Merging Smarter, Generalizing Better: Enhancing Model Merging on OOD Data

Anonymous authors
Paper under double-blind review

Abstract

Multi-task learning (MTL) concurrently trains a model on diverse task datasets to exploit common features, thereby improving overall performance across the tasks. Recent studies have dedicated efforts to merging multiple independent model parameters into a unified model for MTL, thus circumventing the need for training data and expanding the scope of applicable scenarios of MTL. However, current approaches to model merging predominantly concentrate on enhancing performance within in-domain (ID) datasets, often overlooking their efficacy on out-of-domain (OOD) datasets. In this work, we propose LwPTV (Layer-wise Pruning Task Vector) by building a salience score, measuring the redundancy of parameters in task vectors. Designed in this way ours can achieve mask vector for each task and thus perform layer-wise pruning on the task vectors, only keeping the pre-trained model parameters at the corresponding layer in merged model. Owing to its flexibility, our method can be seamlessly integrated with most of existing model merging methods to improve their performance on OOD tasks. Extensive experiments demonstrate that the application of our method results in substantial enhancements in OOD performance while preserving the ability on ID tasks.

1 Introduction

Pre-trained models (PTMs) constitute a cornerstone of deep learning, supporting numerous contemporary methodologies by virtue of their capacity to extract generalized features from vast data repositories (Bommasani et al., 2021; Zhuang et al., 2020). Typically, the fine-tuning of pre-trained models with task-specific data is employed to enhance performance (Shnarch et al., 2022; Kenton & Toutanova, 2019). This process yields numerous checkpoints originating from the PTMs (Poth et al., 2021b). Fine-tuning individual models for each task incurs substantial storage and deployment costs (Dettmers et al., 2023). Multi-task learning (MTL) offers an alternative by training a single model on multiple tasks, reducing storage demands. However, MTL introduces significant computational overhead and is constrained by privacy concerns due to the necessity of aggregating diverse datasets (Jin et al., 2023). Recently, the trend in research has shifted to an alternative paradigm: merging various individual fine-tuned models into one single multi-task model, bypassing the requirement for the original training data. This approach, known as model merging, addresses these limitations by merging model parameters rather than engaging in further training (Wortsman et al., 2022a; Matena & Raffel, 2022; Jin et al., 2023).

The most typical technique is Weighted Averaging, which directly averages the weights of multiple fine-tuned models (Wortsman et al., 2022a). However, averaging often leads to a significant degradation in performance. To this end, various techniques, including Fisher Merging (Matena & Raffel, 2022), RegMean(Jin et al., 2023), and Task Arithmetic (Ilharco et al., 2023), have been proposed. Although these techniques improve performance, they primarily focus on improving accuracy within ID datasets - the datasets used to fine-tune task-specific models. However, real-world applications often encounter data distributions that differ from those seen during training. The ability of a model to perform well in out-of-domain (OOD) data directly impacts its reliability, robustness, and deployment in dynamic environments such as healthcare, autonomous systems. Despite its importance, OOD robustness remains an underexplored aspect in model merging. As shown in Fig.1, although having better performance on in-domain (ID) data, merged models with current

Table 1: Summarization of model merging methods: "No Training/ Test" means no need Training/Test set in designing merging coefficient or merging models, "Pruning" indicates discarding parameters in task vectors, "Orthogonal" in merging coefficients indicates that method is orthogonal to merging coefficient designed by others, "Designed for OOD" indicates that the method is specifically designed for merging model on OOD, "Available on OOD" indicates that the merging model is available to OOD data.

Method	No Training set	No Test	Pruning	Merging coefficients	Available on OOD	Designed for OOD	Storage cost
Weight Averaging (Wortsman et al., 2022a)	✓	✓	✗	Uniform	✓	✗	high
Fisher Merging (Matena & Raffel, 2022)	✗	✓	✗	Fisher Matrix	✓	✗	high
RegMean (Jin et al., 2023)	✗	✓	✗	Inner Product Matrix	✓	✗	high
Task Arithmetic (Ilharco et al., 2023)	✓	✓	✗	Uniform	✓	✗	high
Ties-Merging (Yadav et al., 2023)	✓	✓	Magnitude Metric	Uniform	✓	✗	low
AdaMerging (Yang et al., 2024b)	✓	✗	✗	Optimized	✓	✗	high
Surgery (Yang et al., 2024a)	✓	✗	✗	Orthogonal	✗	✗	high
PCB-MERGING (Du et al., 2024)	✓	✓	PCB matrix	$(\hat{\boldsymbol{\beta}}_k \odot \lambda_k)/\sum_{k=1}^{K} \hat{\boldsymbol{\beta}}_i$	✓	✗	low
FR-Merging (Zheng & Wang, 2025)	✓	✓	✗	$\lambda_k \mathbb{E}(\boldsymbol{\tau}_k) \left(\sum_{k=1}^{K} \mathbb{E}(\boldsymbol{\tau}_k)\right)^{-1}$	✓	✗	high
LwPTV (Ours)	✓	✓	Salience Score	Orthogonal	✓	✓	low

merging methods usually exhibit significant performance degradation on OOD tasks when compared with pre-trained models, which have stronger generalization capabilities. The primary reason for the performance drop lies in: fine-tuning updates the parameters of pre-trained models to fit ID datasets, often at the cost of perturbing generalizable features (Kumar et al., 2022; Zhu et al., 2024). Therefore, a better balance is needed between preserving generalization and incorporating task-specific adaptations.

In this work, we investigate the feasibility of pruning parameters of task vectors and only keeping the corresponding parameters in pre-trained models to enhance the generalization performance on OOD data of merged models. The central challenge lies in designing a principled criterion for identifying and removing task vector components that do not contribute meaningfully to ID tasks. Recall that task vectors can be interpreted as perturbations that align pre-trained models to specific tasks (Yadav et al., 2023; Yang et al., 2024b;a; Ilharco et al., 2023; Du et al., 2024). Shown in (Li et al., 2025), these vectors encode both discriminative and redundant information. With in-depth analysis, we posit that task-specific discriminative features tend to be diverse across tasks, while redundant or low-signal features exhibit consistency across task vectors, indicating low task-specific relevance. This motivates the hypothesis that shared patterns in task vectors reflect low-salience, potentially non-discriminative modifications that do not meaningfully contribute to performance, especially in OOD settings. More interestingly, we find that this diversity of task-specific discriminative features is not uniformly distributed across the model, but rather exhibits a layer-wise structure. Based on these insights, we propose an adaptive method, LwPTV (**L**ayer-**w**ise **P**runing **T**ask **V**ector for model merging). We define a layer-wise salience score, quantifying the deviation of a given task vector from the layer-wise mean across tasks. Formally, for each parameter in a layer, this score captures the absolute difference between its task-specific value and the mean across task vectors. A low salience score implies the parameter shift is consistent across tasks and thus likely redundant, motivating its removal in favor of the original pre-trained value. We construct layer-wise masks based on this score and additionally define a shared mask to preserve ID performance by retaining critical task-specific adaptations across all tasks. Ours can be used as a plug-and-play approach to enhance OOD generalization of most of model merging methods due to its flexibility. Extensive experiments prove that ours effectively maintains the pre-trained model's generalization ability while leveraging task-specific adaptations, leading to improved OOD robustness while preserving ID performance.

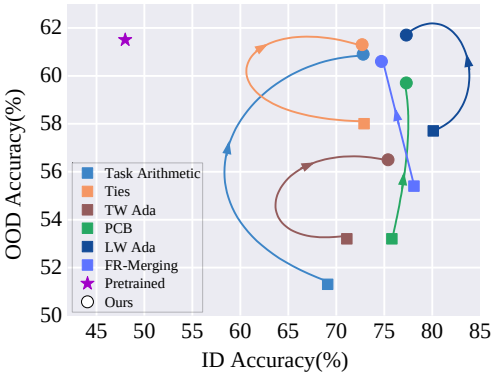


Figure 1: ID and OOD performance of model merging methods on ViT-B/32; see ID and OOD tasks in Experiments. Square and circle markers denote baseline and baseline+ours.

2 Related work

Multi-task Learning. MTL involves training a model on data from multiple tasks simultaneously, with the aim of leveraging shared representations to enhance performance across all tasks (Vandenhende et al., 2021; Zhang et al., 2023). However, when confronted a new task, MTL requires access to labeled data from

multiple tasks to train from scratch. This leads to high training costs and limited data accessibility due to privacy concerns (Pruksachatkun et al., 2020; Poth et al., 2021a; Weller et al., 2022; Fifty et al., 2021).

Model Merging. Model merging (Wortsman et al., 2022a; Matena & Raffel, 2022; Ilharco et al., 2023; Yadav et al., 2023) aims to combine several fine-tuned task-specific models into a single, unified multi-task model, by utilizing the existing task-specific model weights without requiring additional training. Model merging can be classified into two types. The first type involves merging multiple models that were trained on the same task, with the goal of improving the overall generalization of the model (Gupta et al., 2020; Cha et al., 2021; Ainsworth et al., 2023; Singh & Jaggi, 2020). Another type focuses on merging models for different tasks in order to enable MTL (Wortsman et al., 2022a; Ilharco et al., 2023; Yadav et al., 2023; Yang et al., 2024a).

The second type is our focus. A naive method is model averaging technique, which often leads to significant performance degradation (Wortsman et al., 2022a). To address the performance gap between fine-tuned models and the merging model in ID tasks, various methodologies have been proposed. Fisher Merging (Matena & Raffel, 2022) and RegMean (Jin et al., 2023) enhance model merging by utilizing Fisher information matrices and inner-product matrices, respectively, to calculate weighted coefficients for individual models. Task Arithmetic (Ilharco et al., 2023) merges models introduce task vector, defined as the parameter differences between fine-tuned models and the pre-trained model, which is effective in model merging. As a follow up, some works are further proposed. For example, Ties-Merging (Yadav et al., 2023) aims to tackle the task conflicts among multiple models; AdaMerging (Yang et al., 2024b) introduces the adaptive learning about the merging coefficients; Surgery method (Yang et al., 2024a) addresses representation bias between merged model and task-specific models; DARE (Yu et al., 2024) introduces a preprocessing step called drop and rescale, which reduces interference by randomly eliminating most elements and rescaling the remaining ones in each task vector before merging fine-tuned LLMs; WEMoE (Anke et al., 2024) dynamically combines shared and task-specific knowledge based on the input sample; PCB-MERGING (Du et al., 2024) effectively addresses parameter competition by adjusting parameter coefficients during the merging process; FR-Merging (Zheng & Wang, 2025) leverages the Fourier transform to remove low-frequency interference components from model parameters, effectively mitigating cross-task interference.

The methods discussed above focus on enhancing the ID performance of merging model. While Ties-Merging, AdaMerging, WEMoE, and PCB evaluate their performance on OOD data, their primary goal is still the ID tasks. In contrast, we aim to improve merged model on OOD tasks while preserving the ability to tackle the ID tasks. Besides, ours can be achieved only based on task vectors and can be combined with existing task vector-based methods in a flexible way.

Out-of-Distribution. A challenge commonly referred to as OOD generalization, continues to pose a substantial challenge in the field of machine learning. Despite the remarkable zero-shot capabilities demonstrated by large pre-trained models, such as CLIP (Li et al., 2022; Radford et al., 2021b), further finetuning on downstream tasks could potentially lead to decreased performance with OOD data (Nguyen et al., 2024; Kumar et al., 2022; Chen et al., 2023; Shuttleworth et al., 2024). Recent studies have proposed methods to mitigate this issue. WiSE-FT (Wortsman et al., 2022b) enhances the OOD performance of fine-tuned models by performing linear interpolation between the fine-tuned model and its corresponding pre-trained model. Model Stock exploits the anchoring effect of pretrained models and the geometric properties of fine-tuning parameters to utilize two fine-tuned models, obtained through optimization with different random seeds for the same task, to approximate the center of the parameter distribution (Jang et al., 2024). LiNeS (Wang et al., 2024a) employs a depth-dependent scaling strategy for parameter updates. These strategies aim to enhance the fine-tuned model's performance on OOD tasks. Different from them, ours aims to enhance the generalization capabilities of merging models, where we design the plug-and-play salience score to prune redundant parameters of task vectors in a layer-wise manner.

3 Preliminaries

Problem setup. Denote the pre-trained model as $f(\mathbf{x};\boldsymbol{\theta}_{pre})$, where $\boldsymbol{\theta}_{pre} = \{\boldsymbol{\theta}^1_{pre}, ...\boldsymbol{\theta}^l_{pre}, ...\boldsymbol{\theta}^L_{pre}\}$, L is the number of layers and $\boldsymbol{\theta}^l_{pre}$ is the parameter of the l-th layer. We aim to fine-tune the model on K

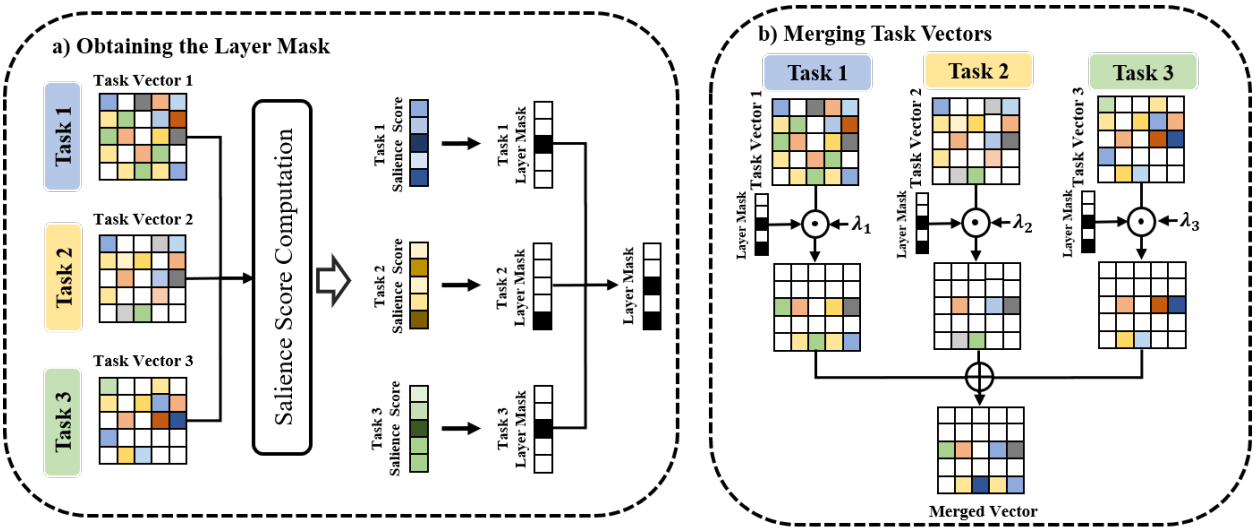


Figure 2: Illustration of our **LwPTV** framework: (a) Obtaining the Layer Mask; (b) Merging Task Vectors. Each row in the task-vector block corresponds to a layer, where white indicates 0 and black indicates 1. We compute layer-wise salience scores, threshold them to obtain masks, prune each task vector accordingly, and finally merge the pruned vectors into the pretrained model.

downstream tasks $\{T_k\}_{k=1}^K$ to get K finetuned models $\{f(\mathbf{x};\boldsymbol{\theta}_K)\}_{k=1}^K$, where each finetuned model has the same size parameters with pretrained model and $\boldsymbol{\theta}_k = \{\boldsymbol{\theta}_k^1, ...\boldsymbol{\theta}_k^l, ...\boldsymbol{\theta}_k^L\}$. Model merging aims to combine the weights $\{\boldsymbol{\theta}_k\}_{k=1}^K$ into a new set of weights $\boldsymbol{\theta}_m$, enabling $f(\mathbf{x};\boldsymbol{\theta}_m)$ to perform K tasks without retraining using task-specific data.

Task vector-based methods. A recent study (Ilharco et al., 2023) proposed the idea of "task vectors" for model merging, which has been further explored by subsequent research (Yadav et al., 2023; Yang et al., 2024b). A task vector $\boldsymbol{\tau}_k$ is defined as the difference between the fine-tuned model parameters $\boldsymbol{\theta}_k$ and the initial pre-trained parameters $\boldsymbol{\theta}_{pre}$. Specifically, for the k-th task, its task vector is given as:

$$\boldsymbol{\tau}_k = \boldsymbol{\theta}_k - \boldsymbol{\theta}_{pre}, \quad \boldsymbol{\tau}_k = \{\boldsymbol{\tau}_k^1, ...\boldsymbol{\tau}_k^l, ...\boldsymbol{\tau}_k^L\}. \tag{1}$$

Now, based on the task vectors, the merged function can be expressed as $\mathcal{F}(\cdot)$ as follows:

$$\boldsymbol{\theta}_m = \mathcal{F}(\boldsymbol{\theta}_{pre}, \boldsymbol{\tau}_1, \boldsymbol{\tau}_2, \cdots, \boldsymbol{\tau}_K). \tag{2}$$

For example, in terms of Task Arithmetic, the merged model weights can be expressed as $\boldsymbol{\theta}_m = \boldsymbol{\theta}_{pre} + \lambda \sum_{k=1}^K \boldsymbol{\tau}_k$, where λ denotes the merging coefficient. As for Task-wise AdaMerging, the merged model weights are computed as $\boldsymbol{\theta}_m = \boldsymbol{\theta}_{pre} + \sum_{k=1}^K \lambda_k \boldsymbol{\tau}_k$, where λ_k denotes the merging coefficient corresponding to the task vector $\boldsymbol{\tau}_k$. However, existing methods primarily focus on improving the accuracy of ID data, neglecting the performance on out-of-distribution OOD data.

