and Johns, E. Auto-lambda: Disentangling dynamic task relationships. *Transactions on Machine Learning Research*, 2022.

Lu, P., Kobyzev, I., Rezagholizadeh, M., Rashid, A., Ghodsi, A., and Langlais, P. Improving generalization of pretrained language models via stochastic weight averaging. In *EMNLP*, pp. 4948–4954, 2022.

Ma, J., Zhao, Z., Yi, X., Chen, J., Hong, L., and Chi, E. H. Modeling task relationships in multi-task learning with multi-gate mixture-of-experts. In *SIGKDD*, pp. 1930–1939. ACM, 2018.

Ma, J., Zhao, Z., Chen, J., Li, A., Hong, L., and Chi, E. H. Snr: Sub-network routing for flexible parameter sharing in multi-task learning. In *AAAI*, volume 33, pp. 216–223, 2019.

Matena, M. S. and Raffel, C. A. Merging models with fisher-weighted averaging. *NeurIPS*, 35:17703–17716, 2022.

Misra, I., Shrivastava, A., Gupta, A., and Hebert, M. Cross-stitch networks for multi-task learning. In *CVPR*, pp. 3994–4003. IEEE Computer Society, 2016.

Navon, A., Shamsian, A., Achituve, I., Maron, H., Kawaguchi, K., Chechik, G., and Fetaya, E. Multi-task learning as a bargaining game. In *ICML*, pp. 16428–16446. PMLR, 2022.

Ortiz-Jimenez, G., Favero, A., and Frossard, P. Task arithmetic in the tangent space: Improved editing of pre-trained models. *NeurIPS*, 2023.

Pan, J., Mao, Y., Ruiz, A. L., Sun, Y., and Flores, A. Predicting different types of conversions with multi-task learning in online advertising. In *SIGKDD*, pp. 2689–2697, 2019.

Peña, F. A. G., Medeiros, H. R., Dubail, T., Aminbeidokhti, M., Granger, E., and Pedersoli, M. Re-basin via implicit sinkhorn differentiation. In *CVPR*, pp. 20237–20246, 2023.

Radford, A., Kim, J. W., Hallacy, C., Ramesh, A., Goh, G., Agarwal, S., Sastry, G., Askell, A., Mishkin, P., Clark, J., et al. Learning transferable visual models from natural language supervision. In *ICML*, pp. 8748–8763. PMLR, 2021.

Ramé, A., Couairon, G., Shukor, M., Dancette, C., Gaya, J.-B., Soulier, L., and Cord, M. Rewarded soups: towards pareto-optimal alignment by interpolating weights fine-tuned on diverse rewards. In *NeurIPS*, 2023.

Sener, O. and Koltun, V. Multi-task learning as multi-objective optimization. In *NeurIPS*, pp. 525–536, 2018.

Shridhar, M., Manuelli, L., and Fox, D. Perceiver-actor: A multi-task transformer for robotic manipulation. In *Conference on Robot Learning*, pp. 785–799. PMLR, 2023.

Song, D., Yang, E., Guo, G., Shen, L., Jiang, L., and Wang, X. Multi-scenario and multi-task aware feature interaction for recommendation system. *ACM Transactions on Knowledge Discovery from Data*, 2024.

Stallkamp, J., Schlipsing, M., Salmen, J., and Igel, C. The german traffic sign recognition benchmark: a multi-class classification competition. In *IJCNN*, pp. 1453–1460. IEEE, 2011.

Stoica, G., Bolya, D., Bjorner, J., Hearn, T., and Hoffman, J. Zipit! merging models from different tasks without training. *arXiv preprint arXiv:2305.03053*, 2023.

Sun, X., Panda, R., Feris, R., and Saenko, K. Adashare: Learning what to share for efficient deep multi-task learning. *NeurIPS*, 33:8728–8740, 2020.

Tang, A., Shen, L., Luo, Y., Ding, L., Hu, H., Du, B., and Tao, D. Concrete subspace learning based interference elimination for multi-task model fusion. *arXiv preprint arXiv:2312.06173*, 2023.

Tang, A., Shen, L., Luo, Y., Zhan, Y., Hu, H., Du, B., Chen, Y., and Tao, D. Parameter efficient multi-task model fusion with partial linearization. *ICLR*, 2024.

Tang, H., Liu, J., Zhao, M., and Gong, X. Progressive layered extraction (ple): A novel multi-task learning (mtl) model for personalized recommendations. In *RecSys*, pp. 269–278, 2020.

Van der Maaten, L. and Hinton, G. Visualizing data using t-sne. *Journal of Machine Learning Research*, 9(11), 2008.

Vandenhende, S., Georgoulis, S., Van Gansbeke, W., Proesmans, M., Dai, D., and Van Gool, L. Multi-task learning for dense prediction tasks: A survey. *TPAMI*, 44(7):3614–3633, 2021.

Wang, Y., Lam, H. T., Wong, Y., Liu, Z., Zhao, X., Wang, Y., Chen, B., Guo, H., and Tang, R. Multi-task deep recommender systems: A survey. *arXiv preprint arXiv:2302.03525*, 2023.

Wang, Z. and Tsvetkov, Y. Gradient vaccine: Investigating and improving multi-task optimization in massively multilingual models. In *ICLR*, 2021.

Wortsman, M., Ilharco, G., Gadre, S. Y., Roelofs, R., Gontijo-Lopes, R., Morcos, A. S., Namkoong, H., Farhadi, A., Carmon, Y., Kornblith, S., et al. Model soups: averaging weights of multiple fine-tuned models improves accuracy without increasing inference time. In *ICML*, pp. 23965–23998. PMLR, 2022.

Wu, C., Wang, T., Ge, Y., Lu, Z., Zhou, R., Shan, Y., and Luo, P. pi-tuning: Transferring multimodal foundation models with optimal multi-task interpolation. In *ICML*, pp. 37713–37727. PMLR, 2023.

Xiao, J., Ehinger, K. A., Hays, J., Torralba, A., and Oliva, A. Sun database: Exploring a large collection of scene categories. *IJCV*, 119:3–22, 2016.

Yadav, P., Tam, D., Choshen, L., Raffel, C., and Bansal, M. Resolving interference when merging models. *NeurIPS*, 2023.

Yang, E., Pan, J., Wang, X., Yu, H., Shen, L., Chen, X., Xiao, L., Jiang, J., and Guo, G. Adatask: A task-aware adaptive learning rate approach to multi-task learning. In *AAAI*, volume 37, pp. 10745–10753, 2023.

Yang, E., Wang, Z., Shen, L., Liu, S., Guo, G., Wang, X., and Tao, D. Adamerging: Adaptive model merging for multi-task learning. *ICLR*, 2024.

Yu, L., Yu, B., Yu, H., Huang, F., and Li, Y. Language models are super mario: Absorbing abilities from homologous models as a free lunch. *arXiv preprint arXiv:2311.03099*, 2023.

Yu, T., Kumar, S., Gupta, A., Levine, S., Hausman, K., and Finn, C. Gradient surgery for multi-task learning. *NeurIPS*, 33:5824–5836, 2020.

Yuval, N. Reading digits in natural images with unsupervised feature learning. In *NIPS Workshop*, 2011.

Zhang, J., Chen, S., Liu, J., and He, J. Composing parameter-efficient modules with arithmetic operations. In *NeurIPS*, 2023.

Zhang, W., Deng, L., Zhang, L., and Wu, D. A survey on negative transfer. *IEEE/CAA Journal of Automatica Sinica*, 10(2):305–329, 2022.

Appendix Overview. The main contents of this appendix are as follows:

- In Appendix A, we describe the datasets, baselines, and training details in detail.
- In Appendix B, we show some experimental results and experimental analyses that are deleted due to the page limit of the main text.

A. Experimental Setting

A.1. Datasets

Following Task Arithmetic (Ilharco et al., 2023), Ties-Merging (Yadav et al., 2023) and AdaMerging (Yang et al., 2024), we merge the models trained on the following eight datasets.