4 Our proposed method

This work introduces a novel plug and play method, denoted as Layer-wise Pruning Task Vector for model merging (LwPTV), to improve the generalization capabilities of merging models, whose overview is shown in Fig.2. We first give the motivation and analysis about the task vector in Sec.4.1 and then introduce the proposed method in Sec.4.2.

4.1 Motivation and analysis about task vector

As shown in Fig.1, the merged model exhibits inferior generalization capabilities on OOD tasks but superior performance on ID tasks when compared with the pre-trained model. According to Eq. 2, merged model $\boldsymbol{\theta}_m$ is composed of $\boldsymbol{\theta}_{pre}$ and $\{\boldsymbol{\tau}_k\}_{k=1}^K$. The pre-trained model mainly contains valuable parameters for handling

OOD tasks as indicated by Fig.1, and $\boldsymbol{\tau}_k$ have the parameters beneficial for the k-th ID task. Motivated by this observation, our idea to improve OOD generalization for merged models by discarding certain redundant parameter layers within the task vectors and replacing them with the corresponding layers from the pretrained model, making the merged models "closer" to pretrained models where necessary. By doing this, we face two fundamental questions: (1) ***Which parameters from task vectors should be pruned?*** (2) ***Can the pruning maintain the performance of merged model on ID tasks?***

Before answering these questions, we provide a theoretical analysis of the task vector inspired by Li et al. (2025), a state-of-the-art theoretical work on the generalization of task vectors. Li et al. (2025) theoretically analyzes one-layer Transformer models based on discriminative patterns that are unique for different tasks. From Corollary 2 and Lemma 1 of Li et al. (2025), one can see that, some neurons of the task vectors for Transformer models can learn the discriminative patterns, while other neurons cannot. Without loss of generality, our theoretical analysis focuses on Transformer models e.g., ViT (Dosovitskiy et al., 2020) and considers one layer, where the task vector in (2) is simplified to $\boldsymbol{\tau}_k = \boldsymbol{\tau}_k^1$. Let $g(\boldsymbol{\tau}_k, \mathcal{S})$ denote the weights of the neuron at the index set $\mathcal{S} \subset \mathcal{L}$ in the task vector $\boldsymbol{\tau}_k$, where $\mathcal{L}$ is the set of all neuron indices. Define the diversity (DV) of $\boldsymbol{\tau}_k$ at neurons from $\mathcal{S}$, with respect to a collection $\{\boldsymbol{\tau}_j\}_{j=1}^K$, as

$$\mathrm{DV}\left(g(\boldsymbol{\tau}_k, \mathcal{S}); \{\boldsymbol{\tau}_j\}_{j=1}^K\right) = \mathbb{E}\Big[\Big\| g(\boldsymbol{\tau}_k, \mathcal{S}) - \frac{1}{K}\sum_{j=1}^{K} g(\boldsymbol{\tau}_j, \mathcal{S})\Big\|\Big]. \tag{3}$$

Then, we can derive the following proposition, the proof of which can be found in Appendix A.1.

Proposition 1. *With a high probability, there exists a set of neurons with indices in $\mathcal{S} \in \mathcal{L}$, $|\mathcal{S}| \geq 1$, where all neurons in $\mathcal{S}$ for $\boldsymbol{\tau}_{k_1}$ learn discriminative features, while all neurons in $\mathcal{S}$ for $\boldsymbol{\tau}_{k_2}$ fail to learn discriminative features, $k_1 \neq k_2 \in [K]$. Then, we have*

$$DV(g(\boldsymbol{\tau}_{k_1}, \mathcal{S}); \{\boldsymbol{\tau}_j\}_{j=1}^K) > \sqrt{K} \cdot \Omega(DV(g(\boldsymbol{\tau}_{k_2}, \mathcal{S}); \{\boldsymbol{\tau}_j\}_{j=1}^K)) \tag{4}$$

This result formalizes the intuition that discriminative neurons (parameters) of task vectors exhibit significantly higher variability across tasks compared to non-discriminative ones. It thus provides a theoretical justification for using diversity as a criterion for selective pruning. In particular, parameters with low diversity, which do not contribute to task-specific discriminative features, can be considered redundant or non-informative. Consequently, pruning these low-diversity components from $\boldsymbol{\tau}_k$ and replacing them with their counterparts from the pre-trained model $\boldsymbol{\theta}_{pre}$ is unlikely to degrade performance on ID tasks. Instead, it preserves ID performance while recovering the generalization ability of the merged model by reintroducing OOD-relevant information encoded in $\boldsymbol{\theta}_{pre}$.

4.2 Layer-wise pruning of task vectors

Based on the aforementioned analysis, we consider a straightforward method to enhance the generalization ability of merging models while preserving its ID performance: Pruning the parameters in task vectors with low diversity. Recalling that in the deep neural network model, the information learned by each layer is usually different, and the different layers in each task vector usually have different contributions for the merging model as stated by Yang et al. (2024b). Motivated by Eq.3, we introduce a layer-wise salience score:

$$s_k^l = \mathbb{E}\Big[\Big\| \boldsymbol{\tau}_k^l - \frac{1}{K}\sum_{k=1}^{K} \boldsymbol{\tau}_k^l \Big\|\Big], s_k^l \in \mathbb{R}_{\geq 0}, \tag{5}$$

where s_k^l represents the salience score of the l-th layer of the $\boldsymbol{\tau}_k$. In Eq.5, we first compute the difference between k-th task vector and mean of all task vectors in layer l, which is a d_l-dimensional vector with d_l denoting the dimension of $\boldsymbol{\tau}_k^l$; then we compute the absolute average of the vector along the dimension d_l, resulting in a scalar value. We use s_k^l to estimate whether the layer-wise parameters contain task-specific information, i.e. have learned the task-relevant discriminative features. The greater the distance between

k-th task vector and mean task vectors, the larger the s_k^l, indicating that the l-th layer of the k-th fine-tuned model contains more task-specific local information. On the contrary, if the value is smaller, it means that the k-th task is close to all tasks in current layer. That is to say, the parameters of the current layer are redundant, failing to learn the task-relevant discriminative features, so the parameters in this layer can be pruned.

To solve the first problem discussed above, we denote the mask vector as a L-dimensional vector, i.e., $\boldsymbol{m}_k = \{m_k^l\}_{l=1}^L \in \mathbb{R}^L$ for the k-th task and we design it according to salience scores:

$$m_k^l = \begin{cases} 1, & if \quad s_k^l > \text{sorted}(\mathbf{s}_k)[\lfloor L \cdot \eta \rfloor], \\ 0, & \text{otherwise}, \end{cases} \tag{6}$$

where η is the hyperparameter controlling the pruning ratio, and $\text{sorted}(\mathbf{s}_k)[\lfloor L \cdot \eta \rfloor]$ represents the ηL-th smallest salience score, ensuring that only the top $1-\eta$ fraction of layers with the highest salience scores are retained. In other words, for each task vector, layers whose salience scores significantly larger are retained, while layers with lower salience scores—indicating redundant parameters—are replaced with the corresponding pre-trained parameters. This adaptive pruning mechanism ensures that the merged model maintains task-specific adaptations while leveraging the generalization capabilities of the pre-trained model, thereby enhancing OOD robustness. Considering the mask defined independently for each task vector, there remains a risk that some task-specific mask vectors may remove parameter layers that are crucial for their respective tasks. To solve the second question mentioned earlier, i.e, maintaining the ability of the merged model on ID tasks as much as possible, we introduce a shared mask vector that consolidates information across all mask vectors:

$$\hat{\mathbf{m}} = \mathbf{m}_1 \vee \mathbf{m}_2 \vee \cdots \vee \mathbf{m}_K, \tag{7}$$

where $\hat{\mathbf{m}}_k \in \mathbb{R}^L$ and $\vee$ is the OR operation. That is to say, a layer l in the merged model will be pruned only if all task-specific mask vectors agree to remove it, i.e., $\hat{m}_k^l = 0$ if and only if $m_{1:K}^l = 0$. This ensures that if at least one task requires a particular layer to be retained, we remain it intact in the merged model, thereby preserving critical task-specific information.

Our proposed masking strategy is flexible so that it can be integrated with existing task vector-based merging methods. Specifically, it can be applied to the Task Arithmetic, where the merging model $\hat{\boldsymbol{\theta}}$ is formulated as follows:

$$\hat{\boldsymbol{\theta}}_m = \boldsymbol{\theta}_{pre} + \hat{\mathbf{m}} \odot \sum_{k=1}^{K} \lambda_k \boldsymbol{\tau}_k. \tag{8}$$

In which, the $\{\lambda_k\}_{k=1}^K$ values are identical. Besides, this mask vector can also be applied to other methods. For example, for AdaMerging series methods, which are designed to optimize coefficient λ_k or λ_k^l with entropy minimization, we can optimize λ_k or λ_k^l based on equation 8. Considering the learnable λ_k^l will be amplified when we introducing the mask, we can further use $\hat{\lambda}_k^l = \eta \cdot \lambda_k^l$ to scale the coefficient after optimizing the equation 8. A detailed analysis is in the B.1 of Appendix. We defer the workflow when combining ours with existing methods to Algorithm B.2 of Appendix.

Remark 1. *By combining the definition of salience score in (5), Proposition 1 indicates that parameters that learn task-specific discriminative patterns result in a large salience score because these parameters are diverse with a large variance. Then, that layer in the merged model will be maintained because it preserves task-specific information. In contrast, parameters that cannot learn task-specific discriminative patterns are close to each other, leading to a small salience score. Then, that layer in the merged model will be pruned. This theoretically explains the success of the proposed layer-wise pruning algorithm, which only maintains layers that learn discriminative patterns.*

5 Experiments

Datasets. Following prior work (Ilharco et al., 2023; Yadav et al., 2023; Yang et al., 2024b), we conduct model merging on eight image classification datasets: Cars (Krause et al., 2013), DTD (Cimpoi et al., 2014), EuroSAT (Helber et al., 2019), GTSRB (Stallkamp et al., 2011), MNIST (Deng, 2012), RESISC45 (Cheng

Table 2: Performance comparison for ID and OOD datasets on ViT-B/32 and ViT-L/14.

	Dataset (→)	In Domain	Out of Domain													H-score
	Method (↓)	Avg.	C10	C100	EMN	FMN	FER	F102	F101	KMN	OxP	PCA	RSS	STL	Ima	Avg.	
ViT-B/32	Pretrained	48.0	89.8	64.2	17.2	63.0	39.0	66.3	82.7	9.8	87.4	60.6	58.6	97.1	63.3	61.5	53.9
	Individual	90.5	65.5	37.3	18.2	54.8	32.2	47.8	58.3	8.5	76.9	54.1	55.9	88.4	48.5	49.7	64.2
	Weight Averaging (Wortsman et al., 2022a)	65.8	89.5	64.2	28.8	65.1	36.4	64.5	79.7	7.4	86.7	59.4	57.7	96.4	61.6	61.3	63.5
	Task Arithmetic (Ilharco et al., 2023)	69.1	76.3	41.9	28.5	63.9	26.3	49.4	55.8	**9.0**	75.4	54.0	53.4	87.8	45.0	51.3	58.9
	w/ LwPTV (Ours)	**72.8**	**88.3**	**60.1**	**30.4**	**65.9**	**36.5**	**62.9**	**79.9**	7.4	**85.4**	**59.5**	**59.0**	**96.4**	**60.6**	**60.9**$^{+9.6}$	**66.3**$^{+7.4}$
	Ties (Yadav et al., 2023)	**72.9**	85.3	55.2	26.5	64.1	33.6	59.0	74.0	**8.5**	83.0	58.4	55.6	94.4	56.3	58.0	64.6
	w/ LwPTV (Ours)	72.7	**89.1**	**61.5**	**29.3**	**65.7**	**37.0**	**63.5**	**80.4**	7.4	**85.5**	**59.8**	**59.4**	**96.6**	**61.1**	**61.3**$^{+3.3}$	**66.5**$^{+1.9}$
	TW AdaMerging (Yang et al., 2024b)	71.1	78.6	45.9	30.6	**63.5**	28.7	51.1	59.9	**8.2**	79.0	53.4	54.0	89.1	49.5	53.2	60.9
	w/ LwPTV (Ours)	**75.4**	**81.2**	**47.0**	**31.3**	62.6	**32.3**	**56.3**	**73.1**	7.7	**81.2**	**55.7**	**57.8**	**93.7**	**54.1**	**56.5**$^{+3.3}$	**64.6**$^{+3.7}$
	TW AdaMerging++ (Yang et al., 2024b)	73.7	79.3	45.5	28.1	**63.2**	30.4	51.7	64.3	**8.3**	78.1	54.6	54.1	90.6	50.6	53.7	62.1
	w/ LwPTV (Ours)	**76.1**	**83.2**	**49.1**	**30.7**	62.8	**32.5**	**55.4**	**73.8**	8.0	**79.9**	**56.3**	**56.7**	**94.2**	**53.5**	**56.6**$^{+2.9}$	**64.9**$^{+2.8}$
	LW AdaMerging (Yang et al., 2024b)	**80.1**	82.1	50.7	28.7	63.2	37.8	58.3	73.8	**8.8**	82.4	57.3	58.0	94.3	54.6	57.7	67.1
	w/ LwPTV (Ours)	77.3	**88.8**	**61.3**	**30.6**	**66.1**	**38.4**	**64.3**	**80.6**	7.6	**86.3**	**60.1**	**59.4**	**96.6**	**61.2**	**61.7**$^{+4.0}$	**68.6**$^{+1.5}$
	LW AdaMerging++ (Yang et al., 2024b)	**81.1**	83.9	51.7	27.2	62.7	36.1	57.4	74.2	**9.1**	82.4	56.4	57.8	94.3	54.7	57.6	67.4
	w/ LwPTV (Ours)	77.1	**89.0**	**61.8**	**29.9**	**65.8**	**37.5**	**63.9**	**80.4**	7.8	**86.0**	**59.6**	**58.9**	**96.5**	**61.1**	**61.4**$^{+3.8}$	**68.4**$^{+1.0}$
	PCB-MERGING (Du et al., 2024)	75.8	77.7	43.6	26.0	61.2	29.4	52.9	65.4	**8.8**	79.3	53.6	52.2	90.5	50.8	53.2	62.5
	w/ LwPTV (Ours)	**77.3**	**86.5**	**57.1**	**29.8**	**64.3**	**35.9**	**60.7**	**78.2**	7.6	**84.6**	**58.4**	**57.7**	**95.6**	**59.5**	**59.7**$^{+6.5}$	**67.4**$^{+4.9}$
	FR-Merging (Zheng & Wang, 2025)	**78.1**	76.6	45.9	23.8	61.7	38.5	54.8	69.6	**8.9**	82.6	55.8	55.4	90.7	55.5	55.4	64.8
	w/ LwPTV (Ours)	74.7	**87.1**	**60.3**	**26.7**	**64.5**	**38.7**	**63.8**	**79.1**	7.2	**86.4**	**60.4**	**56.5**	**95.6**	**61.5**	**60.6**$^{+5.2}$	**66.9** $^{+2.1}$
ViT-L/14	Pretrained	64.5	95.6	75.8	15.6	66.9	38.2	79.2	93.1	10.4	93.4	51.2	68.9	99.4	75.5	66.4	65.4
	Individual	94.2	88.6	65.2	19.3	66.6	38.5	73.7	89.4	10.2	92.4	59.4	62.4	97.8	72.5	64.3	76.4
	Weight Averaging (Wortsman et al., 2022a)	79.6	95.9	79.6	22.2	71.5	38.7	79.0	92.4	8.7	93.6	59.0	64.1	99.2	75.8	67.7	73.2
	Task Arithmetic (Ilharco et al., 2023)	84.5	93.4	73.0	**28.7**	73.2	38.2	74.2	88.6	**9.3**	92.7	**61.4**	58.6	98.2	72.9	66.3	74.3
	w/ LwPTV (Ours)	**85.6**	**95.6**	**78.8**	24.4	**73.4**	**40.1**	**77.6**	**92.5**	8.4	**93.8**	60.2	**63.4**	**99.1**	**76.0**	**67.9**$^{+1.6}$	**75.7**$^{+1.4}$
	Ties (Yadav et al., 2023)	86.0	94.5	74.6	**27.2**	72.5	38.8	75.1	90.3	8.5	92.9	**60.5**	59.7	98.5	74.2	66.7	75.1
	w/ LwPTV (Ours)	**86.9**	**95.1**	**76.9**	26.2	**73.2**	**39.6**	**76.5**	**92.3**	**8.7**	**93.7**	59.5	**63.7**	**99.0**	**75.7**	**67.7**$^{+1.0}$	**76.1** $^{+1.0}$
	TW AdaMerging (Yang et al., 2024b)	84.3	**90.9**	**67.1**	25.2	**71.9**	36.6	**69.3**	85.2	9.2	91.8	**59.4**	57.2	97.2	69.6	63.9	72.7
	w/ LwPTV (Ours)	**89.0**	90.4	65.4	**25.4**	70.7	**39.7**	68.5	**87.5**	**9.2**	**92.7**	58.6	**63.2**	**97.3**	**70.6**	**64.5**$^{+0.6}$	**74.8** $^{+2.1}$
	TW AdaMerging++ (Yang et al., 2024b)	87.5	91.3	66.7	24.5	**71.0**	36.9	**69.1**	86.7	8.2	91.8	57.5	57.6	97.5	70.2	63.7	73.7
	w/ LwPTV (Ours)	**89.7**	**91.3**	**67.0**	**25.8**	70.6	**39.2**	68.6	**88.4**	**8.4**	**92.6**	**57.8**	**65.3**	**97.6**	**71.2**	**64.9**$^{+1.2}$	**75.3**$^{+1.6}$
	LW AdaMerging (Yang et al., 2024b)	**90.8**	92.2	69.5	**29.9**	72.1	38.5	70.8	90.1	8.6	92.6	**60.1**	63.3	98.0	72.5	66.0	76.4
	w/ LwPTV (Ours)	89.8	**95.1**	**77.0**	26.2	**73.3**	**40.4**	**75.2**	**92.1**	**8.7**	**93.6**	59.9	**65.5**	**98.9**	**75.4**	**68.4**$^{+2.4}$	**77.7**$^{+1.3}$
	LW AdaMerging++ (Yang et al., 2024b)	**91.0**	93.5	71.8	**29.9**	72.3	38.4	71.2	90.3	8.7	92.8	**60.6**	60.4	98.2	72.7	66.2	76.6
	w/ LwPTV (Ours)	89.7	**95.1**	**77.3**	28.0	**73.1**	**40.1**	**75.0**	**92.1**	**8.7**	**93.6**	59.3	**65.7**	**99.0**	**75.3**	**67.9**$^{+1.7}$	**77.3**$^{+0.7}$
	PCB-MERGING (Du et al., 2024)	87.6	93.0	70.9	**27.6**	72.3	38.8	74.0	89.6	**9.1**	92.7	**61.1**	57.8	98.3	73.5	66.0	75.3
	w/ LwPTV (Ours)	**88.1**	**94.6**	**75.3**	27.5	**73.0**	**39.8**	**76.2**	**91.9**	8.8	**93.5**	60.1	**62.6**	**98.9**	**75.5**	**67.5**$^{+1.5}$	**76.4** $^{+1.1}$
	FR-Merging (Zheng & Wang, 2025)	**88.3**	92.3	71.3	27.3	71.6	38.9	75.7	90.9	**10.1**	93.6	**59.9**	61.7	98.5	74.5	66.6	75.9
	w/ LwPTV (Ours)	87.8	**94.0**	**75.3**	**28.6**	**72.5**	**39.2**	**76.9**	**92.0**	9.1	**93.9**	59.6	**62.9**	**98.8**	**75.5**	**67.6**$^{+1.0}$	**76.4** $^{+0.5}$

et al., 2017), SUN397 (Xiao et al., 2016), and SVHN (Netzer et al., 2011). To assess OOD generalization, we evaluate the merged models on 13 OOD benchmarks (Wang et al., 2024b; Zhou et al., 2022): CIFAR10 (C10) (Krizhevsky et al., 2009), CIFAR100 (C100) (Krizhevsky et al., 2009), EMNIST (EMN) (Cohen et al., 2017), FashionMNIST (FMN)(Xiao et al., 2017), FER2013 (FER) (Goodfellow et al., 2013), Flowers102 (F102) (Nilsback & Zisserman, 2008), Food101 (F101) (Bossard et al., 2014), KMNIST (Clanuwat et al., 2018) (KMN), OxfordIIITPet (OxP) (Parkhi et al., 2012), PCAM (PCA) (S.Veeling et al., 2018), RenderedSST2 (RSS) (Socher et al., 2013; Radford et al., 2019), STL10 (STL) (Coates et al., 2011) and ImageNet-1k (Ima) (Deng et al., 2009). Additional dataset details are provided in Appendix B.3.

Baselines & details. We consider following baselines, including Weight Averaging (Wortsman et al., 2022a), Fisher Merging (Matena & Raffel, 2022), RegMean (Jin et al., 2023), Task Arithmetic (Ilharco et al., 2023), Ties-Merging (Yadav et al., 2023), AdaMerging (Yang et al., 2024b), Surgery (Yang et al., 2024a), PCB-MERGING (Du et al., 2024) and FR-Merging (Zheng & Wang, 2025). We employ ViT-B/32, ViT-L/14 and ViT-H/14 from CLIP (Radford et al., 2021a) as our pre-trained models, respectively. Motivated by (Zhu et al., 2024), we employ the H-score as evaluation metric, defined as the harmonic mean of the performance on ID and OOD datasets and stated in Appendix B.4. For image classification tasks, our method uses the baseline hyperparameters from the original references, with a fixed pruning ratio of $\eta = 0.7$. Detailed configurations are provided in Appendix B.21.

5.1 Main results

Comparison with other methods. We summarize the characteristics of baselines and ours in Tab.1 for a clear comparison. As listed in Tab.2, we evaluate the performance of merging model ViT-B/32 and ViT-L/14 on ID and OOD datasets respectively. Individual indicates K task-specific fine-tuned model parameters $\{\boldsymbol{\theta}_k\}_{k=1}^K$ without model merging. For ID tasks, each task is only tested by the corresponding individual model; for OOD tasks, each OOD task can be evaluated by all individual models, respectively,

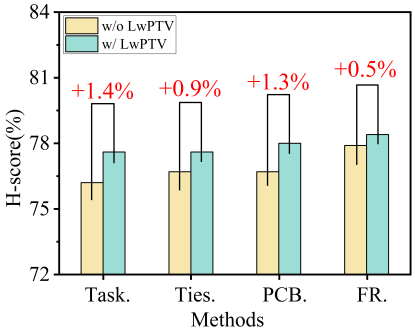


Figure 3: H-score of existing model merging methods with and without ours on ViT-H/14.

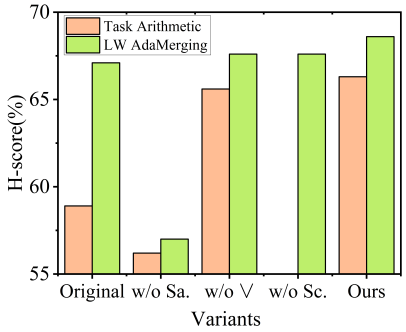


Figure 4: Ablation on ViT-B/32; Sa, ∨, and Sc denote Salience score, OR, and scaling.

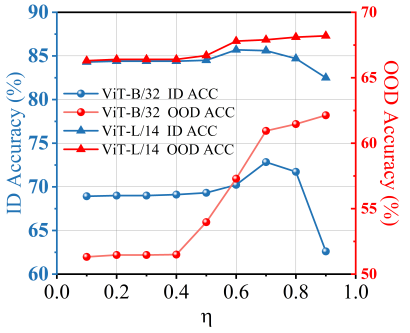


Figure 5: Performance of Task Arithmetic+LwPTV with varying η.

followed by the average operation. Due to the unavailability of Fisher Merging and RegMean, and their suboptimal performance on ID datasets, we did not test their performance on OOD datasets. Besides, Surgery is not available in OOD tasks since it needs to learn a representation surgery module on ID task. The detailed performance of various methods including Fisher Merging, RegMean and Surgery on the ID dataset are deferred to Appendix B.5 due to limited space.

From Tab.2, we have the following observations: (1) For the non-model merging baseline, the Pretrained model exhibits robust generalization capabilities on OOD datasets, whereas the Individual fine-tuned models demonstrates inferior OOD performance and superior ID performance. The underlying cause is that the Pretrained model acquires generalized features from the large-scale pre-training datasets and the Individual Fine-tuned model undergoes feature distortion specific to the current task. (2) For the model merging baselines, Weight Averaging is equivalent to computing the mean of various task vectors and incorporating these into the pretrained model. This method minimizes alterations to the parameters of the pre-trained model, thereby yielding superior OOD generalization performance. Ties prunes task vectors, yielding good OOD performance but relatively weak ID results. TW/LW AdaMerging and PCB aggressively reweight task vectors, and FR-Merging amplifies task-specific high-frequency signals, which improves ID accuracy but harms OOD robustness due to heavy perturbations to pretrained parameters. (3) LwPTV is a plug-and-play technique that enhances the generalization of existing model merging schemes on OOD datasets while minimizing the degradation on (even improving) the ID performance. For instance, on ViT-B/32, ours yields additional improvements of 7.4%, 1.9%, 1.5%, 4.9%, and 2.1% in H-score values compared to the baseline results of Task Arithmetic, Ties, LW AdaMerging, PCB and FR-Merging, respectively. (4) To investigate whether ours is effective for more larger pre-trained model, we conducted experiments on ViT-H/14 in Fig.3, proving its effectiveness and robustness on pre-trained model; see more details in Appendix B.6.

Effect of our components. We conduct ablation studies based on Task Arithmetic and LW AdaMerging with ViT-B/32. LwPTV includes three components: salience score, OR operation ($\vee$), and coefficient scaling (scale) for LW AdaMerging, which learns the layer-wise merging weights. To systematically investigate the contribution of each individual component, we remove each component and observe the resultant impact. Especially, for w/o salience score, we design the pruning metric based on the mean absolute value of the task vectors across each layer i.e., $\mathbb{E}(|\boldsymbol{\tau}_k^l|)$. The results are illustrated in Fig.4; see details in Tab.8 and Tab.9 of Appendix. We can find that w/o salience score but using $\mathbb{E}(|\boldsymbol{\tau}_k^l|)$ can degrade the H-score a lot, where the performance drop is mainly on ID. Besides, those variants with salience score are better than original merging model methods, indicating the effectiveness of our designed salience score in balancing ID and OOD. In addition, w/o OR operation means that we discard more layer-wise parameters in task vectors depending only on salience score, which also reduces the H-score of merged model. When combining ours with LW AdaMerging, removing scaling strategy is also harmful for the merged model. It proves the effectiveness of our proposed OR operation and scaling strategy.

Comparison with OOD methods for fine-tuned models. To explore the advantages of ours, we consider following methods specifically designed for enhancing the OOD performance of fine-tuned models: WiSE-FT (Wortsman et al., 2022b), Model Stock (Jang et al., 2024), and LiNeS (Wang et al., 2024a). Among them, WiSE-FT and Model Stock are specifically designed for the CLIP architecture, and LiNeS targets the

ViT architecture within CLIP. We implement these methods on fine-tuned models and then merge the multiple fine-tuned models with Task Arithmetic. As listed in Table 3, all methods can enhance both ID and OOD performance of the merged model, but ours performs best. As a strong baseline, LiNeS outperforms LwPTV in terms of ID performance, which can be attributed to its amplification of task-specific parameters. However, it is still inferior to LwPTV about the OOD performance and overall H-score. Additionally, for different methods and architectures, LiNeS requires the validation set to determine the optimal scaling coefficient, whereas our pruning ratio of 0.7 is generally applicable. This further demonstrates the advantage of LwPTV in balancing the ID and OOD performance of the merged model. See more details in Appendix B.8.

Comparison with other pruning methods. To investigate the superiority of our method over other pruning methods, we employ the following pruning baselines: (1) DARE (Yu et al., 2024), Drops delta parameters with a ratio p And REscales; (2) Magnitude-based Weight Pruning (MWP) (Sanh et al., 2020), which determines the mask based on the absolute value of each parameter within task vector; (3) random mask, wherein the mask is stochastically generated via a Bernoulli distribution; (4) the mean absolute value of the task vectors per layer, denoted as absolute value. As depicted in Tab.3, LwPTV's H-score exceeds that of the most favorable Magnitude-based Weight Pruning among the four baselines by 3.2%. Although random mask exhibits superior performance on the OOD dataset, it incurs a significant performance deterioration on the ID dataset. This demonstrates that LwPTV's effectiveness of proposed salience score in pruning parameters. See more details in Appendix B.9.

Table 3: Comparison with other OOD methods (above) and pruning methods (below) on ViT-B/32.

	Performance (%) (→) Method (↓)	ID. Avg	OOD. Avg	H-score
Baseline	Task Arithmetic (Ilharco et al., 2023)	69.1	51.3	58.9
OOD	w/ WiSE-FT (Wortsman et al., 2022b)	70.5	55.4	62.0
	w/ Model Stock (Jang et al., 2024)	72.0	58.7	64.7
	w/ LiNeS (Wang et al., 2024a)	74.1	58.0	65.1
Pruning	w/ DARE (Yu et al., 2024)	49.1	61.5	54.6
	w/ MWP (Sanh et al., 2020)	71.7	56.4	63.1
	w/ random mask	49.1	61.7	54.7
	w/ absolute value	48.9	61.5	54.5
Ours	**w/ LwPTV**	72.8	60.9	**66.3** (+7.4)

5.2 Additional analysis

Trade-off between ID and OOD. To explore the impact of pruning ratio η on the trade-off between ID and OOD task performance, we conducted the experiments using Task Arithmetic w/ LwPTV on the ViT-B/32 and ViT-L/14. Notably, η serves as a parameter to balance the pretrained model and task vectors of the fine-tuned models. $\eta = 1$ signifies a predominance of the pretrained model, while $\eta = 0$ represents the original merging model. We can see that the OOD performance gradually increases with a development pruning ratio. Conversely, the ID task performance shows an initial improvement followed by a decline. It is reasonable since a proper pruning ratio can increase the ID and OOD performance for pruning redundant parameters and improve the generalization performance. And a too large pruning ratio will prune the useful parameters for ID tasks and hurt its performance, indicating the trade-off between ID and OOD tasks. Besides, a pruning ratio η between 0.6 and 0.8 offers a relatively desired trade-off, ensuring improved OOD performance while maintaining desired ID accuracy, where we set $\eta = 0.7$ for all experiments.

Salience score and mask vectors. To facilitate an intuitive analysis of salience score, masks associated with each task vector and their interrelationships, we visualize them for ViT-B/32 in Fig.6. We can see that, for different tasks, the salience score vectors usually have low values in certain same layers, indicating their redundancy in these layers. Besides, in each salience score vector, most of the elements have low values, which explains why the model produces better OOD and ID performance even when we throw away most layers (pruning ratio $\eta = 0.7$). Therefore, the mask vectors of different tasks have a lot of overlapping layers, which can be viewed as those parameters unimportant to all tasks. Those non-overlapping mask information might be viewed as the beneficial parameters for the corresponding ID task, which is the reason that we choose to keep them. Therefore, we adopt the OR operation to extract a shared mask vector from all masks, i.e., replacing these overlapping layers with the corresponding layers of pre-trained model. To explore whether ours can extract the discriminative features intuitively, we present a T-SNE visualization for features from Task Arithmetic and the Task Arithmetic+LwPTV on OOD tasks in Fig.7. Compared to Task Arithmetic, introducing our LwPTV can enhance the separability of representations across different categories. Due to

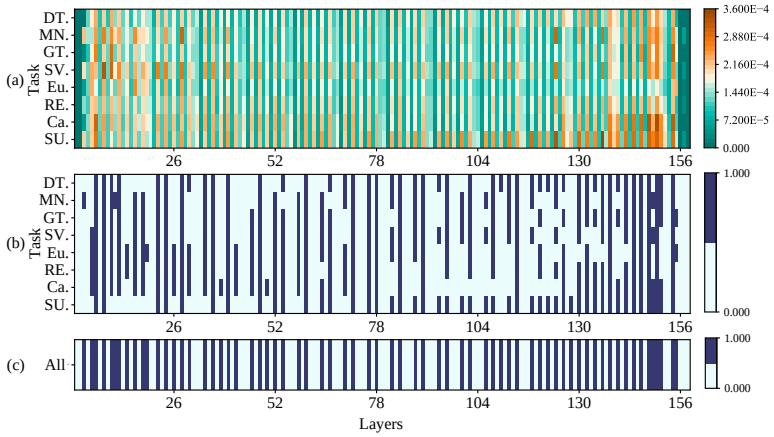


Figure 6: Visualization of (a) salience score matrix, (b) mask vector for each task, and (c) final mask vector, all on ViT-B/32, where x-axis denotes the layer index, y-axis in (a-b) denotes task name.

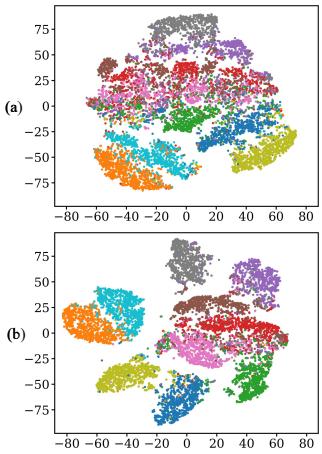


Figure 7: T-SNE visualizations of (a) Task Arithmetic, and (b) Task Arithmetic w/ LwPTV on CIFAR10.

space limitations, we report additional results in Appendix B and Appendix C, including NLP experiments, storage overhead analysis, and further experimental analyses.

6 Conclusion

In this work, we first systematically analyze existing model merging techniques and identify their limitations in generalizing to OOD data. To address this issue, we propose a novel plug-and-play framework, termed LwPTV, and introduce a training-free salience score to measure the redundancy of task vectors. By pruning the layer-wise parameters with low salience score, ours enhances the OOD generalization capability of merged models while preserving ID performance. Extensive experimental results demonstrate that our approach significantly improves the OOD generalization performance of existing model merging algorithms.

References

Samuel Ainsworth, Jonathan Hayase, and Siddhartha Srinivasa. Git re-basin: Merging models modulo permutation symmetries. In *Proceedings of the International Conference on Learning Representations (ICLR 2023)*, 2023.