- **SUN397** (Xiao et al., 2016): The database is a benchmark dataset for Scene Understanding (SUN) and contains a total of 108,753 images from 397 classes, where each class contains a different number of images, but each class has at least 100 images.
- **Cars** (Krause et al., 2013): Stanford Cars is a dataset used for fine-grained recognition in the field of computer vision. It contains images of 196 car classes with a total of 16,185 images. The images of each class are divided roughly 1:1 into training set and test set.
- **RESISC45** (Cheng et al., 2017): The RESISC45 dataset is a publicly available benchmark for scene classification in remote sensing images. It contains 45 scene classes and each class contains 700 images (each image resolution is 256×256), for a total of about 31,500 images.
- **EuroSAT** (Helber et al., 2019): EuroSAT is a Sentinel-2-based satellite image dataset primarily used to classify land use in geospatial imagery and contains 27,000 labeled and geo-referenced images in 10 classes.
- **SVHN** (Yuval, 2011): SVHN is a real image dataset containing 10 classes of color house number images, SVHN is very similar in style to MNIST (LeCun, 1998) (handwritten digits in grayscale images), but it contains a larger number of images (more than 600,000 digital images).
- **GTSRB** (Stallkamp et al., 2011): The German Traffic Sign Recognition Benchmark (GTSRB) contains images with different lighting conditions and rich backgrounds. These images are classified into 43 classes of traffic signs, totaling more than 50,000 images.
- **MNIST** (LeCun, 1998): MNIST is a large database of handwritten digits in 10 classes that is one of the most famous datasets in machine learning. It contains 60,000 training images and 10,000 test images, each of which is 28x28 pixels.
- **DTD** (Cimpoi et al., 2014): The Describable Textures Dataset (DTD) contains 5,640 real labeled texture images, divided into 47 classes, each with about 120 images (all between 300×300 and 640×640 pixels).

A.2. Baselines

The comparison methods in all our experiments are divided into three categories in total: non-model merging methods, model merging methods, and our methods. Specific information is as follows:

(i) *Non-model merging methods*:

- **Pretrained** directly uses the pre-trained model to predict multiple tasks. Since it does not utilize any downstream task-related information for model training, its performance on these downstream tasks is usually very poor.
- **Individual** makes predictions using models that are fine-tuned independently for each task. It is usually the best performance because it has no interference between tasks. However, it requires maintaining a copy of the model parameters for each downstream task, which can be prohibitive in terms of memory cost.
- **Traditional MTL** mixes training data from multiple tasks to fine-tune a pre-trained model (i.e., a hard-parameter sharing network). Due to potential interference between tasks, it often suffers from negative transfer (Zhang et al., 2022) problems, that is, the predictive performance on a single task is not as good as that of individual models. However, it is very attractive in terms of parameters and computational efficiency.

(ii) *Model merging methods*:

- **Weight Averaging** is the most direct and simple model merging method. It directly averages the model parameters trained on multiple tasks into one model to perform multi-task learning.
- **Fisher Merging** (Matena & Raffel, 2022) measures the importance of each parameter through the Fisher information matrix (Fisher, 1922), thus merging model parameters based on this importance.

- **RegMean** (Jin et al., 2023) adjusts the weights and forms linear combinations of rows in weight matrices using statistical information derived from training data.
- **Task Arithmetic** (Ilharco et al., 2023) defines the concept of "task vector", which takes the fine-tuned model parameters minus the pre-trained model parameters as a task vector, and then combines multiple task vectors and adds them to the pre-trained model to perform multi-task learning.
- **Ties-Merging** (Yadav et al., 2023) adds three steps, TRIM, ELECT SIGN and MERGE, based on Task Arithmetic (Ilharco et al., 2023). These steps delete unimportant parameters in the task vector and correct the sign conflict problem of parameters, thereby easing the interference when the final task vector is merged.
- **Concrete TA** (Tang et al., 2023) merges models in a parameter subspace shared between tasks, where the subspace is a learnable mask matrix, and Task Arithmetic (Ilharco et al., 2023) is used when merging models in the subspace.
- **Concrete AM** (Tang et al., 2023) also merges models in subspace, and uses AdaMerging (Yang et al., 2024) to learn model merging coefficients during merging.
- **TW AdaMerging** (Yang et al., 2024) uses an unsupervised test set to adaptively learn the merging coefficient of each task vector in Task Arithmetic (Ilharco et al., 2023).
- **AdaMerging** (Yang et al., 2024) uses an unlabeled test set to adaptively learn the merging coefficients of each layer in each task vector in Task Arithmetic (Ilharco et al., 2023).

(iii) *Our methods*:

Note that our representation surgery scheme is orthogonal to existing model merging schemes and can be seamlessly integrated into arbitrary model merging schemes. In this paper, we choose four representative model merging methods for experiments.

- **Weight Averaging w/ Surgery(Ours)**: The representation surgery scheme proposed in this paper is performed on the model merged using the Weight Averaging scheme.
- **Task Arithmetic w/ Surgery(Ours)**: Based on Task Arithmetic (Ilharco et al., 2023), the representation surgery scheme proposed in this paper is adopted.
- **Ties-Merging w/ Surgery(Ours)**: The representation surgery scheme, as suggested in this paper, is executed on the model created through the Ties-Merging (Yadav et al., 2023) scheme.
- **AdaMerging w/ Surgery(Ours)**: The proposed representation surgery scheme is applied on AdaMerging (Yang et al., 2024), an advanced model merging method.

A.3. Discussion of Related Work

By reading related work recommended by the anonymous reviewer, we find that this work has some technical connections with KD4MTL (Li & Bilen, 2020), but also has the following essential differences:

- **Different learning paradigms**: KD4MTL belongs to the traditional MTL paradigm, which trains an MTL from scratch (i.e., Learn From Data) on raw data from multiple tasks. However, our work directly combines multiple independently trained models to complete MTL, and provides a new scheme for MTL (i.e., Learn From Model), which is orthogonal to traditional MTL. In addition, the paradigm of model merging based MTL effectively reduces the data management cost and data privacy issues of training data.
- **Different goals**: KD4MTL uses the trained model as an additional regularization term to alleviate the imbalance loss optimization problem in traditional MTL, which is caused by differences in task difficulty and loss magnitude. On the other hand, our work focuses on mitigating the representation bias problem caused by model merging, which is caused by parameter interference.
- **Different learning resources**: Our approach only requires some individually trained models to be merged, which is a more practical scenario. In contrast, KD4MTL requires both individually trained models and raw data to be trained from scratch, which is more expensive.

It should be noted that the traditional MTL paradigm, including KD4MTL (Li & Bilen, 2020) and Naive MTL, is an *upper bound* of the model merging based MTL (as shown in Tab. 6). In this paper, we focus on comparing various model merging based MTL methods.

A.4. Implementation Details

Our proposed representation surgery module contains a small number of trainable parameters (i.e., $\mathbf{W}_{down}$ and $\mathbf{W}_{up}$ in Eq. 3). In this paper, we do not require arbitrarily labeled training data. Instead, we use unlabeled test data and individual models to construct self-supervised training signals to update these parameters. Specifically, we use the Adam optimizer (Kingma & Ba, 2014) to update these parameters with a learning rate of $1e-3$ and a momentum of (0.9, 0.999). We update for 1,000 iterations with a batch size of 16. In addition, we set the rank (i.e., r in Eq. 3) of the surgery module to 16 by default, and we also tried values such as {4, 8, 16, 32, 64} in the experimental analysis. Finally, we report the accuracy on each task, as well as the average accuracy (i.e., **Avg.**) on the eight tasks.

B. Experimental Results and Analysis

B.1. Performance in Computer Vision Domain

Performance on ViT-B/16. Tab. 3 shows the results of various model merging methods in the ViT-B/16 architecture. We can observe that when using the proposed representation surgery module on Weight Averaging, Task Arithmetic, Ties-Merging and AdaMerging, the performance of all methods has been significantly improved. For example, on Weight Averaging, the performance without representation surgery is 70.7%, but with the use of surgery, the performance is improved to 82.6%. Similarly, on AdaMerging, with the use of surgery, the performance is improved from 84.9% to 88.8%.

Table 3. Multi-task performance when merging ViT-B/16 models on eight tasks.