Tang Anke, Shen Li, Luo Yong, Yin Nan, Zhang Lefei, and Tao Dacheng. Merging multi-task models via weight-ensembling mixture of experts. In *Proceedings of the International Conference on Machine Learning (ICML 2024)*, 2024.

Udit Arora, William Huang, and He He. Types of out-of-distribution texts and how to detect them. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP 2021)*, pp. 10687–10701, 2021.

Rishi Bommasani, Drew A Hudson, Ehsan Adeli, Russ Altman, Simran Arora, Sydney von Arx, Michael S Bernstein, Jeannette Bohg, Antoine Bosselut, Emma Brunskill, et al. On the opportunities and risks of foundation models. *arXiv preprint arXiv:2108.07258*, 2021.

Lukas Bossard, Matthieu Guillaumin, and Luc Van Gool. Food-101–mining discriminative components with random forests. In *Proceedings of the European Conference on Computer Vision (ECCV 2014)*, pp. 446–461, 2014.

Junbum Cha, Sanghyuk Chun, Kyungjae Lee, Han-Cheol Cho, Seunghyun Park, Yunsung Lee, and Sungrae Park. Swad: Domain generalization by seeking flat minima. In *Proceedings of the Conference on Neural Information Processing Systems (NeurIPS 2021)*, pp. 22405–22418, 2021.

Sishuo Chen, Wenkai Yang, Xiaohan Bi, and Xu Sun. Fine-tuning deteriorates general textual out-of-distribution detection by distorting task-agnostic features. In *Findings of the Association for Computational Linguistics: EACL 2023*, pp. 564–579, 2023.

Tianlong Chen, Zhenyu Zhang, Yu Cheng, Ahmed Awadallah, and Zhangyang Wang. The principle of diversity: Training stronger vision transformers calls for reducing all levels of redundancy. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR 2022)*, pp. 12010–12020, 2022.

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

Mircea Cimpoi, Subhransu Maji, Iasonas Kokkinos, Sammy Mohamed, and Andrea Vedaldi. Describing textures in the wild. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR 2014)*, pp. 3606–3613, 2014.

Tarin Clanuwat, Mikel Bober-Irizar, Asanobu Kitamoto, Alex Lamb, Kazuaki Yamamoto, and David Ha. Deep learning for classical japanese literature. *arXiv preprint arXiv:1812.01718*, 2018.

Adam Coates, Andrew Ng, and Honglak Lee. An analysis of single-layer networks in unsupervised feature learning. In *Proceedings of the International conference on artificial intelligence and statistics (AISTATS 2011)*, pp. 215–223, 2011.

Gregory Cohen, Saeed Afshar, Jonathan Tapson, and Andre Van Schaik. Emnist: Extending mnist to handwritten letters. In *Proceedings of the International Joint Conference on Neural Networks (IJCNN 2017)*, pp. 2921–2926, 2017.

Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR 2009)*, pp. 248–255, 2009.

Li Deng. The mnist database of handwritten digit images for machine learning research [best of the web]. *IEEE signal processing magazine*, 29(6):141–142, 2012.

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

Tim Dettmers, Artidoro Pagnoni, Ari Holtzman, and Luke Zettlemoyer. Qlora: Efficient finetuning of quantized llms. In *Proceedings of the Conference on Neural Information Processing Systems (NeurIPS 2023)*, 2023.

Yizhuo Ding, Xinwei Sun, Yanwei Fu, and Guosheng Hu. Revisiting large language model pruning using neuron semantic attribution. *arXiv e-prints*, pp. arXiv–2503, 2025.

Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. In *International Conference on Learning Representations*, 2020.

Guodong Du, Junlin Lee, Jing Li, Runhua Jiang, Yifei Guo, Shuyang Yu, Hanting Liu, Sim Kuan Goh, Ho-Kin Tang, Daojing He, et al. Parameter competition balancing for model merging. In *Proceedings of the Conference on Neural Information Processing Systems (NeurIPS 2024)*, 2024.

Aparna Elangovan, Jiayuan He, and Karin Verspoor. Memorization vs. generalization: Quantifying data leakage in nlp performance evaluation. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume (EACL 2021)*, pp. 1325–1335, 2021.

Chris Fifty, Ehsan Amid, Zhe Zhao, Tianhe Yu, Rohan Anil, and Chelsea Finn. Efficiently identifying task groupings for multi-task learning. In *Proceedings of the Conference on Neural Information Processing Systems (NeurIPS 2021)*, pp. 27503–27516, 2021.

Ian J Goodfellow, Dumitru Erhan, Pierre Luc Carrier, Aaron Courville, Mehdi Mirza, Ben Hamner, Will Cukierski, Yichuan Tang, David Thaler, Dong-Hyun Lee, et al. Challenges in representation learning: A report on three machine learning contests. In *Proceedings of the International Conference on Neural Information Processing (ICONIP 2013)*, pp. 117–124, 2013.

Vipul Gupta, Santiago Akle Serrano, and Dennis DeCoste. Stochastic weight averaging in parallel: Large-batch training that generalizes well. In *Proceedings of the International Conference on Learning Representations (ICLR 2020)*, 2020.

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

Zejiang Hou and Sun-Yuan Kung. Multi-dimensional model compression of vision transformer. In *Proceedings of the IEEE International Conference on Multimedia and Expo (ICME 2022)*, pp. 01–06, 2022.

Gabriel Ilharco, Marco Tulio Ribeiro, Mitchell Wortsman, Suchin Gururangan, Ludwig Schmidt, Hannaneh Hajishirzi, and Ali Farhad. Editing models with task arithmetic. In *Proceedings of the International Conference on Learning Representations (ICLR 2023)*, 2023.

Dong-Hwan Jang, Sangdoo Yun, and Dongyoon Han. Model stock: All we need is just a few fine-tuned models. In *Proceedings of the European Conference on Computer Vision (ECCV 2024)*, pp. 207–223, 2024.

Jiarui Jiang, Wei Huang, Miao Zhang, Taiji Suzuki, and Liqiang Nie. Unveil benign overfitting for transformer in vision: Training dynamics, convergence, and generalization. *Proceedings of the Conference on Neural Information Processing Systems (NeurIPS 2024)*, pp. 135464–135625, 2024.

Xisen Jin, Xiang Ren, Daniel Preotiuc-Pietro, and Pengxiang Cheng. Dataless knowledge fusion by merging weights of language models. In *Proceedings of the International Conference on Learning Representations (ICLR 2023)*, 2023.

Jacob Devlin Ming-Wei Chang Kenton and Lee Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the Annual Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT 2019)*, pp. 2, 2019.

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

Alex Krizhevsky et al. Learning multiple layers of features from tiny images. 2009.

Ananya Kumar, Aditi Raghunathan, Robbie Matthew Jones, Tengyu Ma, and Percy Liang. Fine-tuning can distort pretrained features and underperform out-of-distribution. In *Proceedings of the International Conference on Learning Representations (ICLR 2022)*, 2022.

Patrick Lewis, Pontus Stenetorp, and Sebastian Riedel. Question and answer test-train overlap in open-domain question answering datasets. In *Proceedings of the Conference of the European Chapter of the Association for Computational Linguistics: Main Volume (EACL 2021)*, pp. 1000–1008, 2021.

Hongkang Li, Meng Wang, Sijia Liu, and Chen Pin-Yu. A theoretical understanding of shallow vision transformers: Learning, generalization, and sample complexity. In *Proceedings of the International Conference on Learning Representations (ICLR 2023)*, 2023.

Hongkang Li, Meng Wang, Songtao Lu, Xiaodong Cui, and Pin-Yu Chen. How do nonlinear transformers learn and generalize in in-context learning? In *Proceedings of the International Conference on Learning Representations (ICLR 2024)*, pp. 28734–28783, 2024.

Hongkang Li, Yihua Zhang, Shuai Zhang, Pin-Yu Chen, Sijia Liu, and Meng Wang. When is task vector provably effective for model editing? a generalization analysis of nonlinear transformers. In *Proceedings of the International Conference on Learning Representations (ICLR 2025)*, 2025.

Manling Li, Ruochen Xu, Shuohang Wang, Luowei Zhou, Xudong Lin, Chenguang Zhu, Michael Zeng, Heng Ji, and Shih-Fu Chang. Clip-event: Connecting text and images with event structures. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 16420–16429, 2022.

Ashok Vardhan Makkuva, Marco Bondaschi, Adway Girish, Alliot Nagle, Hyeji Kim, Michael Gastpar, and Chanakya Ekbote. Local to global: Learning dynamics and effect of initialization for transformers. In *Proceedings of the Conference on Neural Information Processing Systems (NeurIPS 2024)*, pp. 86243–86308, 2024.

Michael S. Matena and Colin A. Raffel. Merging models with fisher-weighted averaging. In *Proceedings of the International Conference on Learning Representations (ICLR 2023)*, pp. 17703–17716, 2022.

Yuval Netzer, Tao Wang, Adam Coates, Alessandro Bissacco, Baolin Wu, Andrew Y Ng, et al. Reading digits in natural images with unsupervised feature learning. In *NIPS workshop on deep learning and unsupervised feature learning*, pp. 4, 2011.

Bac Nguyen, Stefan Uhlich, Fabien Cardinaux, Lukas Mauch, Marzieh Edraki, and Aaron Courville. Saft: Towards out-of-distribution generalization in fine-tuning. In *European Conference on Computer Vision*, pp. 138–154, 2024.

Maria-Elena Nilsback and Andrew Zisserman. Automated flower classification over a large number of classes. In *Proceedings of the Indian Conference on Computer Vision, Graphics & Image Processing (ICCVGIP 2008)*, pp. 722–729, 2008.

Omkar M Parkhi, Andrea Vedaldi, Andrew Zisserman, and CV Jawahar. Cats and dogs. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR 2012)*, pp. 3498–3505, 2012.

Clifton Poth, Jonas Pfeiffer, Andreas Rücklé, and Iryna Gurevych. What to pre-train on? efficient intermediate task selection. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP 2021)*, pp. 10585–10605, 2021a.

Clifton Poth, Jonas Pfeiffer, Andreas Rücklé, and Iryna Gurevych. What to pre-train on? efficient intermediate task selection. *Empirical Methods in Natural Language Processing,Empirical Methods in Natural Language Processing*, 2021b.

Yada Pruksachatkun, Jason Phang, Haokun Liu, Phu Mon Htut, Xiaoyi Zhang, Richard Yuanzhe Pang, Clara Vania, Katharina Kann, and Samuel R Bowman. Intermediate-task transfer learning with pretrained models for natural language understanding: When and why does it work? In *Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL 2020)*, pp. 5231–5247, 2020.

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

Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, Gretchen Krueger, and Ilya Sutskever. Learning transferable visual models from natural language supervision. In Marina Meila and Tong Zhang (eds.), *Proceedings of the International Conference on Machine Learning (ICML 2021)*, pp. 8748–8763. PMLR, 2021a.

Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language supervision. In *Proceedings of the International Conference on Machine Learning (ICML 2021)*, pp. 8748–8763, 2021b.

Hassan Sajjad, Fahim Dalvi, Nadir Durrani, and Preslav Nakov. On the effect of dropping layers of pre-trained transformer models. *Computer Speech and Language*, 77, 2023.

Victor Sanh, Thomas Wolf, and Alexander Rush. Movement pruning: Adaptive sparsity by fine-tuning. 2020.

Victor Sanh, Albert Webson, Colin Raffel, Stephen H Bach, Lintang Sutawika, Zaid Alyafeai, Antoine Chaffin, Arnaud Stiegler, Teven Le Scao, Arun Raja, et al. Multitask prompted training enables zero-shot task generalization. In *Proceedings of the International Conference on Learning Representations (ICLR 2022)*, 2022.

Eyal Shnarch, Alon Halfon, Ariel Gera, Marina Danilevsky, Yannis Katsis, Leshem Choshen, Martin Santillan Cooper, Dina Epelboim, Zheng Zhang, Dakuo Wang, et al. Label sleuth: From unlabeled text to a classifier in a few hours. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP 2022)*, 2022.

Reece Shuttleworth, Jacob Andreas, Antonio Torralba, and Pratyusha Sharma. Lora vs full fine-tuning: An illusion of equivalence. *arXiv e-prints*, pp. arXiv–2410, 2024.

Sidak Pal Singh and Martin Jaggi. Model fusion via optimal transport. In *Proceedings of the Conference on Neural Information Processing Systems (NeurIPS 2020)*, pp. 22045–22055, 2020.

Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D Manning, Andrew Y Ng, and Christopher Potts. Recursive deep models for semantic compositionality over a sentiment treebank. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP 2013)*, pp. 1631–1642, 2013.

Johannes Stallkamp, Marc Schlipsing, Jan Salmen, and Christian Igel. The german traffic sign recognition benchmark: a multi-class classification competition. In *Proceedings of the International Joint Conference on Neural Networks (IJCNN 2011)*, pp. 1453–1460, 2011.

Bastiaan S.Veeling, Jasper Linmans, Jim Winkens, Taco Cohen, and Max Welling. Rotation equivariant cnns for digital pathology. In *Proceedings of the International Conference on Medical Image Computing and Computer-Assisted Intervention (MICCAI 2018)*, pp. 210–218, 2018.

Simon Vandenhende, Stamatios Georgoulis, Wouter Van Gansbeke, Marc Proesmans, Dengxin Dai, and Luc Van Gool. Multi-task learning for dense prediction tasks: A survey. *IEEE transactions on pattern analysis and machine intelligence*, 44(7):3614–3633, 2021.

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

Ke Wang, Nikolaos Dimitriadis, Guillermo Ortiz-Jimenez, François Fleuret, and Pascal Frossard. Localizing task information for improved model merging and compression. In *Proceedings of the International Conference on Machine Learning (ICML 2024)*, 2024b.

Orion Weller, Kevin Seppi, and Matt Gardner. When to use multi-task learning vs intermediate fine-tuning for pre-trained encoder transfer learning. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL 2022)*, 2022.

Mitchell Wortsman, Gabriel Ilharco, Samir Ya Gadre, Rebecca Roelofs, Raphael Gontijo-Lopes, Ari S Morcos, Hongseok Namkoong, Ali Farhadi, Yair Carmon, Simon Kornblith, et al. Model soups: averaging weights of multiple fine-tuned models improves accuracy without increasing inference time. In *Proceedings of the International Conference on Machine Learning (ICML 2022)*, pp. 23965–23998, 2022a.

Mitchell Wortsman, Gabriel Ilharco, Jong Wook Kim, Mike Li, Simon Kornblith, Rebecca Roelofs, Raphael Gontijo Lopes, Hannaneh Hajishirzi, Ali Farhadi, Hongseok Namkoong, et al. Robust fine-tuning of zero-shot models. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition (CVPR 2022)*, pp. 7959–7971, 2022b.

Han Xiao, Kashif Rasul, and Roland Vollgraf. Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms. *arXiv preprint arXiv:1708.07747*, 2017.

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

Prateek Yadav, Derek Tam, Leshem Choshen, Colin Raffel, and Mohit Bansal. Resolving interference when merging models. In *Proceedings of the Conference on Neural Information Processing Systems (NeurIPS 2023)*, 2023.

Enneng Yang, Li Shen, Zhenyi Wang, Guibing Guo, Xiaojun Chen, Xingwei Wang, and Dacheng Tao. Representation surgery for multi-task model merging. In *Proceedings of the International Conference on Machine Learning (ICML 2024)*, 2024a.

Enneng Yang, Zhenyi Wang, Li Shen, Shiwei Liu, Guibing Guo, Xingwei Wang, and Dacheng Tao. Adamerging: Adaptive model merging for multi-task learning. In *Proceedings of the International Conference on Learning Representations (ICLR 2024)*, 2024b.

Le Yu, Bowen Yu, Haiyang Yu, Fei Huang, and Yongbin Li. Language models are super mario: Absorbing abilities from homologous models as a free lunch. In *Proceedings of the International Conference on Machine Learning (ICML 2024)*, 2024.

Bo Zhang, Jiakang Yuan, Botian Shi, Tao Chen, Yikang Li, and Yu Qiao. Uni3d: A unified baseline for multi-dataset 3d object detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR 2023)*, pp. 9253–9262, 2023.

Shenghe Zheng and Hongzhi Wang. Free-merging: Fourier transform for efficient model merging. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 3863–3873, 2025.

Kaiyang Zhou, Jingkang Yang, Chen Change Loy, and Ziwei Liu. Learning to prompt for vision-language models. *International Journal of Computer Vision*, 130(9):2337–2348, 2022.

Didi Zhu, Zhongyisun Sun, Zexi Li, Tao Shen, Ke Yan, Shouhong Ding, Chao Wu, and Kun Kuang. Model tailor: Mitigating catastrophic forgetting in multi-modal large language models. In *Proceedings of the International Conference on Machine Learning (ICML 2024)*, 2024.

Fuzhen Zhuang, Zhiyuan Qi, Keyu Duan, Dongbo Xi, Yongchun Zhu, Hengshu Zhu, Hui Xiong, and Qing He. A comprehensive survey on transfer learning. *Proceedings of the IEEE*, 109:43–76, 2020.

A Theoretical analyses

A.1 Proof of proposition 1

We start by introducing the theoretical setting.