Method	SUN397	Cars	RESISC45	EuroSAT	SVHN	GTSRB	MNIST	DTD	Avg.
Pretrained	63.8	64.6	65.7	54.5	52.0	43.3	51.7	45.1	55.0
Individual	81.8	86.8	96.9	99.7	97.8	99.1	99.7	82.0	92.9
Weight Averaging	67.7	70.0	75.3	79.5	74.9	60.1	94.4	43.8	70.7
Fisher-Merging (Matena & Raffel, 2022)	68.5	69.9	75.2	80.4	73.2	61.2	94.5	50.7	71.7
RegMean (Jin et al., 2023)	69.1	71.6	77.6	88.8	83.7	70.2	96.9	54.6	76.6
Task Arithmetic (Ilharco et al., 2023)	61.1	65.9	74.0	76.2	88.0	73.9	98.4	53.0	73.8
Ties-Merging (Yadav et al., 2023)	69.1	72.5	80.5	84.0	85.0	71.5	98.1	54.9	77.0
AdaMerging (Yang et al., 2024))	70.2	80.7	81.6	94.8	91.6	95.8	98.5	66.2	84.9
Weight Averaging w/ Surgery (Ours)	70.3	72.4	88.8	97.6	82.0	83.1	98.1	68.5	82.6
Task Arithmetic w/ Surgery (Ours)	68.3	72.3	88.7	97.7	91.0	89.5	98.9	72.9	84.9
Ties-Merging w/ Surgery (Ours)	73.0	76.2	90.7	98.1	89.7	86.7	98.7	75.2	86.0
AdaMerging w/ Surgery (Ours)	73.6	81.5	90.4	98.5	93.2	97.4	98.9	77.0	88.8

Available Data Ratio. Our scheme relies on unlabeled test data to build self-supervision signals and thus update the "representation surgery" module. As shown in Tab. 4 and Fig. 8(a), we tested the performance of model merging using representation surgery when different ratios (e.g., 1%, 5%, 10% or 100%) of unlabeled test data are visible. We observe that the representation surgery is consistently effective for different amounts of data. For example, with only 1% of visible data, the accuracy is 82.8% using representation surgery, which is significantly higher than 80.1% without surgery. In addition, as the visible data increases, the gain obtained by representing the surgery clearly increases. For example, when 10% of the data is visible, the average accuracy improves to 84.7%.

Table 4. Impact of the amount of available test data in "representation surgery" module on model performance.

Method	Available Test Set	SUN397	Cars	RESISC45	EuroSAT	SVHN	GTSRB	MNIST	DTD	Avg.
AdaMerging (Yang et al., 2024)	-	64.5	68.1	79.2	93.8	87.0	91.9	97.5	59.1	80.1
AdaMerging w/ Surgery (Ours)	1%	68.6	64.1	85.3	96.3	90.9	95.9	98.4	63.0	82.8
AdaMerging w/ Surgery (Ours)	5%	69.4	67.4	87.9	97.3	91.5	95.7	98.5	62.6	83.8
AdaMerging w/ Surgery (Ours)	10%	69.7	68.9	88.0	97.5	91.5	95.6	98.5	67.8	84.7
AdaMerging w/ Surgery (Ours)	100%	69.8	71.0	88.9	98.1	91.7	96.5	98.8	73.6	86.1

Online Data Ratio. In a more realistic scenario, the test data may arrive *online*, and receive only one sample at a time, where each sample is used to train the model only once rather than multiple times. As shown in Tab. 5 and Fig. 8(b), we tested the performance of the proposed "representation surgery" scheme in this scenario. We observe that when only a small amount of data is used to train the model once, there is also a performance gain compared to not using the representation surgery scheme. For example, with 10% of the data, the average accuracy of the merged model with "representation surgery"

is 83.3%, while the average accuracy without "representation surgery" is 80.1%. In addition, as the amount of online data increases, the average performance of using the "representation surgery" scheme improves significantly. For example, increasing from 10% to 50% improves the average performance from 83.3% to 84.8%.

Table 5. Impact of the amount of **online test data** on model performance in "representation surgery" module.

| Method	Available Test Set	SUN397	Cars	RESISC45	EuroSAT	SVHN	GTSRB	MNIST	DTD	**Avg.**
AdaMerging (Yang et al., 2024)	-	64.5	68.1	79.2	93.8	87.0	91.9	97.5	59.1	80.1
AdaMerging w/ Surgery (Ours)	1%	67.7	68.0	81.8	91.2	88.4	94.5	97.7	60.5	81.2
AdaMerging w/ Surgery (Ours)	10%	69.2	68.7	85.2	95.3	90.0	95.8	98.2	64.2	83.3
AdaMerging w/ Surgery (Ours)	20%	69.5	69.1	86.7	95.9	91.1	95.7	98.2	67.5	84.2
AdaMerging w/ Surgery (Ours)	30%	69.8	69.2	87.0	96.3	91.1	95.7	98.4	67.5	84.3
AdaMerging w/ Surgery (Ours)	40%	69.6	68.7	87.5	97.1	91.4	96.1	98.5	67.9	84.6
AdaMerging w/ Surgery (Ours)	50%	69.5	69.9	87.9	97.0	91.3	96.1	98.5	68.7	84.8
AdaMerging w/ Surgery (Ours)	60%	69.6	68.6	87.9	97.4	91.6	96.1	98.5	69.6	84.9
AdaMerging w/ Surgery (Ours)	70%	69.5	69.5	87.9	97.1	91.6	96.2	98.6	71.0	85.1
AdaMerging w/ Surgery (Ours)	80%	69.7	69.6	88.4	97.4	91.4	96.5	98.5	71.0	85.3
AdaMerging w/ Surgery (Ours)	90%	69.5	69.7	88.1	97.6	92.0	96.4	98.5	70.6	85.3
AdaMerging w/ Surgery (Ours)	100%	69.4	69.0	88.2	97.6	92.0	96.3	98.7	70.7	85.2
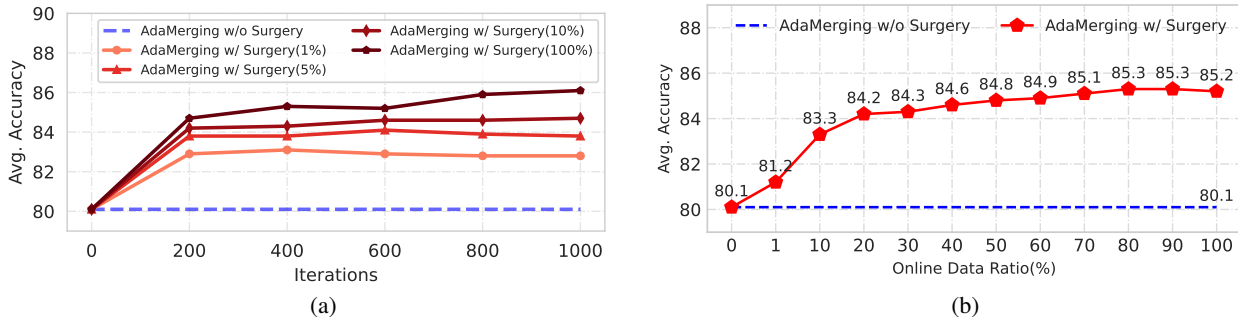
Figure 8. (a) Performance variation of the "representation surgery" scheme for different amounts of test data in the *offline* scenario. (b) Performance change of the "representation surgery" scheme for different test data volumes in the *online* scenario.

B.2. Performance in Natural Language Processing (NLP) Domain

In this section, we demonstrate that the phenomenon of "representation bias" also exists in the merged model in the NLP domain, and the performance of the merged model after applying the proposed "representation surgery" is significantly improved. Specifically, as shown in Tab. 6 and Tab. 7, we have the following observations: (i) Traditional MTL collects data from multiple tasks in advance and jointly trains an MTL model, which has better performance. In addition, more advanced MTL (e.g., KD4MTL (Li & Bilen, 2020)) can further improve performance through some additional designs to alleviate the negative transfer problem in multi-task joint training. These learning-from-data MTL methods can be viewed as *upper bounds* on model merging-based MTL methods. (ii) Weight Averaging and Task Arithmetic suffer from the "representation bias" problem (in Tab. 7), and the performance of the merged model has a significant gap compared with Traditional MTL or KD4MTL (in Tab. 6). (iii) When the "representation surgery" scheme proposed in this paper is used in the merged model, the representation bias is significantly alleviated (in Tab. 7), and the performance of the merged model is also significantly improved (in Tab. 6), very close to traditional MTL.

Table 6. Multi-task performance (higher better) when merging BERT models on five NLP tasks.

Method	Learning Source	AG News	Yelp Sentiment	Amazon Sentiment	Yahoo Q&A	DBPedia Wikipedia	**Avg.**
Traditional MTL	Original Training Data	90.6	59.1	55.6	71.3	98.5	75.0
KD4MTL (Li & Bilen, 2020)	Original Training Data & Trained Models	91.6	59.2	57.0	71.2	98.7	75.5
Weight Averaging	Trained Models	79.2	49.8	45.0	50.3	55.1	55.8
Weight Averaging w/ Surgery (Ours)	Trained Models	90.3	58.0	54.2	70.8	98.4	74.3
Task Arithmetic (Ilharco et al., 2023)	Trained Models	82.9	55.8	48.4	53.1	81.5	64.3
Task Arithmetic w/ Surgery (Ours)	Trained Models	89.8	58.4	55.4	70.3	98.0	74.4

B.3. Analysis

Different Ranks. Tab. 8 shows the performance changes when the surgery module uses different ranks under the ViT-B/32 architecture. We have consistently observed that in both model merging methods, Task Arithmetic and AdaMerging, as

Table 7. Representation bias (lower better) when merging BERT models on five NLP tasks.