Theoretical formulation. Consider a binary classification problem, where the target is to predict $y \in \{+1, -1\}$ given the input $\boldsymbol{X} = (\boldsymbol{x}_1, \boldsymbol{x}_2, \cdots, \boldsymbol{x}_P) \in \mathbb{R}^{d \times P}$ that contains P d-dimensional tokens. Following the state-of-the-art theoretical works (Li et al., 2023; Makkuva et al., 2024; Li et al., 2024; Jiang et al., 2024) on the generalization and learning dynamics of neural models, we use a one-layer single-head Transformer as the learner model for theoretical analysis, which can be characterized as $f(\boldsymbol{X}; \boldsymbol{\theta}) = 1/P \sum_{l=1}^{P} \boldsymbol{a}_l^\top \text{Relu}(\boldsymbol{V} \sum_{s=1}^{P} \boldsymbol{x}_s \text{softmax}_l(\boldsymbol{x}_s^\top \boldsymbol{W} \boldsymbol{x}_l))$, where $\boldsymbol{\theta} = \{\{\boldsymbol{a}_l\}_{l=1}^{P}, \boldsymbol{V}, \boldsymbol{W}\}$ with $\boldsymbol{a}_l \in \mathbb{R}^m$, $\boldsymbol{V} \in \mathbb{R}^{m \times d}$, and $\boldsymbol{W} \in \mathbb{R}^{d \times d}$. $\text{softmax}_l(\boldsymbol{x}_s^\top \boldsymbol{W} \boldsymbol{x}_l) = e^{\boldsymbol{x}_s^\top \boldsymbol{W} \boldsymbol{x}_l} / \sum_{j=1}^{P} e^{\boldsymbol{x}_j^\top \boldsymbol{W} \boldsymbol{x}_l}$. $\text{Relu}(\boldsymbol{z}) = \max\{0, \boldsymbol{z}\}$.

Denote $\mathcal{T}_k$, $k \leq K$ as the k-th task function that we aim to learn with the training dataset $\{\boldsymbol{X}^n, y^n\}_{n=1}^{N}$. Starting from the initialization $\boldsymbol{\theta}_{pre}$, we train the model using stochastic gradient descent with Hinge loss to obtain the fine-tuned model $\boldsymbol{\theta}_i$. The task vector is then computed as $\boldsymbol{\tau}_k = \boldsymbol{\theta}_k - \boldsymbol{\theta}_{pre}$.

The data formulation follows Definition 2 of Li et al. (2025), where the label of each data for task $\mathcal{T}_k$ is determined by the majority between tokens with two discriminative features $\boldsymbol{\mu}_{\mathcal{T}_k}$ and $-\boldsymbol{\mu}_{\mathcal{T}_k}$. If the number of tokens equal to $\boldsymbol{\mu}_{\mathcal{T}_k}$ (or $-\boldsymbol{\mu}_{\mathcal{T}_k}$) is large than that of $-\boldsymbol{\mu}_{\mathcal{T}_k}$ (or $\boldsymbol{\mu}_{\mathcal{T}_k}$), then $y = +1$ (or $y = -1$). Each data can also contain task-irrelevant tokens from an orthonormal set $\{\boldsymbol{v}_1, \boldsymbol{v}_2, \cdots, \boldsymbol{v}_M\}$ that are orthogonal to discriminative features. Given $\boldsymbol{\tau}_k$, $k \in [K]$, trained with a batch size of $B \geq \Omega(\epsilon^{-2} \log M)$ to achieve an ϵ generalization error on $\mathcal{T}_k$, denote $\boldsymbol{u}_k^i$ as the i-th row of the part of $\boldsymbol{V}$ in $\boldsymbol{\tau}_k$. Then, we can prove Proposition 1 as follows.

Proof. By Corollary 2 and Lemma 5 of Li et al. (2025), we know that with a constant probability, which we denote by p, $\boldsymbol{u}_k^i$, $k \in [K]$, $i \in [m]$, learns the discriminative feature. Then, with a high probability of $1 - p^K$, there exists $k_1 \neq k_2 \in [K]$, such that the i-th neuron of $\boldsymbol{\tau}_{k_1}$ learns discriminative features, while the i-th neuron of $\boldsymbol{\tau}_{k_2}$ fails to learn discriminative features.

By their Definition 4 of Li et al. (2025), we have that for any $i\mathcal{S}$,

$$\|\boldsymbol{u}_k^i\| \geq \Omega(m^{-1/2}). \tag{9}$$

By Lemma 5 in Li et al. (2025), we know that neurons of $\boldsymbol{\tau}_{k_1}$ in $\mathcal{S}$ are mainly in the direction of $\boldsymbol{\mu}_{\mathcal{T}_k}$ or $-\boldsymbol{\mu}_{\mathcal{T}_k}$, with the norm of all other directions smaller than a $1/\sqrt{M}$ scaling of that in the direction of the discriminative pattern. Then, the variance of lucky neurons in different task vectors can be computed as

$$\begin{aligned} \mathrm{DV}(g(\boldsymbol{\tau}_{k_1}, \mathcal{S}); \{\boldsymbol{\tau}_j\}_{j=1}^K) =& \mathbb{E}\Big[\Big\|g(\boldsymbol{\tau}_{k_1}, \mathcal{S}) - \frac{1}{K}\sum_{j=1}^K g(\boldsymbol{\tau}_j, \mathcal{S})\Big\|\Big] \\ \geq& |\mathcal{S}| \cdot \Omega(m^{-1}). \end{aligned} \tag{10}$$

Hence, the diversity of neurons that learn discriminative patterns is in the order of $|\mathcal{S}| \cdot \Omega(m^{-1/2})$.

For neurons that learn no discriminative features, $k \in [K]$, $i \in [m]$, we have

$$\begin{aligned} \mathrm{DV}(g(\boldsymbol{\tau}_{k_2}, \mathcal{S}); \{\boldsymbol{\tau}_j\}_{j=1}^K) =& \mathbb{E}\Big[\Big\|g(\boldsymbol{\tau}_{k_2}, \mathcal{S}) - \frac{1}{K}\sum_{j=1}^K g(\boldsymbol{\tau}_j, \mathcal{S})\Big\|\Big] \\ \leq& |\mathcal{S}| \cdot O(m^{-1}) \cdot \frac{1}{\sqrt{K}}. \end{aligned} \tag{11}$$

The last step comes from that (i) $\|g(\boldsymbol{\tau}_{k_2}, \mathcal{S})\| \leq O(m^{-1/2})\sqrt{\log B/B}$. (ii)

$$\Big\|\frac{1}{K}\sum_{j=1}^K g(\boldsymbol{\tau}_j, \mathcal{S})\Big\| \leq O(m^{-1/2}) \cdot \frac{1}{\sqrt{K}}. \tag{12}$$

Then, by combining (10) and (12), we have

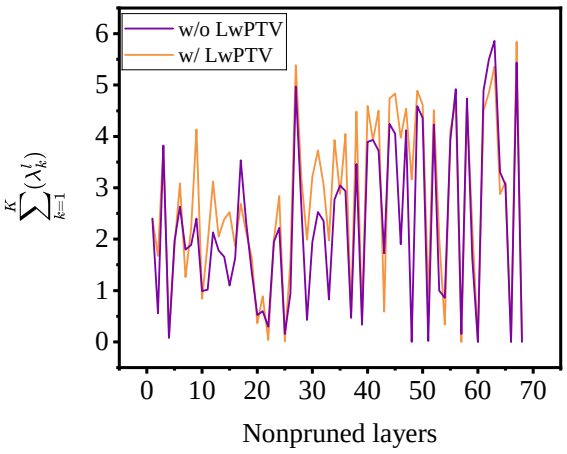
$$\mathrm{DV}(g(\boldsymbol{\tau}_{k_1}, \mathcal{S}); \{\boldsymbol{\tau}_j\}_{j=1}^K) \geq \sqrt{K} \cdot \Omega(\mathrm{DV}(g(\boldsymbol{\tau}_{k_2}, \mathcal{S}); \{\boldsymbol{\tau}_j\}_{j=1}^K)). \tag{13}$$

□

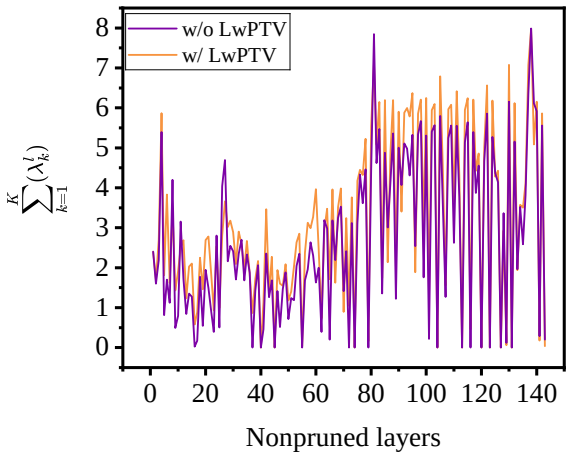
B Experimental details

B.1 The analysis of scaling

To investigate the impact of LwPTV (w/o scale) on the layer-wise coefficients λ_k^l of LW AdaMerging, we have plotted a comparative figure of the unpruned layer coefficients under two scenarios: LW AdaMerging and LW AdaMerging w/ LwPTV. The results are illustrated in Fig.8a and Fig.8b. In these figures, $\sum_{k=1}^K(\lambda_k^l)$ represents the sum of the coefficients of K tasks on the l-th layer. It can be clearly observed from these



(a) Lw AdaMerging w/ LwPTV *vs.* Lw AdaMerging on ViT-B/32.



(b) Lw AdaMerging w/ LwPTV *vs.* Lw AdaMerging on ViT-L/14.

Figure 8: The variations in λ_k^l of unpruned layers subsequent to the application of the LW AdaMerging and LW AdaMerging w/ LwPTV.

two figures that the λ_k^l obtained after pruning and optimization are significantly increased. This is because, during the process of optimizing λ_k^l by entropy minimization, the adjustable coefficients λ_k^l are amplified to enhance the influence of unpruned parameters in order to minimize the loss function as much as possible. This allows the merged model to maintain its ID performance even as the pruning ratio continuously increases, but its OOD performance does not show a significant improvement. In order to improve the OOD performance, we further use $\hat{\lambda}_k^l = \eta \cdot \lambda_k^l$ to scale the coefficient after optimizing the equation 8.

B.2 Algorithm

For Task Arithmetic, Ties, TW AdaMerging, TW AdaMerging++, LW AdaMerging, LW AdaMerging++, PCB-MERGING and FR-Merging, we present the algorithmic workflows of their integration with LwPTV in Algorithm 1.

B.3 Dataset details

Following Yadav et al. (Yadav et al., 2023; Zhou et al., 2022), we utilize eight datasets, including Stanford Cars (Krause et al., 2013), DTD (Cimpoi et al., 2014), EuroSAT (Helber et al., 2019), GTSRB (Stallkamp et al., 2011), MNIST (Deng, 2012), RESISC45 (Cheng et al., 2017), SUN397 (Xiao et al., 2016), and SVHN (Netzer et al., 2011), for merging vision models. These datasets are considered ID datasets. Following Wang et al. (Wang et al., 2024b), we select CIFAR100 (Krizhevsky et al., 2009), STL10 (Coates et al., 2011), Flowers102 (Nilsback & Zisserman, 2008), OxfordIIITPet (Parkhi et al., 2012), PCAM (S.Veeling et al., 2018), FER2013 (Goodfellow et al., 2013), EMNIST (Cohen et al., 2017), CIFAR10 (Krizhevsky et al., 2009), Food101 (Bossard et al., 2014), FashionMNIST (Xiao et al., 2017), RenderedSST2 (Socher et al., 2013; Radford et al., 2019), KMNIST (Clanuwat et al., 2018) and ImageNet-1k (Deng et al., 2009) as OOD datasets. The details of these datasets are shown in Tab.4

B.4 Evaluation metrics

To rigorously assess the efficacy of our method in enhancing the generalization capability of model merging methods on OOD datasets, while concurrently preserving their efficacy on ID datasets, we utilize a key evaluation metrics: H-score. The H-score is defined by the harmonic mean of the mean performance for the ID datasets, denoted as $Avg(P_{ID})$, coupled with the mean performance for the OOD datasets, denoted as $Avg(P_{OOD})$. It can be formulated as follows:

Algorithm 1 Overall algorithm when combining our LwPTV with existing model merging methods.

Input: pre-trained model parameters $\boldsymbol{\theta}_{pre}$, K task-specific fine-tuned model parameters $\{\boldsymbol{\theta}_k\}_{k=1}^K$ and hyperparameter η;
for $k = 1$ **to** K **do**
    Compute the task vector $\boldsymbol{\tau}_k$:
        $\boldsymbol{\tau}_k = \boldsymbol{\theta}_k - \boldsymbol{\theta}_{pre}$;
    Compute the salience score $\mathbf{s}_k$ of $\boldsymbol{\tau}_k$ by equation 5;
    Compute the mask $\mathbf{m}_k$ by equation 6;
end for
Calculate the final mask $\hat{\mathbf{m}}$ by equation 7.
Model merging methods execute:
If Task Arithmetic: $\hat{\boldsymbol{\theta}}_m = \boldsymbol{\theta}_{pre} + \hat{\mathbf{m}} \odot \lambda \sum_{k=1}^K \boldsymbol{\tau}_k$.
If Ties: Perform trim and sign election operations on $\{\hat{\mathbf{m}} \odot \boldsymbol{\tau}_k\}_{k=1}^K$ to obtain $\{\hat{\boldsymbol{\tau}}_k\}_{k=1}^K$, then perform $\hat{\boldsymbol{\theta}}_m = \boldsymbol{\theta}_{pre} + \lambda \sum_{k=1}^K \hat{\boldsymbol{\tau}}_k$.
If TW AdaMerging: Get $\{\lambda_k\}_{k=1}^K$ by minimizing entropy based on equation 8 , then perform equation 8.
If TW AdaMerging++: Perform trim and sign election operations on $\{\hat{\mathbf{m}} \odot \boldsymbol{\tau}_k\}_{k=1}^K$ to obtain $\{\hat{\boldsymbol{\tau}}_k\}_{k=1}^K$, then get $\{\lambda_k\}_{k=1}^K$ by minimizing entropy based on $\boldsymbol{\theta}_{pre} + \sum_{k=1}^K \lambda_k \hat{\boldsymbol{\tau}}_k$, then perform $\hat{\boldsymbol{\theta}}_m = \boldsymbol{\theta}_{pre} + \sum_{k=1}^K \lambda_k \hat{\boldsymbol{\tau}}_k$.
If LW AdaMerging: Get $\{\{\lambda_k^l\}_{l=1}^L\}_{k=1}^K$ by minimizing entropy based on equation 8, then perform $\hat{\boldsymbol{\theta}}_m = \boldsymbol{\theta}_{pre} + \hat{\mathbf{m}} \odot \{\sum_{k=1}^K \eta \lambda_k^l \boldsymbol{\tau}_k\}_{l=1}^L$.
If LW AdaMerging++: Perform trim and sign election operations on $\{\hat{\mathbf{m}} \odot \boldsymbol{\tau}_k\}_{k=1}^K$ to obtain $\{\hat{\boldsymbol{\tau}}_k\}_{k=1}^K$, and then get $\{\{\lambda_k^l\}_{l=1}^L\}_{k=1}^K$ by minimizing entropy based on $\boldsymbol{\theta}_{pre} + \{\sum_{k=1}^K \lambda_k^l \hat{\boldsymbol{\tau}}_k\}_{l=1}^L$, then perform $\hat{\boldsymbol{\theta}}_m = \boldsymbol{\theta}_{pre} + \{\sum_{k=1}^K \eta \lambda_k^l \hat{\boldsymbol{\tau}}_k\}_{l=1}^L$.
If PCB: Based on $\{\hat{\mathbf{m}} \odot \boldsymbol{\tau}_k\}_{k=1}^K$ to get importance score $\{\hat{\boldsymbol{\beta}}_k\}_{k=1}^K$, then perform $\hat{\boldsymbol{\theta}}_m = \boldsymbol{\theta}_{pre} + \hat{\mathbf{m}} \odot \sum_{k=1}^K (\hat{\boldsymbol{\beta}}_k \odot \lambda_k \boldsymbol{\tau}_k) / \sum_{k=1}^K \hat{\boldsymbol{\beta}}_i$.
If FR-Merging:Apply a Fourier filter to remove low-frequency components from $\{\hat{\mathbf{m}} \odot \boldsymbol{\tau}_k\}_{k=1}^K$, yielding $\{\tilde{\boldsymbol{\tau}}_k\}_{k=1}^K$. Then compute merging coefficients $\{\lambda_k \mathbb{E}(\tilde{\boldsymbol{\tau}}_k) \left(\sum_{k=1}^K \mathbb{E}(\tilde{\boldsymbol{\tau}}_k)\right)^{-1}\}_{k=1}^K$. Finally, perform $\hat{\boldsymbol{\theta}}_m = \boldsymbol{\theta}_{pre} + \sum_{k=1}^K \lambda_k \mathbb{E}(\tilde{\boldsymbol{\tau}}_k) \left(\sum_{k=1}^K \mathbb{E}(\tilde{\boldsymbol{\tau}}_k)\right)^{-1} \tilde{\boldsymbol{\tau}}_k$.

$$P_H = \frac{2 \times Avg(P_{ID}) \times Avg(P_{OOD})}{Avg(P_{ID}) + Avg(P_{OOD})}. \tag{14}$$

B.5 ID performance

We evaluate the performance of our proposed method alongside baseline approaches on ID datasets across three distinct architectures, namely ViT-B/32, ViT-L/14, and ViT-H/14. The corresponding results are systematically summarized in Tab.5 and Tab.6, respectively. The experimental results demonstrate that our method, when integrated with existing model merging methods, induces minimal performance degradation on ID datasets, and in some cases even enhances ID performance. This improvement can be attributed to the reduced parameter conflicts achieved through our task vector pruning strategy, which effectively mitigates interference between different task-specific parameters.