Method	AG News	Yelp Sentiment	Amazon Sentiment	Yahoo Q&A	DBPedia Wikipedia	Avg.
Weight Averaging	0.448	0.336	0.349	0.418	0.539	0.418
Weight Averaging w/ Surgery (Ours)	0.208	0.171	0.189	0.211	0.188	0.193
Task Arithmetic (Ilharco et al., 2023)	0.373	0.305	0.362	0.378	0.395	0.362
Task Arithmetic w/ Surgery (Ours)	0.190	0.179	0.194	0.226	0.172	0.192

the rank size increases, the performance of the merged model improves. For example, when the rank r increases from 4 to 64, Task Arithmetic's performance improves from 74.3% to 84.3%, while AdaMerging's performance improves from 83.5% to 87.5%. This is due to the fact that as mentioned in Sec. 4.2, the surgery module can be regarded as being used to accommodate task-private information. When the dimension of the rank is larger, more task-private information can be stored, so the performance is better.

Table 8. Multi-task performance on the ViT-B/32 model when different ranks in the representation surgery module.

Method	SUN397	Cars	RESISC45	EuroSAT	SVHN	GTSRB	MNIST	DTD	Avg.
Task Arithmetic (Ilharco et al., 2023)	55.2	54.9	66.7	78.9	80.2	69.7	97.3	50.4	69.1
Task Arithmetic w/ Surgery (r=4)	62.6	55.9	76.7	78.7	83.4	79.6	97.7	60.0	74.3
Task Arithmetic w/ Surgery (r=8)	63.2	58.5	79.9	93.9	84.5	82.1	98.5	64.2	78.1
Task Arithmetic w/ Surgery (r=16)	63.8	59.9	83.3	97.9	87.0	87.0	98.6	69.4	80.9
Task Arithmetic w/ Surgery (r=32)	64.7	62.4	87.3	98.3	88.3	92.8	98.8	72.5	83.1
Task Arithmetic w/ Surgery (r=64)	65.6	62.7	89.8	98.3	88.7	95.6	98.9	74.7	84.3
AdaMerging (Yang et al., 2024)	64.5	68.1	79.2	93.8	87.0	91.9	97.5	59.1	80.1
AdaMerging w/ Surgery (r=4)	68.7	68.9	85.5	95.6	88.3	95.6	97.9	67.4	83.5
AdaMerging w/ Surgery (r=8)	69.0	68.1	87.0	97.4	89.3	95.6	98.4	71.2	84.5
AdaMerging w/ Surgery (r=16)	69.8	71.0	88.9	98.1	91.7	96.5	98.8	73.6	86.1
AdaMerging w/ Surgery (r=32)	70.5	70.9	90.4	98.6	92.1	97.5	98.8	75.4	86.8
AdaMerging w/ Surgery (r=64)	71.2	72.0	92.3	99.0	92.2	97.9	99.0	76.1	87.5

Different Loss Functions. We aim to minimize the gap between "representations after performing surgery" and "representations from individual models". Therefore, any distance function can be used, not limited to L_1 loss in Eq. 3. As shown in the following Tab. 9, we further test the effectiveness of our method under different loss functions: MSELoss, SmoothL1Loss, and Negative CosineSimilarity. The proposed surgery method is effective under all these loss functions, further demonstrating the robustness of our approach.

Table 9. Multi-task performance on the ViT-B/32 model when different loss functions are used in the representation surgery module.

Method	SUN397	Cars	RESISC45	EuroSAT	SVHN	GTSRB	MNIST	DTD	Avg.
Task Arithmetic (Ilharco et al., 2023)	55.2	54.9	66.7	78.9	80.2	69.7	97.3	50.4	69.1
Task Arithmetic w/ Surgery (L1 Loss)	63.8	59.9	83.3	97.9	87.0	87.0	98.6	69.4	80.9
Task Arithmetic w/ Surgery (MSELoss)	64.3	59.8	84.0	97.8	87.6	88.7	98.8	69.8	81.4
Task Arithmetic w/ Surgery (SmoothL1Loss)	64.1	59.7	84.1	97.9	88.1	89.7	98.8	70.6	81.6
Task Arithmetic w/ Surgery (Negative CosineSimilarity)	63.8	59.2	84.5	97.5	88.8	86.4	99.0	70.3	81.2

Parameters Costs. As shown in Tab. 10, we counted the number of additional parameters introduced by the proposed "representation surgery" module. We observe that the number of parameters in the surgery module is insignificant compared to the number of parameters that need to be merged, which is usually around one in ten thousand. For example, on ViT-B/32, the number of parameters to be merged is 907,589,640, while the number of parameters for the surgery module is only 131,072, and the latter only accounts for 0.014% of the former.

Training Step v.s. Avg. Accuracy. As shown in Fig. 9, we show how the average accuracy of the four model merging methods (i.e., Weight Averaging, Task Arithmetic, Ties-Merging and AdaMerging) using the proposed "representation surgery" changes with the number of iterations on ViT-B/32 and ViT-B/16 architectures. We consistently observe that with only a small number of iterations (e.g., 200), the accuracy with representation surgery has a very significant improvement over the initial point (i.e., without surgery). This also indicates that the proposed "representation surgery" scheme has the dual advantages of efficiency and effectiveness.

Table 10. The parameter cost of the representation surgery module ($r = 16$).

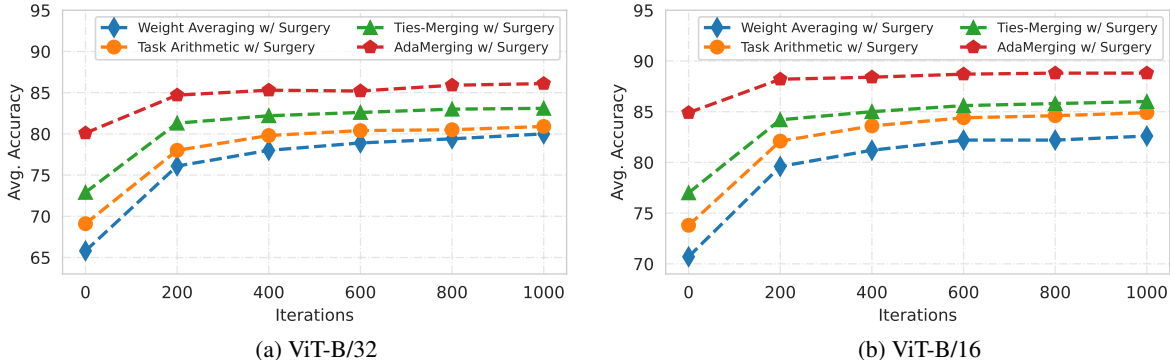| Architectures	ViT-B/32	ViT-B/16	ViT-L/14
The total number of merged model parameters	907,589,640	894,337,032	2,740,496,392
The total number of parameters in the representation surgery module	131,072	131,072	196,608
Ratio	0.000144	0.000146	0.000071

(a) ViT-B/32 (b) ViT-B/16

Figure 9. The average accuracy of the merged model changes with the number of iterations on ViT-B/32 and ViT-B/16.

B.4. Visualization

Representation Bias or L1 Distance. Fig. 10, Fig. 11, and Fig. 12 demonstrate the L_1 distances (i.e., Eq. 1) of the feature representations extracted with and without surgery compared to the feature representations extracted by individual models on three architectures, ViT-B/32, ViT-B/16 and ViT-L/14, respectively. We have consistently observed that after using the representation surgery (red column), the L_1 distance is significantly reduced, which means that the representation surgery effectively alleviates the representation bias problem.

Representation Distribution. Tab. 11 shows that on three architectures of ViT-B/32, ViT-B/16 and ViT-L/14, representation distribution of the features extracted by the individual model versus the features extracted by the merged model with and without the proposed representation surgery for four model merging methods: Weight Averaging, Task Arithmetic (Ilharco et al., 2023), Ties-Merging (Yadav et al., 2023), and AdaMerging (Yang et al., 2024).

Table 11. Visualization of the distribution of the four model merging methods performed and without performed representation surgery under the three architectures.

	Weight Averaging	Task Arithmetic	Ties-Merging	AdaMerging
ViT-B/32	w/o Surgery: Fig. 13 v.s. w/ Surgery: Fig. 14	w/o Surgery: Fig. 15 v.s. w/ Surgery: Fig. 16	w/o Surgery: Fig. 17 v.s. w/ Surgery: Fig. 18	w/o Surgery: Fig. 19 v.s. w/ Surgery: Fig. 20
ViT-B/16	-	-	-	w/o Surgery: Fig. 21 v.s. w/ Surgery: Fig. 22
ViT-L/14	-	-	-	w/o Surgery: Fig. 23 v.s. w/ Surgery: Fig. 24
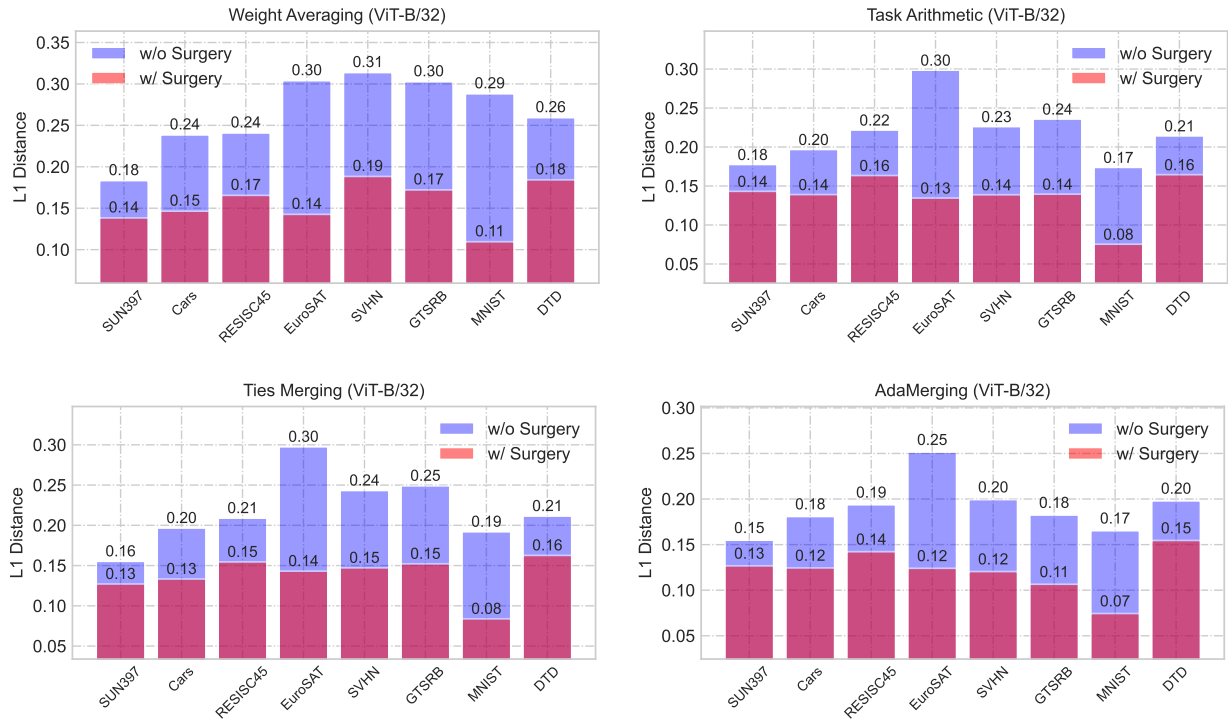
Figure 10. Visualization of the L_1 distance (or "representation bias" in Eq. 1) of the representation of the merged model with and without representation surgery versus the individual model. All results are performed on **ViT-B/32** architecture.
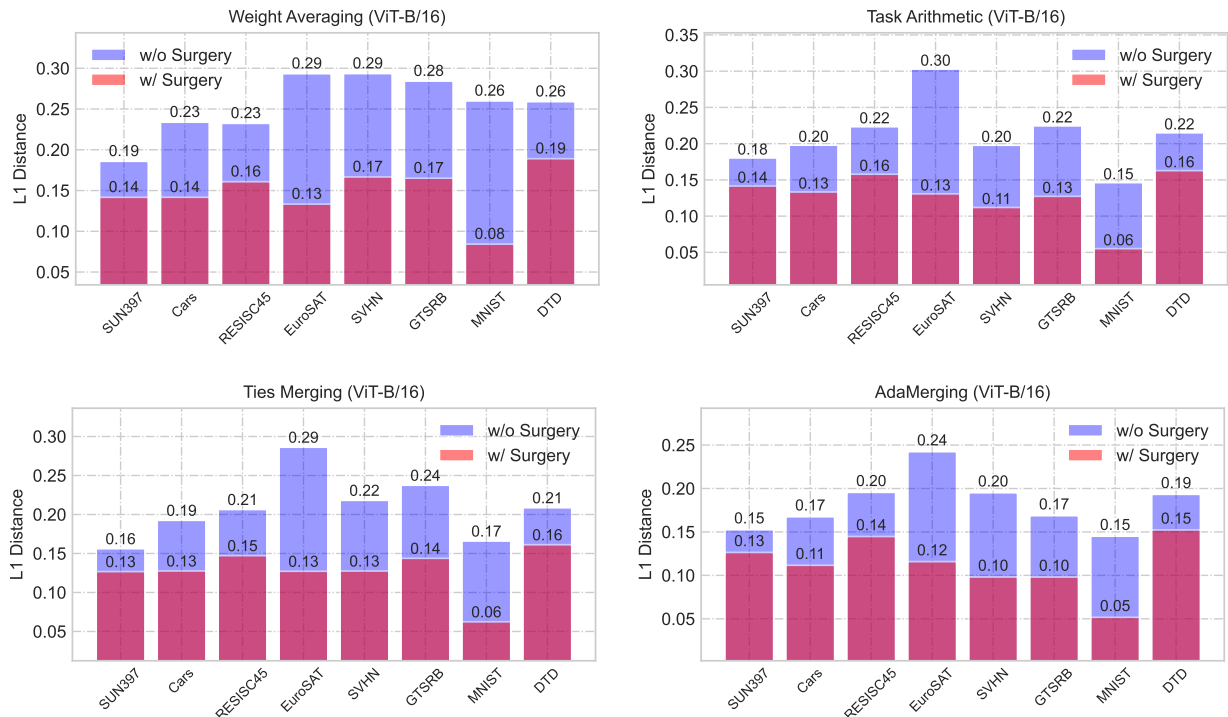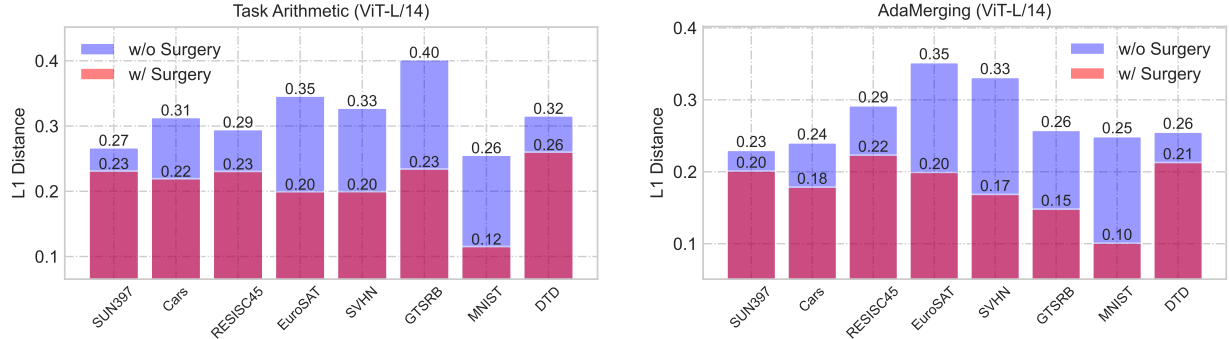

Figure 11. Visualization of the L_1 distance (or "representation bias" in Eq. 1) of the representation of the merged model with and without representation surgery versus the individual model. All results are performed on **ViT-B/16** architecture.