B.6 Performance comparison on ViT-H/14

Tab.7 demonstrates the comparative evaluation of our method against multiple baselines across ID and OOD datasets, implemented on the ViT-H/14 architecture. It can be seen that although the ViT-H/14 model is already very large, there is still a gap in the generalization performance between the pretrained model and the individual fine-tuned model on the OOD datasets. Therefore, our method still works. Experimental

Table 4: In Domain and Out of Domain datasets details

Feature(→) Dataset (↓)	Type	Sample Size	Test Set Sample Size	Number of Classes	Samples per Class	Class Balance
Stanford Cars	**ID**	16185	8041	196	1:1 split	Yes
DTD		5640	1880	47	40	Yes
EuroSAT		27000	2700	10	2000-3000	Yes
GTSRB		51839	12630	43	/	No
MNIST		70000	10000	10	1000	Yes
RESISC45		31500	6300	45	approximately 700	Yes
SUN397		108754	7940	397	at least 100 images per category	No
SVHN		60000	26032	10	/	No
CIFAR100	**OOD**	60000	10000	100	100	Yes
STL10		13000	8000	10	800	Yes
Flowers102		8189	6129	102	per class 40-250	No
OxfordIIITPet		7349	3669	37	approximately 100	Yes
PCAM		327680	40960	2	/	/
FER2013		35887	7178	7	/	No
EMNIST		280000	40000	10	4000	Yes
CIFAR10		60000	10000	10	1000	Yes
Food101		101000	25250	101	250	Yes
FashionMNIST		70000	10000	10	1000	Yes
RenderedSST2		9613	1821	2	/	/
KMNIST		70000	10000	10	1000	Yes
ImageNet1000		3200000	50000	1000	/	No

results reveal that our method yields significant performance gains over existing techniques, with H-score improvements of 1.4%, 0.9%, 1.3%, and 0.5% over Task Arithmetic, Ties, PCB, and FR-Merging respectively, when combined with these approaches.

B.7 More ablation study details

Tab.8 and Tab.9 provides a detailed summary of the experimental results from our comprehensive ablation studies performed on the ID dataset and OOD dataset.

B.8 Comparison with OOD methods for fine-tuned models

Tab.11 and Tab.12 provide a detailed summary of the experimental results comparing OOD methods for fine-tuned models on the ID dataset and OOD dataset. Following Wortsman et al. (Wortsman et al., 2022b), we set the mixing coefficient $\alpha = 0.5$ for WiSE-FT. For the Model Stock, we employed the fine-tuning code from Ilharco et al. (Ilharco et al., 2023) and set the random seed to 10 to obtain the second set of fine-tuned models. We set the search range for λ in the Task Arithmetic to $(0, 1]$ with a step size of 0.1 based on Ilharco et al. (Ilharco et al., 2023; Wang et al., 2024a).

Table 10: The comparison of layer-wise pruning and parameter-wise pruning on ViT-B/32.

Method (↓)		H-score	Time consumption
Baseline	Task Arithmetic	58.9	8.3
Ours	**w/ PwPTV**	61.7	135.4s
	w/ LwPTV	66.3	9.0s

B.9 Comparison with other pruning methods

Tab.11 and Tab.12 provide a detailed summary of the experimental results comparing other pruning methods on the ID dataset and OOD dataset. In which, for the DARE and random mask methods, we select the best results obtained from three runs with different random seeds.

B.10 The analysis of layer-wise pruning and parameter-wise pruning

To investigate the distinctions between layer-wise pruning (LwPTV), which computes significance scores layer by layer, and parameter-wise pruning (PwPTV), which computes significance scores for individual parameters, under identical experimental setups, we conducted a comparative analysis, with a focus on performance and time consumption, based on the Task Arithmetic method on the ViT-B/32 architecture. The results are delineated in Tab.10. The detailed results of the two methods on the ID and OOD datasets are presented in Tab.13 and Tab.14, respectively. From this, it can be observed that LwPTV achieves better performance with less time consumption. This performance improvement can be attributed to LwPTV,

Table 5: Performance comparison for ID datasets on ViT-B/32 and ViT-L/14.

	Method (↓)	SUN397	Cars	RESISC45	EuroSAT	SVHN	GTSRB	MNIST	DTD	Avg.
ViT-B/32	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
	Weight Averaging (Wortsman et al., 2022a)	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
	AdaMerging w/ Surgery (Yang et al., 2024a)	69.8	71.0	88.9	98.1	91.7	96.5	98.8	73.6	86.1
	Task Arithmetic (Ilharco et al., 2023)	55.2	54.9	66.7	78.9	**80.2**	**69.7**	**97.3**	50.4	69.1
	w/ LwPTV (Ours)	**66.8**	**66.0**	**77.2**	**79.1**	78.3	68.7	92.3	**54.4**	**72.8**$^{(+3.7)}$
	Ties (Yadav et al., 2023)	65.0	64.4	74.8	77.4	**81.2**	**69.3**	**96.5**	54.5	**72.9**
	w/ LwPTV (Ours)	**67.6**	**66.7**	**77.4**	**77.4**	77.7	66.8	92.4	**55.3**	72.7$^{(-0.2)}$
	TW AdaMerging (Yang et al., 2024b)	58.0	53.2	68.8	85.7	81.1	84.4	92.4	44.8	71.1
	w/ LwPTV (Ours)	**61.6**	**59.7**	**77.0**	**87.3**	**87.0**	**84.9**	**95.5**	**50.6**	**75.4**$^{(+4.3)}$
	TW AdaMerging++ (Yang et al., 2024b)	60.8	56.9	73.1	83.4	87.3	**82.4**	95.7	50.1	73.7
	w/ LwPTV (Ours)	**63.0**	**63.2**	**78.5**	**84.9**	**88.4**	79.1	**96.7**	**55.1**	**76.1**$^{(+2.4)}$
	LW AdaMerging (Yang et al., 2024b)	64.5	68.1	79.2	**93.8**	**87.0**	**91.9**	**97.5**	**59.1**	**80.1**
	w/ LwPTV (Ours)	**68.0**	**68.5**	**80.7**	88.3	80.0	81.2	94.1	57.7	77.3$^{(-2.8)}$
	LW AdaMerging++ (Yang et al., 2024b)	66.6	68.3	**82.2**	**94.2**	**89.6**	**89.0**	**98.3**	**60.6**	**81.1**
	w/ LwPTV (Ours)	**68.9**	**69.0**	81.0	85.8	82.2	76.7	95.0	58.2	77.1$^{(-4.0)}$
	PCB-MERGING (Du et al., 2024)	63.8	62.0	77.1	80.6	**87.5**	**78.5**	98.7	58.4	75.8
	w/ LwPTV (Ours)	**68.6**	**67.7**	**80.7**	**83.0**	85.4	77.2	**96.2**	**60.0**	**77.3**$^{(+1.5)}$
	FR-Merging (Zheng & Wang, 2025)	66.2	64.5	77.2	**90.1**	**85.4**	**82.3**	**98.5**	**60.0**	**78.1**
	w/ LwPTV (Ours)	**67.2**	**67.8**	**77.9**	81.9	75.5	74.9	93.8	58.4	74.7$^{(-3.4)}$
ViT-L/14	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
	Weight Averaging (Wortsman et al., 2022a)	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
	AdaMerging w/ Surgery (Yang et al., 2024a)	80.3	90.8	94.3	98.2	94.1	98.7	99.2	82.5	92.3
	Task Arithmetic (Ilharco et al., 2023)	73.9	82.1	86.6	94.1	**87.9**	**86.7**	**98.9**	65.6	84.5
	w/ LwPTV (Ours)	**74.7**	**86.2**	**89.3**	**94.6**	86.7	86.1	98.8	**68.6**	**85.6**$^{(+1.1)}$
	Ties (Yadav et al., 2023)	**76.5**	85.0	89.3	**95.7**	**90.3**	83.3	99.0	68.8	86.0
	w/ LwPTV (Ours)	76.3	**87.4**	**90.9**	95.1	89.1	**87.3**	**99.0**	**70.5**	**86.9**$^{(+0.9)}$
	TW AdaMerging (Yang et al., 2024b)	75.6	83.5	79.7	90.3	83.5	96.5	98.0	67.3	84.3
	w/ LwPTV (Ours)	**76.7**	**88.5**	**87.7**	**96.3**	**90.7**	**97.6**	**98.7**	**75.6**	**89.0**$^{(+4.7)}$
	TW AdaMerging++ (Yang et al., 2024b)	76.6	86.2	85.6	94.6	89.7	**96.8**	98.2	72.1	87.5
	w/ LwPTV (Ours)	**78.0**	**89.3**	**89.3**	**96.4**	**91.8**	96.4	**98.8**	**77.2**	**89.7**$^{(+2.2)}$
	LW AdaMerging (Yang et al., 2024b)	79.0	90.3	90.8	**96.2**	**93.4**	**98.0**	**99.0**	**79.9**	**90.8**
	w/ LwPTV (Ours)	**79.0**	**90.3**	**91.6**	95.7	89.3	96.4	98.8	77.6	89.8$^{(-1.0)}$
	LW AdaMerging++ (Yang et al., 2024b)	**79.4**	**90.3**	91.6	**97.4**	**93.4**	**97.5**	**99.0**	**79.2**	**91.0**
	w/ LwPTV (Ours)	79.2	90.1	**92.4**	96.4	89.1	94.9	98.8	76.6	89.7$^{(-1.3)}$
	PCB-MERGING (Du et al., 2024)	76.2	86.0	89.6	**95.9**	**89.9**	**92.3**	**99.2**	71.4	87.6
	w/ LwPTV (Ours)	**76.8**	**88.2**	**91.1**	95.8	89.2	92.1	99.1	**72.7**	**88.1**$^{(+0.5)}$
	FR-Merging (Zheng & Wang, 2025)	**76.4**	87.0	**90.2**	**96.8**	**92.0**	92.8	**99.3**	**71.5**	**88.3**
	w/ LwPTV (Ours)	74.0	**87.5**	90.0	96.5	88.9	**95.2**	99.2	71.0	87.8$^{(-0.5)}$

Table 6: Performance comparison for ID datasets on ViT-H/14.

Method (↓)	SUN397	Cars	RESISC45	EuroSAT	SVHN	GTSRB	MNIST	DTD	Avg.
Pretrained	73.8	93.4	75.7	73.1	52.5	58.4	72.8	67.8	70.9
Individual	81.7	95.1	97.5	99.3	98.1	99.3	99.7	83.8	94.3
Weight Averaging (Wortsman et al., 2022a)	76.2	94.5	85.8	92.7	84.6	79.8	97.5	70.1	85.2
Task Arithmetic (Ilharco et al., 2023)	77.0	94.5	89.6	94.7	91.3	**91.0**	**99.3**	69.6	88.4
w/ LwPTV (Ours)	**78.0**	**95.2**	**92.0**	**96.4**	**92.1**	90.4	99.1	**71.8**	**89.4**$^{(+1.0)}$
Ties (Yadav et al., 2023)	77.7	94.8	90.1	94.3	91.2	86.8	99.0	70.1	88.0
w/ LwPTV (Ours)	**78.9**	**95.2**	**92.6**	**96.3**	**93.8**	**91.2**	**99.3**	**71.3**	**89.8**$^{(+1.8)}$
PCB-MERGING (Du et al., 2024)	77.8	94.6	90.6	96.2	93.5	95.1	**99.4**	76.4	90.5
w/ LwPTV (Ours)	**78.9**	**95.3**	**92.8**	**96.7**	**94.0**	**95.2**	99.3	**78.7**	**91.4**$^{(+1.8)}$
FR-Merging (Zheng & Wang, 2025)	95.3	82.7	**97.5**	**97.7**	**99.5**	91.3	**92.5**	**77.2**	**91.7**
w/ LwPTV (Ours)	**95.4**	**82.8**	97.4	96.2	99.4	**91.4**	91.1	77.0	91.3 $^{(-0.4)}$

Table 7: Performance comparison for ID and OOD datasets on ViT-H/14.

Dataset (→)	In Domain	Out of Domain													H-score
Method (↓)	Avg.	C10	C100	EMN	FMN	FER	F102	F101	KMN	OxP	PCA	RSS	STL	Ima	Avg.	
Pretrained	70.9	97.4	84.7	14.1	79.0	36.2	80.1	92.2	11.7	94.5	51.0	64.0	98.5	77.9	67.8	69.3
Individual	94.3	91.7	72.6	18.7	74.2	37.4	77.7	88.8	10.1	93.4	52.1	63.9	97.4	75.0	65.6	77.4
Weight Averaging (Wortsman et al., 2022a)	85.2	97.1	83.0	21.5	79.7	38.2	79.7	91.5	9.9	94.3	51.3	63.9	98.6	77.7	68.2	75.8
Task Arithmetic (Ilharco et al., 2023)	88.4	94.4	74.8	**28.9**	78.7	38.1	77.9	88.0	9.4	93.3	**52.3**	62.9	97.6	75.0	67.0	76.2
w/ LwPTV (Ours)	**89.4**	**96.4**	**81.4**	28.6	**79.9**	**39.0**	**79.3**	**91.6**	**9.7**	**94.3**	51.7	**64.1**	**98.5**	**77.7**	**68.6** (+1.6)	**77.6** (+1.4)
Ties (Yadav et al., 2023)	88.9	95.0	76.4	27.0	78.8	38.4	78.6	89.2	9.2	93.8	**51.9**	64.1	98.0	76.0	67.4	76.7
w/ LwPTV (Ours)	**89.8**	**96.0**	**79.9**	**27.7**	**79.5**	**38.8**	**79.1**	**91.4**	**9.5**	**94.3**	51.7	**64.1**	**98.5**	**77.5**	**68.3** (+0.9)	**77.6** (+0.9)
PCB-MERGING (Du et al., 2024)	90.5	93.5	72.7	27.8	78.8	37.3	76.8	87.9	9.4	93.3	**52.4**	63.3	97.7	75.0	66.6	76.7
w/ LwPTV (Ours)	**91.4**	**95.3**	**78.0**	**28.2**	**79.3**	**38.6**	**78.6**	**91.1**	**9.4**	**94.3**	51.9	**64.1**	**98.4**	**77.3**	**68.0** (+1.4)	**78.0** (+1.3)
FR-Merging (Zheng & Wang, 2025)	**91.7**	94.5	75.7	**29.9**	78.7	38.3	77.4	90.4	**9.7**	93.7	**52.8**	64.1	98.2	76.5	67.7	77.9
w/ LwPTV (Ours)	91.3	**96.4**	**81.0**	29.8	**80.0**	**38.8**	**78.8**	**91.6**	9.6	**94.4**	52.7	**64.3**	**98.6**	**77.7**	**68.7** (+1.0)	**78.4** (+0.5)

Table 8: The ablation study of LwPTV on the ViT-B/32 regarding the ID dataset.

Method	Components			Performance (%)								
	Salience Score	∨	Scale	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
	✓	✗	-	68.1	65.7	76.0	77.4	69.2	61.2	87.1	54.4	69.9
	✗	✓	-	64.3	62.1	65.0	57.9	35.9	36.6	47.6	45.9	51.9
	✓	✓	-	66.8	66.0	77.2	79.1	78.3	68.6	92.3	54.4	72.8
LW AdaMerging (Yang et al., 2024b)	✗	✗	✗	64.5	68.1	79.2	93.8	87.0	91.9	97.5	59.1	80.1
	✓	✓	✗	66.0	67.7	82.0	91.4	87.5	87.7	97.3	59.0	79.8
	✓	✗	✓	68.6	66.6	79.8	83.7	79.1	75.4	91.4	57.3	75.2
	✗	✓	✓	64.5	62.5	67.2	62.4	38.3	39.7	47.8	46.7	53.6
	✓	✓	✓	68.0	68.5	80.7	88.3	80.0	81.2	94.1	57.7	77.3

which aligns the representations of OOD data generated by the merged model more closely with those produced by the pretrained model, thereby significantly enhancing OOD performance.

B.11 Neural network component analysis

To investigate whether our method primarily prunes task vectors specific to certain structures, such as parameters in MLP, whose pre-trained parameters inherently have stronger generalization capabilities, leading to the effectiveness of LwPTV. We conducted a more in-depth analysis of ours by categorizing the layers of ViT into three types: **Attention Layer**, **MLP Layer**, and **Other Layer** (including position embedding, layer normalization, etc.). We then analyzed their salience scores of the task vectors for SUN397. As shown in Fig.10 the salience scores do not reflect this trend, suggesting that ours does not selectively filter out specific types of layers. Therefore, the effectiveness of ours lies in pruning the redundant parameters, instead of pruning certain layer types.