Figure 12. Visualization of the L_1 distance (or "representation bias" in Eq. 1) of the representation of the merged model with and without representation surgery versus the individual model. All results are performed on **ViT-L/14** architecture.
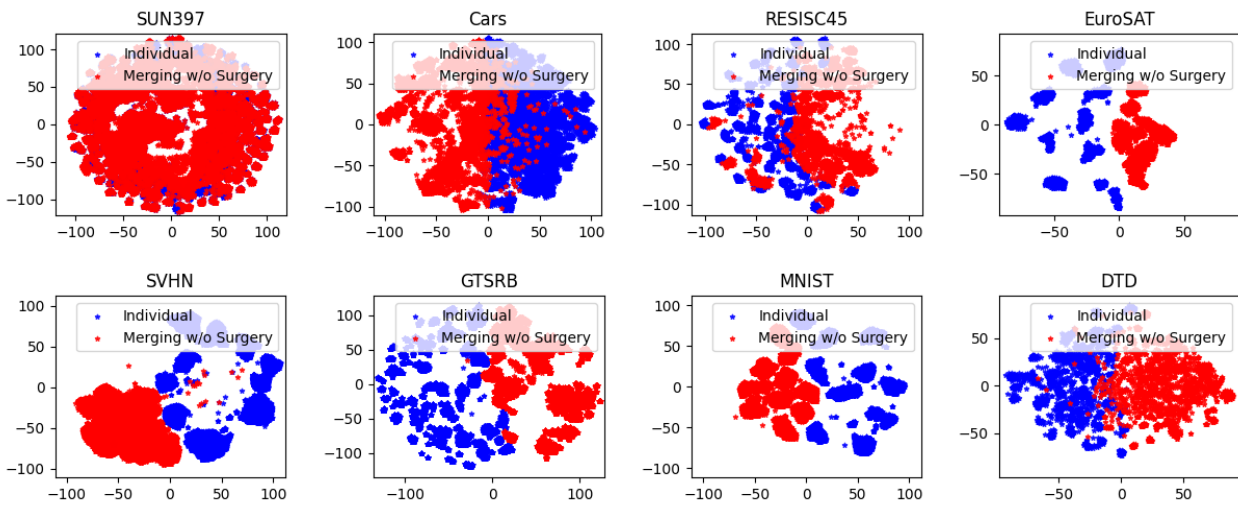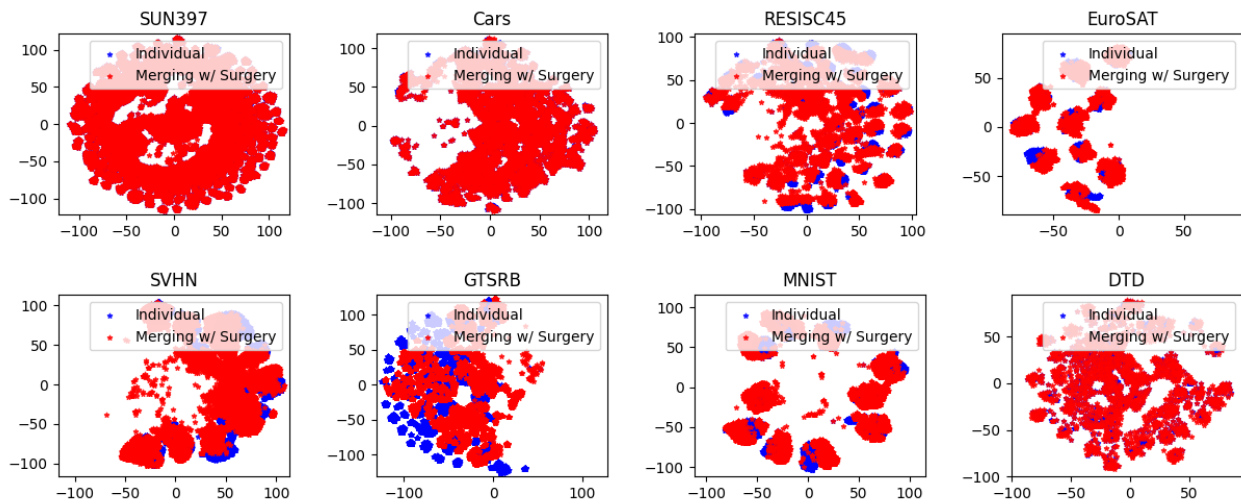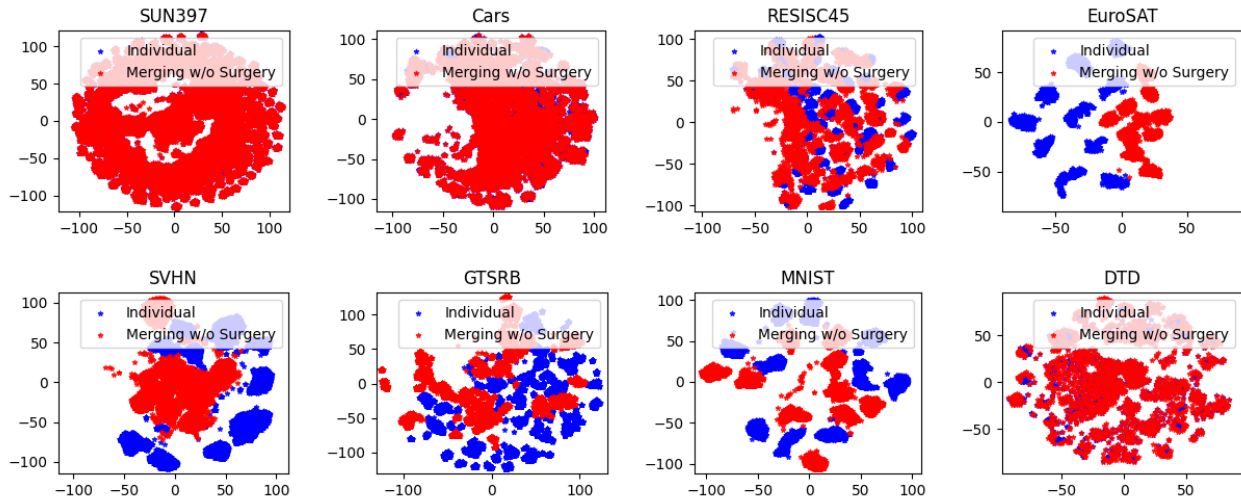

Figure 13. Visualization of representation distribution of **Weight Averaging (w/o Surgery)** on **ViT-B/32** architecture.
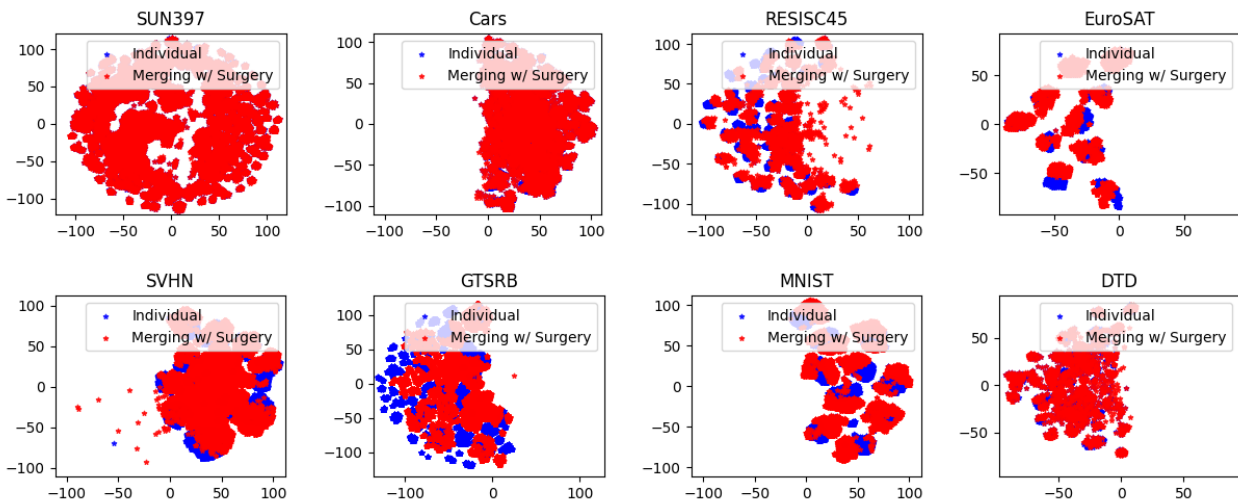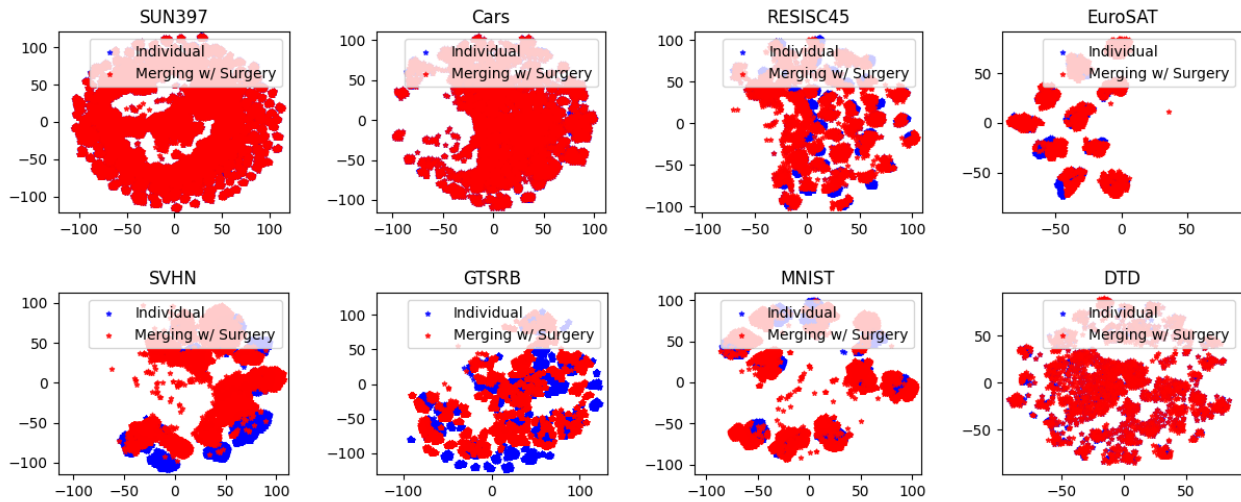

Figure 14. Visualization of representation distribution of **Weight Averaging (w/ Surgery)** on **ViT-B/32** architecture.
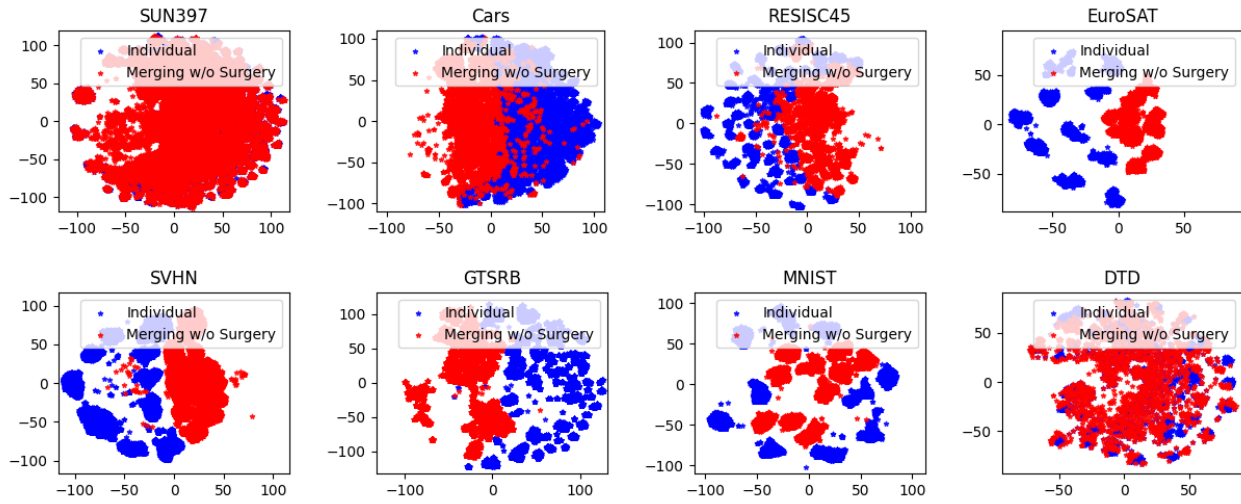
Figure 15. Visualization of representation distribution of **Task Arithmetic (w/o Surgery)** on **ViT-B/32** architecture.
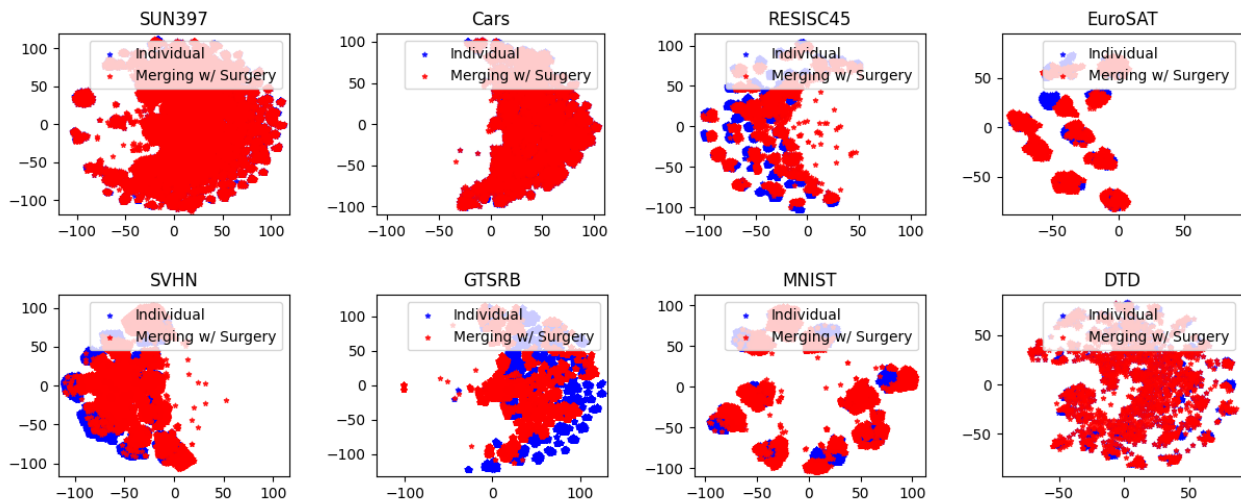
Figure 16. Visualization of representation distribution of **Task Arithmetic (w/ Surgery)** on **ViT-B/32** architecture.
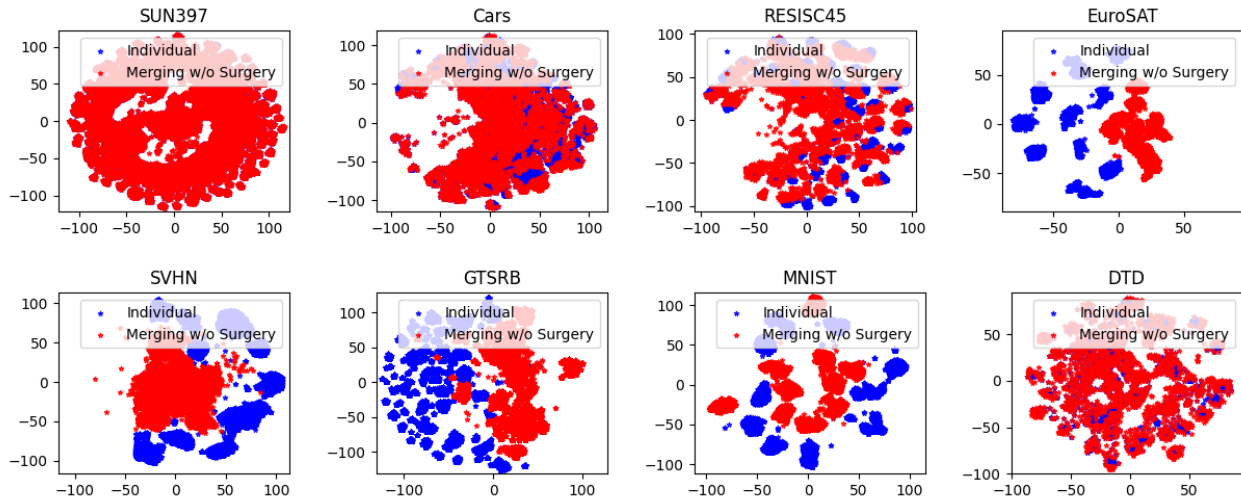
Figure 17. Visualization of representation distribution of **Ties-Merging (w/o Surgery)** on **ViT-B/32** architecture.