B.12 LwPTV aligns the representations between the merged model and the pretrained model

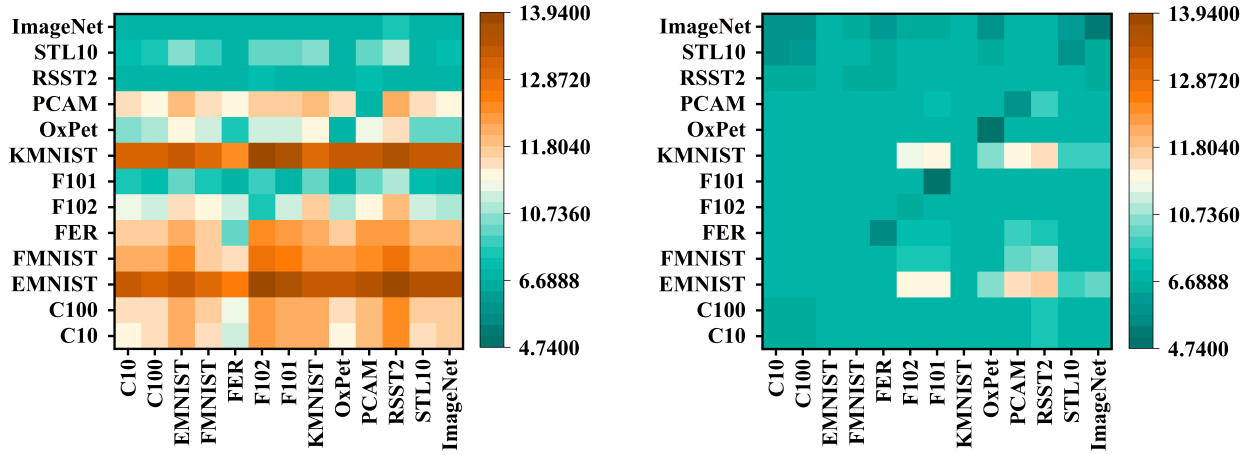
To investigate how our method LwPTV influences the merged model in generating representations for OOD data, we compute two sets of ℓ_2 distances: (1) Task Arithmetic *vs.* Pretrained model (2) Task Arithmetic w/ LwPTV *vs.* Pretrained model. The results are shown in Fig.9. It can be seen that LwPTV aligns the representations between the merged model and the pretrained model. This observation is consistent with our goal and further supports the notion that our method improves OOD performance.

Table 9: The ablation study of LwPTV on the ViT-B/32 regarding the OOD dataset.

Method	Components			Performance (%)												
	Salience Score	∨	Scale	C10	C100	EMN	FMN	FER	F102	F101	KMN	OxP	PCA	RSS	STL	Ima	Avg.
Task Arithmetic (Ilharco et al., 2023)	✗	✗	-	76.3	41.9	28.5	63.9	26.3	49.4	55.8	9.0	75.4	54.0	53.4	87.8	45.0	51.3
	✓	✗	-	89.4	62.6	28.5	66.1	38.6	64.8	81.2	7.2	86.5	59.6	59.2	96.9	62.2	61.8
	✗	✓	-	90.3	64.4	17.9	63.8	39.4	66.5	82.7	9.4	86.9	59.0	57.0	97.1	62.7	61.3
	✓	✓	-	88.3	60.1	30.4	65.9	36.5	62.9	79.9	7.4	85.4	59.5	59.0	96.4	60.6	60.9
LW AdaMerging (Yang et al., 2024b)	✗	✗	✗	82.1	50.7	28.7	63.2	37.8	58.3	73.8	8.8	82.4	57.3	58.0	94.3	54.6	57.7
	✓	✓	✗	84.1	52.0	30.1	64.1	37.1	59.6	76.5	8.3	83.3	57.6	58.1	95.4	56.4	58.7
	✓	✗	✓	88.4	60.7	30.6	65.8	38.7	63.5	80.4	7.2	86.0	59.4	58.8	96.7	61.3	61.3
	✗	✓	✓	90.4	64.5	19.5	64.5	39.0	65.1	82.0	9.2	85.5	57.4	56.0	97.0	61.3	60.9
	✓	✓	✓	88.8	61.3	30.6	66.1	38.4	64.3	80.6	7.6	86.3	60.1	59.4	96.6	61.2	61.7

Table 11: Comparison with other OOD methods (above) and purning methods (below) on ViT-B/32 on the ID dataset.

	Dataset (→) Method (↓)	**SUN397**	**Cars**	**RESISC4**	**EuroSAT**	**SVHN**	**GTSRB**	**MNIST**	**DTD**	**Avg.**
Baseline	Task Arithmetic (Ilharco et al., 2023)	55.2	54.9	66.7	78.9	80.2	69.7	97.3	50.4	69.1
OOD	w/ WiSE-FT (Wortsman et al., 2022b)	60.6	59.0	70.2	79.4	78.2	68.6	96.1	51.5	70.5
	Model Stock (Jang et al., 2024)	60.7	65.0	75.0	81.7	80.0	68.5	95.2	49.6	72.0
	w/ LiNeS (Wang et al., 2024a)	63.9	63.9	75.1	85.6	79.4	72.2	96.2	56.5	74.1
Pruning	w/ DARE (Yu et al., 2024)	64.6	58.7	60.5	47.4	32.1	33.6	51.0	44.8	49.1
	w/ MWP (Sanh et al., 2020)	62.6	61.6	72.1	78.1	80.0	69.9	96.6	53.0	71.7
	w/ random mask	66.2	59.9	61.3	48.6	32.1	32.3	48.2	44.5	49.1
	w/ absolute value	64.6	61.0	60.7	47.1	32.0	32.8	48.4	44.4	48.9
Ours	**w/ LwPT**	66.8	66.0	77.2	79.1	78.3	68.6	92.3	54.4	72.8



(a) Task Arithmetic *vs.* Pretrained model

(b) Task Arithmetic w/ LwPTV *vs.* Pretrained model

Figure 9: Visualization of ℓ_2 distance between merged model representations and pre-trained model representations.

B.13 Additional analysis on out-of-domain (OOD) datasets

To further clarify the behavior of our method on out-of-domain (OOD) tasks, we include here a detailed supplementary analysis. Specifically, we investigate how the effectiveness of LwPTV varies with the degree of distributional shift between in-domain (ID) and OOD datasets. This experiment complements Sec.5.1 in the main paper and aims to demonstrate that our method primarily benefits truly challenging OOD scenarios rather than those that are superficially similar to the ID data.

Table 15: Performance and distance comparison on ViT-B/32.

Groups (↓)	**Method (↓)**	OOD Acc. (%)	**Distance**
Demanding OOD	Task Arithmetic	51.3	6.3
	w / LwPTV (Ours)	60.9	
Easy OOD	Task Arithmetic	61.7	3.5
	w / LwPTV (Ours)	60.9	

Table 12: Comparison with other OOD methods (above) and purning methods (below) on ViT-B/32 on the OOD dataset.

	Dataset (→) Method (↓)	C10	C100	EMN	FMN	FER	F102	F101	KMN	OxP	PCA	RSS	STL	Ima	Avg.
Baseline	Task Arithmetic (Ilharco et al., 2023)	76.3	41.9	28.5	63.9	26.3	49.4	55.8	9.0	75.4	54.0	53.4	87.8	45.0	51.3
OOD	w/ WiSE-FT (Wortsman et al., 2022b)	81.6	49.9	29.6	64.5	30.3	55.2	65.9	8.4	81.1	56.7	54.2	91.4	52.0	55.4
	Model Stock (Jang et al., 2024)	86.6	57.7	30.9	65.1	32.1	60.8	74.9	8.7	84.2	57.0	54.4	94.9	56.5	58.7
	w/ LiNeS (Wang et al., 2024a)	84.4	54.9	30.3	65.4	33.6	60.4	73.7	8.4	83.6	58.8	51.4	93.9	55.6	58.0
Pruning	w/ DARE (Yu et al., 2024)	88.7	64.7	20.8	64.1	38.0	65.9	82.3	8.8	87.0	61.7	58.2	96.9	63.1	61.5
	w/ MWP (Sanh et al., 2020)	83.7	52.1	27.8	64.4	32.0	56.1	69.2	8.2	81.3	57.5	54.2	92.9	53.6	56.4
	w/ random mask	89.6	64.9	18.1	63.9	38.8	66.5	82.9	9.4	87.7	60.2	59.2	97.0	63.6	61.7
	w/ absolute value	89.9	64.6	17.5	63.7	38.8	66.6	82.9	9.4	87.4	60.2	57.9	97.2	63.3	61.5
Ours	**w/ LwPT**	88.3	60.1	30.4	65.9	36.5	62.9	79.9	7.4	85.4	59.5	59.0	96.4	60.6	60.9

Table 13: The comparison of layer-wise pruning and parameter-wise pruning on ID datasets.

	Dataset (→) Method (↓)	SUN397	Cars	RESISC4	EuroSAT	SVHN	GTSRB	MNIST	DTD	Avg.
Baseline	Task Arithmetic (Ilharco et al., 2023)	55.2	54.9	66.7	78.9	80.2	69.7	97.3	50.4	69.1
Ours	w/ PwPTV	59.8	59.0	70.6	79.0	81.9	72.2	97.2	52.3	71.5
	w/ LwPTV	66.8	66.0	77.2	79.1	78.3	68.6	92.3	54.4	72.8

Experiment setup: We created two test groups. One is demanding OOD, i.e., our main benchmark of 13 OOD datasets and 8 ID tasks used in the paper. Another is easy OOD, a control group where we used two of the ID datasets (MNIST and EuroSAT) as the OOD tasks and set the remaining six as ID tasks. To prove the "easy" group was indeed more similar, we measured the Mahalanobis distance between the ID and OOD datasets, resulting in an 8×13 distance matrix for demanding OOD and a 6×2 matrix for easy OOD in the control group. As shown in Tab.15, the averaged distance for the "easy" group was much lower (3.5) than for our main demanding benchmark (6.3), confirming it was less of a challenge. **Results and insight:** We evaluate Task Arithmetic and Task Arithmetic + LwPTV under both settings. As shown in Tab. 15, LwPTV yields a large improvement under the demanding OOD setting but no gain under the Easy OOD setting. This outcome indicates that LwPTV is not a generic booster but a principled method that improves performance on challenging, truly out-of-distribution scenarios by preserving the generalizable features of the pre-trained model.

B.14 Analysis of layers with high mean magnitude but low variance

To further examine the validity of our salience score and explore the role of layers with high mean absolute values but low variance, we conducted an additional analysis on the ViT-B/32 model. First, in the ViT-B/32 model, we identified layers that exhibit high mean absolute values and low cross-task variance, and experimentally verified that removing these layers does not harm performance. Specifically, we found 17 such layers, all of which are pruned by our proposed method, LwPTV. Among them, 16 are bias layers and the remaining one is the final layer normalization weight (model.ln_final.weight). We then evaluated two variants of our pruning strategy: (V1) we keep all 17 of these layers unpruned, and (V2) we prune only the final layer normalization weight while keeping the other 16 bias layers. As shown in the Tab.16, neither variant improves performance over our method, suggesting that the salience score correctly identifies these layers as redundant and pruning them helps improve OOD generalization. Second, from the perspective of feature learning, we provide a simple theoretical justification for why pruning layers with high mean absolute values but low cross-task variance, which are mainly bias layers, is reasonable. Consider a linear layer $f(x) = Wx + b$. During gradient updates, the gradient with respect to W depends on the input x, so the corresponding component of the task vector, i.e.,

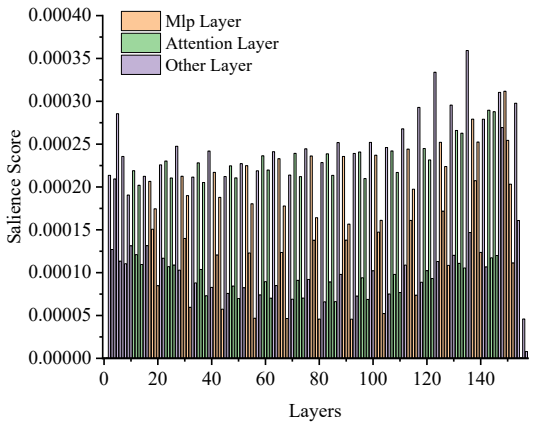


Figure 10: The salience score of the task vectors with different components on ViT-B/32.

Table 14: The comparison of layer-wise pruning and parameter-wise pruning on OOD datasets.

	Dataset (→) Method (↓)	C10	C100	EMN	FMN	FER	F102	F101	KMN	OxP	PCA	RSS	STL	Ima	Avg.
Baseline	Task Arithmetic (Ilharco et al., 2023)	76.3	41.9	28.5	63.9	26.3	49.4	55.8	9.0	75.4	54.0	53.4	87.8	45.0	51.3
Ours	w/ PwPTV	80.4	46.6	27.8	64.5	29.0	53.0	63.9	8.4	79.0	56.8	54.1	90.8	50.1	54.2
	w/ LwPTV	88.3	60.1	30.4	65.9	36.5	62.9	79.9	7.4	85.4	59.5	59.0	96.4	60.6	60.9

the accumulated gradient updates for W, contains task-specific information related to x. In contrast, the gradient with respect to b is independent of the input, and thus the corresponding component of the task vector lacks discriminative information across tasks. As a result, although the magnitude of the task vectors for the bias may be large, they lack diversity. According to Proposition 1, our diversity-based salience score is designed to identify and prune precisely such non-diverse, task-irrelevant layers, as its large magnitude is misleading in the context of finding task-specific information.

B.15 Effect of the number of in-domain tasks on LwPTV

To further assess the stability of our salience score and examine whether LwPTV depends strongly on the number of in-domain tasks K, we conduct an additional study on ViT-B/32 by varying K from 2 to 8. This analysis complements our main results (where $K = 8$) and evaluates whether the diversity-based salience estimation remains reliable in low-task settings. As shown in Tab.17, the relative H-score improvement when applying LwPTV to the Task Arithmetic baseline increases steadily as K grows. This trend aligns with Proposition 1: our salience score identifies task-specific features (high diversity) by measuring deviations from the cross-task mean, and larger K yields a more stable estimate of "common" adaptation. As a result, a greater number of tasks creates a clearer signal that leads to more precise pruning and larger OOD gains.

Table 16: Performance comparison on ViT-B/32.

Method (↓)	ID	OOD	H-score
Task Arithmetic	69.1	51.3	58.9
V1 (keep 17 layers)	72.9	61.0	66.4
V2 (keep 16 layers)	72.9	61.0	66.4
Ours	72.8	60.9	66.3

Importantly, LwPTV also remains effective under low-task scenarios. Even with only $K = 2$ tasks, we observe a substantial improvement of +2.0 in H-score over the baseline, indicating that the method is robust and beneficial even with a small number of tasks. Overall, these findings empirically support the theoretical analysis and confirm that LwPTV scales favorably with the number of tasks while retaining practical utility in small-K scenarios.

B.16 Effect of normalization on the salience score

To evaluate whether differences in parameter scale across layers may bias the salience score computation, we conduct a controlled experiment applying layer-wise normalization to task vectors before computing salience. Specifically, for each layer, we subtract the mean and divide by the standard deviation of its task-vector entries. This standardization enforces equal scale across layers and removes magnitude differences.

Table 17: Relative performance improvement as the number of ID tasks changes.

Task Num	2	3	4	5	6	7	8
H-score	+2.0	+3.3	+3.5	+3.8	+4.5	+6.0	+7.4

As shown in Tab.18, this normalization substantially degrades performance: the H-score drops from 66.3 to 59.1 on ViT-B/32. The reason for this is that for task vectors, the scale of parameters across layers is a meaningful signal that reflects task-specific adaptation. The original magnitude and variance of the parameters at different layers naturally encode their relative importance for the downstream tasks. Normalization erases this vital information by putting all layers on an equal footing. This can amplify noise in layers with small-magnitude changes and mislead the pruning process. Therefore, we compute salience score in the original parameter space to preserve the informative structural differences across layers.

B.17 Storage overhead

LwPTV can effectively diminish the storage requirements for task vectors for each merging method. As shown in Tab.19, we report the storage cost derived from various model merging techniques applied to the ViT-B/32 architecture. It can be seen that, LwPTV can significantly reduce the storage cost of the merged model in various methods. Since both Ties and PCB prune the parameters, applying ours to them further lead to smaller storage cost.

Table 18: Effect of layer-wise normalization on H-score (ViT-B/32).

Method	H-score
Task Arithmetic	58.9
w/ LwPTV (Ours)	66.3
w/ LwPTV + Normalization	59.1

Table 19: The storage cost of various model merging methods on ViT-B/32 and ViT-L/14.