Figure 18. Visualization of representation distribution of **Ties-Merging (w/ Surgery)** on **ViT-B/32** architecture.
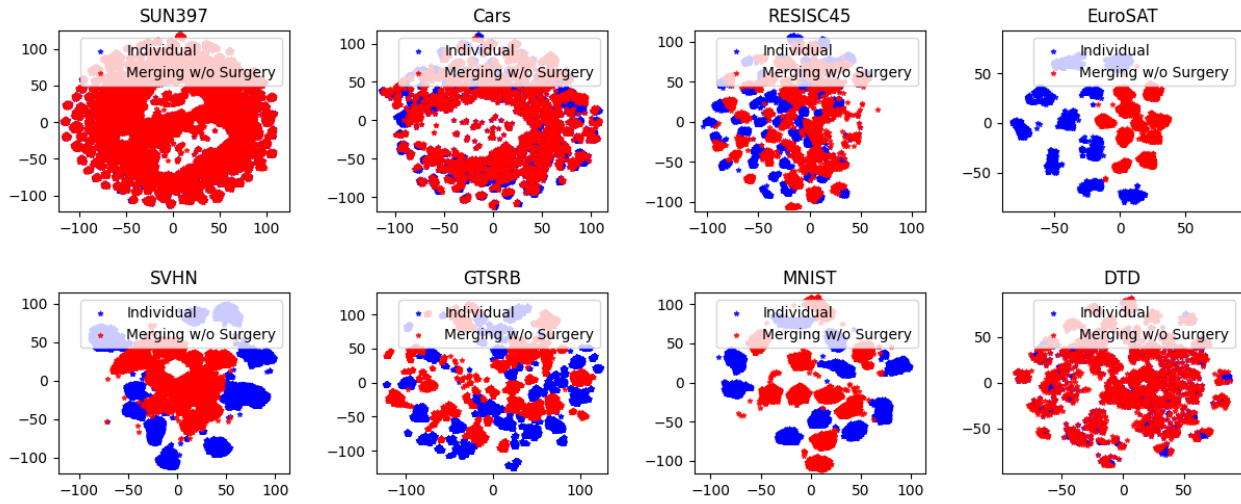
Figure 19. Visualization of representation distribution of **AdaMerging (w/o Surgery)** on **ViT-B/32** architecture.

Figure 20. Visualization of representation distribution of **AdaMerging (w/ Surgery)** on **ViT-B/32** architecture.
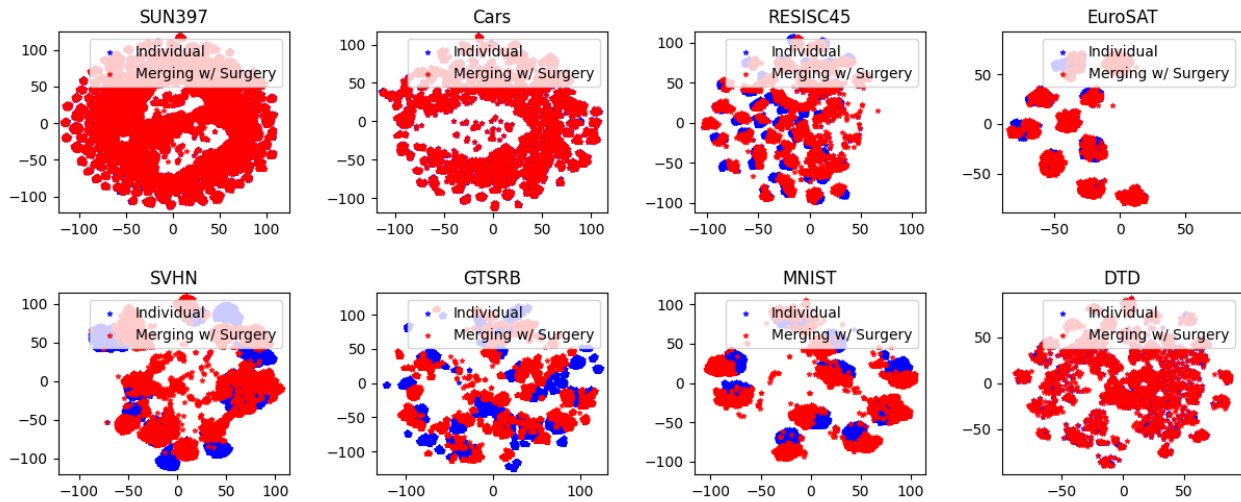
Figure 21. Visualization of representation distribution of **AdaMerging (w/o Surgery)** on **ViT-B/16** architecture.

Figure 22. Visualization of representation distribution of **AdaMerging (w/ Surgery)** on **ViT-B/16** architecture.
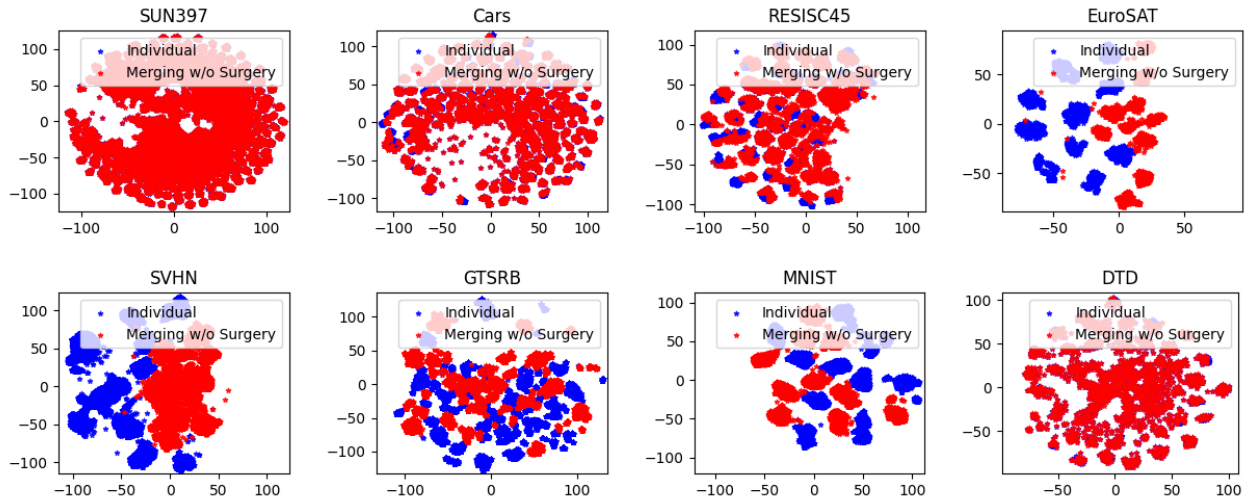
Figure 23. Visualization of representation distribution of **AdaMerging (w/o Surgery)** on **ViT-L/14** architecture.
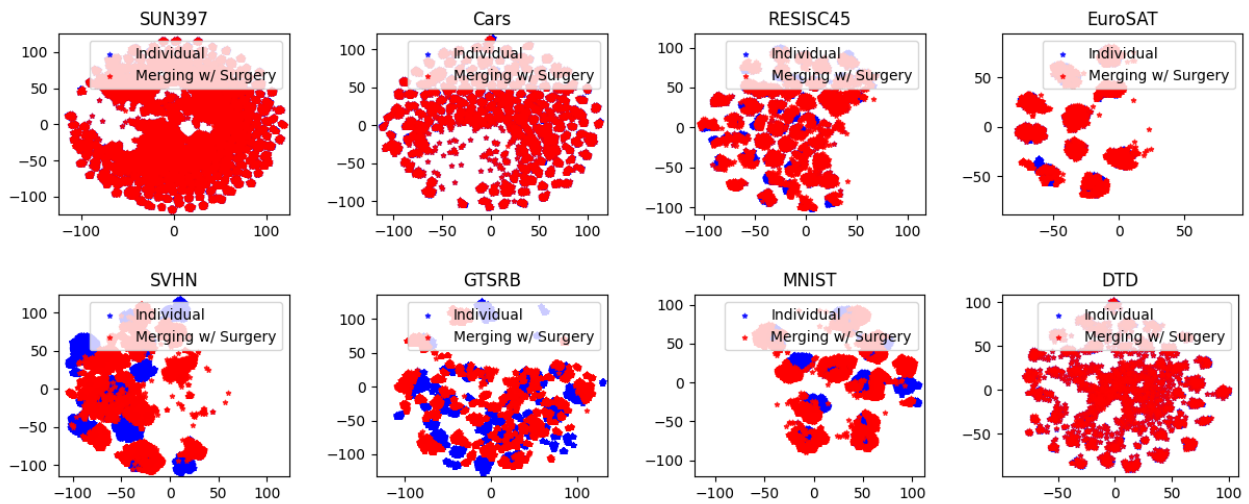

Figure 24. Visualization of representation distribution of **AdaMerging (w/ Surgery)** on **ViT-L/14** architecture.