Model Architecture	Method (→) w/LwPTV (↓)	Pretrained model	Task Arithmetic	Ties	TW AdaMerging	TW AdaMerging++	LW AdaMerging	LW AdaMerging++	PCB	FR-Merging
ViT-B/32	✗	432.77 MB	432.77 MB	285.25 MB	432.77 MB	285.25 MB	432.77 MB	285.25 MB	123.39 MB	432.77 MB
	✓	-	226.75 MB	220.03 MB	226.75 MB	220.03 MB	226.75 MB	220.03 MB	115.23 MB	226.75 MB
ViT-L/14	✗	1306.77 MB	1306.77 MB	965.18 MB	1306.77 MB	965.18 MB	1306.77 MB	965.18 MB	397.26 MB	1306.77 MB
	✓	-	836.35 MB	788.08 MB	836.35 MB	788.08 MB	836.35 MB	788.08 MB	377.21 MB	836.35 MB

B.18 Time complexity analysis

To investigate whether our method incurs significant computational overhead for layer-wise pruning on larger models, we conducted a time complexity analysis based on the ViT-H/14 architecture. Suppose the model has L layers, the l-th layer has d_l parameters, and K models need to be merged: (1) The time complexity of calculating $\mathbb{E}\left(\left|\boldsymbol{\tau}_k^l - \frac{1}{K}\sum_{k=1}^{K}\boldsymbol{\tau}_k^l\right|\right)$ is $\mathcal{O}(KLd_l)$. (2) When the l-th layer of the final mask $\hat{m}$ is 0, we directly set the l-th layer parameters of the merged task vector to 0. In the worst case, all parameters need to be set to 0. In this case, the time complexity is $\mathcal{O}(Ld_l)$. So the total time complexity is $\mathcal{O}(KLd_l)$. In addition, we list the time overhead of Task Arithmetic, Ties, PCB and FR-Merging with and without LwPTV on ViT-H/14 in Tab.20. The results show that whether we introduce ours into Task Arithmetic, Tie, PCB or FR-Merging, it takes no more than 67.6 seconds. The time introduced by ours is acceptable or can even be ignored, especially for the model merging algorithms, which are relatively time-consuming such as PCB.

Table 20: Time complexity on ViT-H/14.

Method (↓)	w/o LwPTV	w/ LwPTV
Task Arithmetic (Ilharco et al., 2023)	57.5s	74.9s
Ties (Yadav et al., 2023)	147.2s	214.8s
PCB (Du et al., 2024)	1861.0s	1928.6s
FR-Merging (Zheng & Wang, 2025)	692.64s	710.04s

B.19 Computational resources

Following Ilharco et al. (Ilharco et al., 2023), we fine-tune separate instances of the pretrained ViT-H/14 on eight distinct datasets: Cars, DTD, EuroSAT, GTSRB, MNIST, RESISC45, SUN397, and SVHN. Training employs the AdamW optimizer with a learning rate of 0.001 and a batch size of 128. To ensure convergence while mitigating overfitting, dataset-specific training epochs are assigned: 35 epochs for Cars, 76 for DTD, 12 for EuroSAT, 11 for GTSRB, 5 for MNIST, 15 for RESISC45, 14 for SUN397, and 4 for SVHN. All implementations use PyTorch on NVIDIA A800 GPUs.

B.20 NLP task

To evaluate the performance of our method on NLP tasks, we conducted experiments using T5-Large-LM-Adapt as the base model. We selected PAWS, QASC, QuaRTz, StoryCloze, WikiQA, Winogrande, and WSC as the ID datasets (Derek et al., 2024), and rte, cb, wic, copa, h-swag, anli-r1, anli-r2, and anli-r3 as the OOD datasets (Sanh et al., 2022). As shown in Tab.21, in contrast to image classification tasks, the merged model demonstrates significantly better OOD performance than the pretrained model in the field of NLP. This superiority can be attributed to the high degree of semantic space overlap across different tasks, domains, and fields in NLP (Dosovitskiy et al., 2020; Hou & Kung, 2022). This overlap implies that the ID task vectors contain information that is beneficial for OOD tasks (Arora et al., 2021; Lewis et al., 2021; Elangovan et al., 2021). In other words, the discriminative patterns of OOD tasks have a non-zero mapping onto the discriminative patterns of ID tasks (Li et al., 2025). Under these circumstances, enhancing the OOD performance of the merged model requires amplifying the role of task-specific parameters within the ID task, which can be achieved by pruning redundant parameters. Unlike images, which are spatial and have higher

Table 21: Performance comparison for ID and OOD datasets on T5-large.

Dataset (→)	In Domain	Out of Domain									H-score
Method (↓)	Avg.	rte	cb	wic	copa	h-swag	anli-r1	anli-r2	anli-r3	Avg.	
Pretrained	44.9	53.1	33.9	51.4	55.0	30.5	32.3	35.4	32.8	40.6	42.7
Individual	85.6	56.0	39.8	51.1	58.3	29.6	32.5	34.1	33.2	41.8	56.2
Weight Averaging (Wortsman et al., 2022a)	60.5	47.7	48.2	49.5	55.0	30.1	33.7	34.9	33.3	41.5	49.2
Task Arithmetic (Ilharco et al., 2023)	73.3	48.0	50.0	49.7	**73.0**	28.6	**34.1**	34.9	**33.1**	43.9	54.9
w/ Ours	**76.3**	**66.1**	**57.1**	**50.3**	71.0	**28.7**	31.5	**34.9**	33.0	**46.6** (+2.7)	**57.9** (+3.0)
Ties (Yadav et al., 2023)	71.0	64.3	62.5	**52.5**	70.0	30.2	**33.9**	35.8	**35.0**	48.0	57.3
w/ Ours	**71.9**	**67.9**	**67.9**	52.4	**74.0**	**31.2**	33.0	**37.2**	34.8	**49.8** (+1.8)	**58.8** (+1.5)
PCB-MERGING (Du et al., 2024)	**75.9**	70.0	**64.3**	51.7	73.0	30.6	**35.1**	**36.4**	**34.3**	49.4	59.8
w/ Ours	75.8	**70.0**	62.5	**52.2**	**78.0**	**31.2**	35.0	36.3	33.7	**49.9** (+0.5)	**60.2** (+0.4)
FR-Merging (Zheng & Wang, 2025)	73.9	50.9	48.2	50.5	72.0	28.3	34.2	34.3	32.8	43.9	50.1
w/ Ours	75.4	52.3	48.2	50.3	74.0	28.9	34.5	34.6	32.5	44.4 (+0.5)	55.9 (+0.8)

Table 22: Performance comparison for ID datasets on T5-large.

Method (↓)	PAWS	QASC	QuaRTz	StoryCloze	WikiQA	Winogrande	WSC	Avg.
Pretrained	48.1	33.8	53.1	47.0	37.9	50.9	43.3	44.9
Individual	95.4	97.6	91.9	90.4	95.8	79.2	51.0	85.9
Weight Averaging (Wortsman et al., 2022a)	55.9	71.6	54.4	56.4	76.7	53.4	54.8	60.5
Task Arithmetic (Ilharco et al., 2023)	64.6	74.4	75.0	80.8	**92.8**	63.1	**62.5**	73.3
w/ Ours	**85.5**	**77.4**	**77.9**	**81.0**	85.7	**67.2**	59.6	**76.3** (+3.0)
Ties (Yadav et al., 2023)	92.6	62.6	75.0	78.5	**58.7**	75.7	53.8	71.0
w/ Ours	**92.8**	**67.0**	**77.1**	**78.8**	55.6	**76.3**	**55.8**	**71.9** (+0.9)
PCB-MERGING (Du et al., 2024)	**92.3**	78.4	**77.3**	**76.8**	**70.8**	**74.4**	61.5	**75.9**
w/ Ours	92.2	**79.0**	76.6	76.0	69.9	74.3	**62.5**	75.8 (-0.1)
FR-Merging (Zheng & Wang, 2025)	84.7	66.0	70.8	78.4	92.9	64.9	59.6	73.9
w/ Ours	85.8	68.6	71.9	81.5	93.1	65.4	61.5	75.4 (+1.5)

redundancy with many local blocks containing overlapping information, allowing for robust compression by deleting entire Transformer layers while preserving important global information and reducing overfitting to high-frequency noise or specific dataset artifacts (Chen et al., 2022; Hou & Kung, 2022), language is sequential with meanings that are compositional and context-sensitive. Discarding entire layers might disrupt the sentence-level or token-level dependencies crucial for semantic understanding (Sajjad et al., 2023; Ding et al., 2025; Kenton & Toutanova, 2019). Therefore, for NLP tasks, we perform parameter pruning at the parameter level. As shown in Tab.21, compared to the baseline methods Task Arithmetic, Ties, PCB, and FR-Merging, our method achieves relative improvements of 3%, 1.5%, 0.4% and 0.8% on the H-score, respectively. The detailed results on the ID dataset are presented in Tab.22.

B.21 Parameter settings

Since the tasks and model architectures vary across our experiments, our hyperparameter selection strategy also differs accordingly.

B.21.1 Vision tasks

The vision experiments involve ViT-B/32, ViT-L/14, and ViT-H/14. To ensure fair comparison, we follow the tuning protocol below:

Tuning protocol.

- **ViT-B/32 / ViT-L/14:** We use the optimal hyperparameters reported in prior works and evaluate both baseline and baseline+Ours under these fixed settings.
- **ViT-H/14:** Since ViT-H/14 is fine-tuned by us, we first conduct a grid search within the hyperparameter ranges suggested by prior works to identify the optimal configuration for the baseline. We then apply the same configuration to the baseline+Ours setting for a fair comparison.

Hyperparameters. Following (Yadav et al., 2023; Yang et al., 2024b;a; Du et al., 2024; Zheng & Wang, 2025), we use:

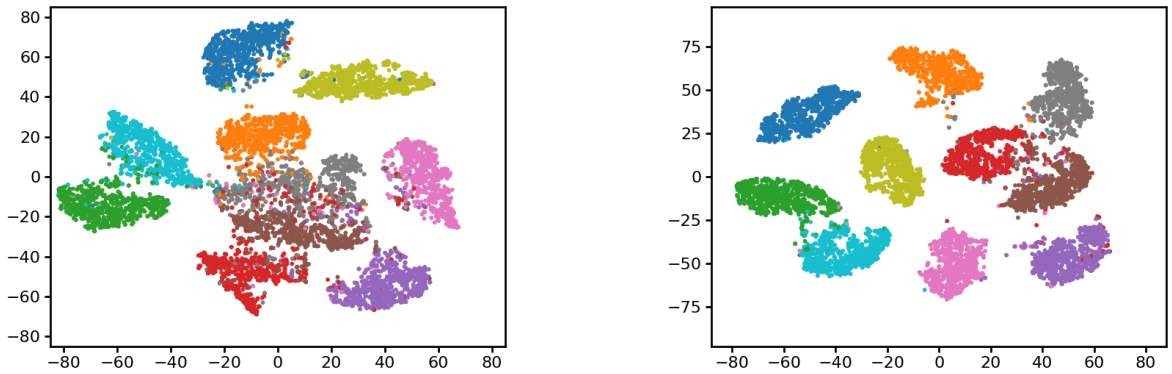
- **Task Arithmetic:** $\lambda = 0.3$
- **Ties:** $\lambda = 0.3$ (ViT-B/32), $\lambda = 0.4$ (ViT-L/14, ViT-H/14); retain the top-20% parameters
- **PCB:** $\lambda = 1.2$, mask ratio $r = 0.05$
- **FR-Merging:** $\lambda = 0.7$(ViT-B/32, ViT-L/14), $\lambda = 0.6$(ViT-H/14); filter ratio $f = 0.7$;
- **Ours (LwPTV):** pruning ratio $\eta = 0.7$

B.21.2 NLP Tasks

For T5-Large-LM-Adapt, although baseline and baseline+Ours use separate hyperparameter selections, the search ranges are consistent across both settings. Specifically, for models based on T5-Large-LM-Adapt, the existing baseline results are incomplete and were obtained with relatively narrow hyperparameter ranges. To ensure fairness, we independently tune the hyperparameters for both the baseline and baseline+Ours within the broader ranges provided in the relevant literature.

Search ranges. Following (Ilharco et al., 2023; Yadav et al., 2023; Du et al., 2024; Derek et al., 2024; Zheng & Wang, 2025), we adopt:

- **Task Arithmetic:** $\lambda \in [0.1, 1.5]$ (step 0.1)
- **Ties / PCB:** mask ratio $r \in \{0.05, 0.1, 0.2\}$; $\lambda \in [0.1, 2.5]$ (step 0.1);
- **FR-Merging:** $\lambda \in [0.1, 1.0]$ (step 0.1); filter ratio $f \in [0.1, 1.0]$ (step 0.1)
- **Ours (LwPTV):** pruning ratio $\eta \in [0.1, 0.7]$ (step 0.1)



(a) STL10: Task Arithmetic

(b) STL10: Task Arithmetic w/ LwPTV

Figure 11: T-SNE visualizations on STL10 comparing Task Arithmetic without and with LwPTV .

C Additional visualization results

C.1 T-SNE visualization

To explore the effectiveness of ours intuitively, we also present a T-SNE visualization for Task Arithmetic and the Task Arithmetic+LwPTV on OOD tasks, where the STL10 dataset is shown in Fig.11. Compared to

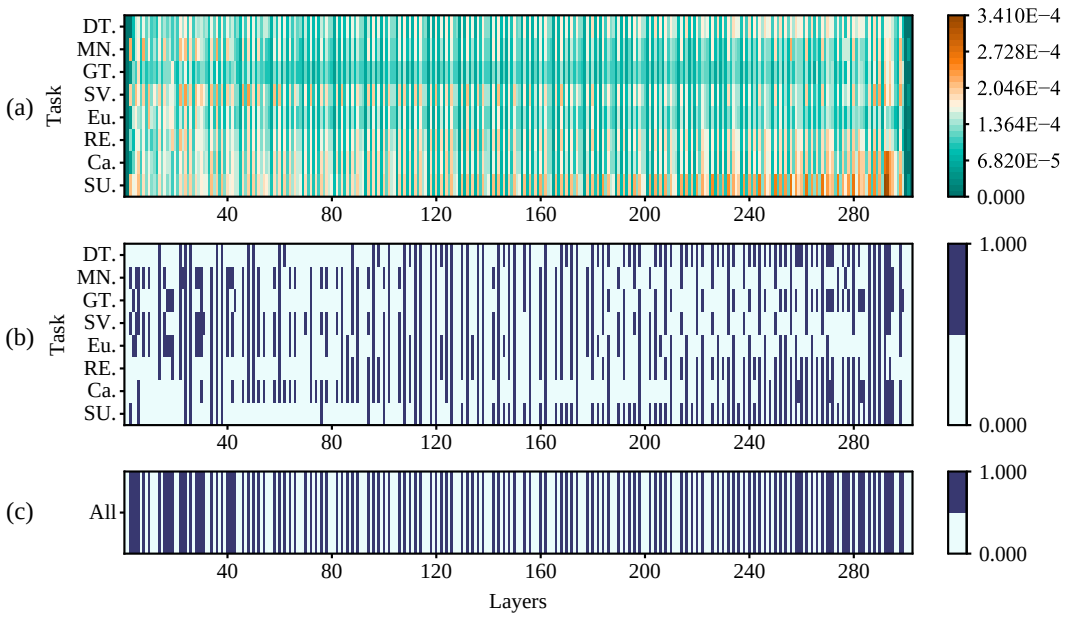


Figure 12: Visualization of (a) salience score matrix, (b) mask vector for each task, and (c) final mask vector, all on ViT-L/14, where x-axis denotes the layer index, y-axis in (a-b) denotes task name.

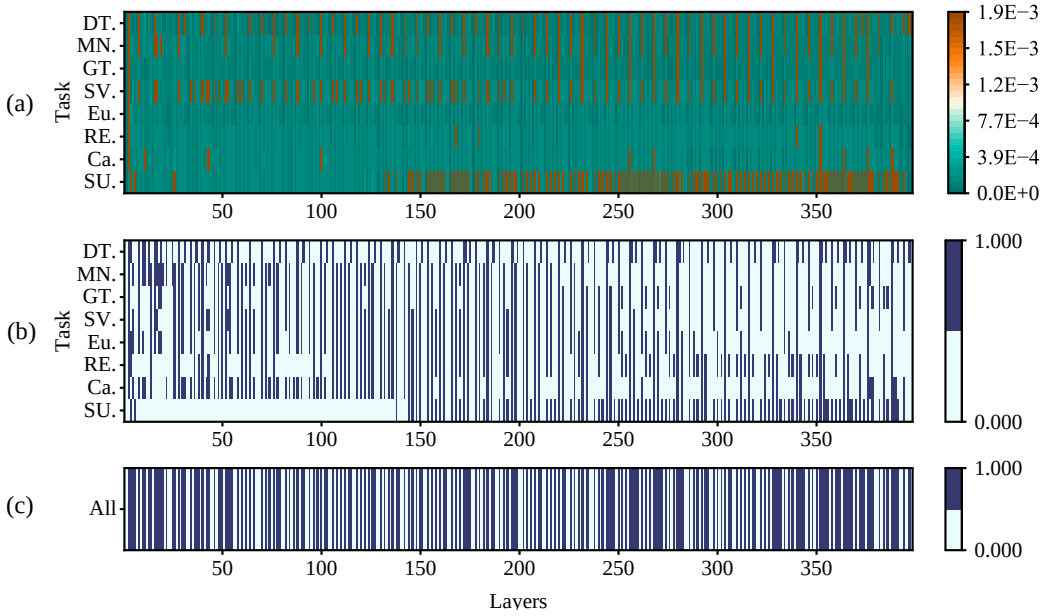


Figure 13: Visualization of (a) salience score matrix, (b) mask vector for each task, and (c) final mask vector, all on ViT-H/14, where x-axis denotes the layer index, y-axis in (a-b) denotes task name.

Task Arithmetic, our LwPTV demonstrates significantly enhanced the separability of representations across different categories. Furthermore, the intra-class clustering exhibits greater compactness when employing our approach. It indicates that introducing our proposed mask vector to prune the task vectors, LwPTV can extract discriminative features and thus enhances the generalization ability of the merged model.

C.2 Salience score and mask vectors

To facilitate an intuitive analysis of salience score, masks associated with each task vector and their interrelationships, we also visualize them for ViT-L/14 in Fig.12 and ViT-H/14 in Fig.13.