Trajectory Regularized: Reducing Storage Dependency in Model Merging for Continual Learning

Anonymous submission

Abstract

Model merging combines multiple expert models from independent tasks to construct a single powerful multi-task model. This objective aligns well with continual learning (CL), which requires the integration of knowledge dynamically acquired. However, the requirement for all expert models to be available simultaneously conflicts with the storage limitations inherent to CL. In real-world CL scenarios, the newly trained model can only be merged with the model available at the beginning of the current training task, and the resulting merged model must serve as the initialization for the next training stage. Surprisingly, existing model merging methods fail to demonstrate their expected advantages under these constraints. In this paper, we first conduct extensive evaluations of existing merging methods under the constraints of CL, revealing that they primarily prioritize global (task-agnostic) optimization, while local (task-specific) errors are progressively amplified during sequential training. Furthermore, the gradients of the merged model evolve slowly in the early stages of subsequent training, hindering its ability to rapidly adapt to new tasks. To address these challenges, we propose a trajectory regularized merging framework that reformulates the model merging phase as a search for an optimal point within the parameter subspace spanned by the new and previous task vectors. To guide the search progress, we design an evaluation mechanism that integrates three complementary optimization objectives, task alignment, prediction consistency, and curvature balance, which serve as supervisory signals to maximize the effectiveness and potential of model fusion during continual training. Extensive experimental results demonstrate that our method achieves state-of-the-art performance across multiple benchmarks.

Introduction

The rapid evolution of data in the real world presents ongoing challenges for deep learning models. Even for pre-trained models (PTMs), training on dynamic data streams often biases them toward new task data, leading to progressive degradation in performance on previously learned knowledge, a phenomenon known as catastrophic forgetting (Zhou et al. 2024). To address this issue, continual learning (CL) has been proposed to balance the model's stability on previously learned knowledge with its plasticity for learning new ones, thereby enabling learning from dynamic data streams. In recent years, CL methods for PTMs have often been adapted from parameter-efficient fine-tuning (PEFT) techniques, such as prompt (Wang et al. 2022c,b; Smith et al. 2023) and adapter (Huang et al. 2024). Recently, model merging (Ilharco et al. 2023; Wortsman et al. 2022) has emerged as a training-free paradigm for model adapting, where multiple independently trained, task-specific models are integrated into a unified multi-task model by selecting or interpolating task vectors, which capture the parameter differences between fine-tuned and initial model across tasks.

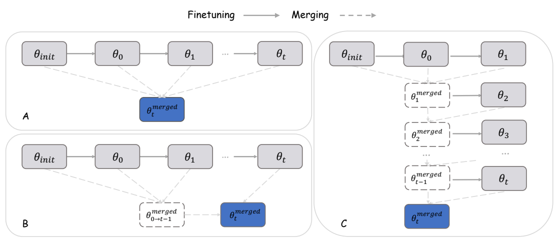

Figure 1: Description of the different merging paradigms. A) is the baseline that imposes no storage restrictions; B) requires retaining the fine-tuned model and the merged task vectors. C) is our proposed model merging paradigm suitable for real-world continuous learning. All experiments in this paper were conducted under the C.

However, as shown in Figure 1 A), we observe that existing merging methods require access to all previous task vectors and the initial model during the merging phase. This allows for the reconstruction of any previous model at any point in time, effectively equivalent to storing all prior models. Such a practice contradicts a fundamental principle of continual learning, which prohibits retaining old data or models in any form to prevent data leakage. Furthermore, storing task vectors requires the same amount of memory as storing models. As the number of tasks grows, this constant accumulation results in a substantial increase in memory consumption. Even though MagMax (Marczak et al. 2024), a specially proposed for continual learning, which reducing the need to store all previous models, still requires retaining both the finetuned model and the merged task vectors from all previous tasks, as shown in Figure 1 B), which does not fully eliminate the aforementioned issues. Therefore, in the scenario of continual learning, the ideal implementation is to directly merge the initial model of each stage with the trained model for each task, with the merged model serving as the initialization for the next training stage and no data storage or replay is permitted during the entire training phase, as shown in Figure 1 C). To evaluate the impact of this implementation of existing model merging methods, we conducted experiments using both the proposed storage constrained implementation and the baseline without storage limitations. The results clearly indicate that, under strict storage constraints, the performance of all selected methods degraded to varying degrees, with the largest performance drop reaching 6%. (Detailed experiments are provided in the supplementary materials.) Building on these findings, this paper aims to investigate the feasibility of merging for continual learning under stricter data storage constraints.

According to the findings of (Dziadzio et al. 2025), performance variation is primarily influenced by the initial model in each task. In this paper, we first design a series of experiments to examine the stability and plasticity of merged models. Based on our experimental results, we demonstrate that existing merging methods primarily focus on task agnostic (global) performance while neglecting task specific (local) optimality, as a result, errors in individual tasks are amplified during continual training, leading to significant negative consequences, and undermining the model's stability with respect to the currently learned knowledge. Furthermore, the merged model exhibits slow updates in the initial process when trained on new tasks, indicating limited adaptability to new tasks and making subsequent optimization more challenging. To jointly address these challenges, we propose a trajectory regularized merging framework that reformulates the merging phase as an optimization problem over the plane defined by the task vectors of the previous and current tasks. Simultaneously, we introduce an objective composed of three constraints designed to achieve task alignment, prediction consistency, and curvature balance, thereby identifying the optimal merging point. The merged model under this trajectory supervision striking a balance between the stability of previously learned knowledge and the plasticity required for acquiring new knowledge, enabling improved performance on continuous data streams without relying on any stored models or old data.

Our main contributions can be summarized as follows:

- We pointed out that existing model merging methods do not fully adhere to the principles of continual learning, and their performance degraded significantly when model storage is disallowed.
- We analyzed the cause of this performance gap and reformulated the merging phase as an optimal point search problem within the plane spanned by the task vectors.
- We proposed an objective composed of three constraints to guide the search for the optimal merging point.
- Our method achieves state-of-the-art performance across multiple benchmarks.

Related Work

Continual Learning

Continual learning aims to acquire new knowledge from a never-ending data stream continuously (Wang et al. 2022a; Zhu et al. 2021; Zhao et al. 2023; Wang et al. 2025). The primary challenge is learning without catastrophic forgetting: as new data arrives, the model's performance on previously learned tasks should not degrade significantly (Li and Hoiem 2017; Rebuffi et al. 2017). Recently, with the widespread adoption of pre-trained models, many continual learning methods have been developed as extensions of parameter-efficient fine-tuning (PEFT) methods. Prompt-based methods (Wang et al. 2022c,b; Smith et al. 2023; Qiao et al. 2024; Le et al. 2024) have demonstrated the effectiveness of migrating pre-trained models into continuous data streams, adapter-based methods (Huang et al. 2025; Gao et al. 2024; Tan et al. 2024; Yu et al. 2024; Liu et al. 2023), achieved high performance by training only a small number of parameters. Additionally, there are methods (Khan et al. 2023; Yu et al. 2025) that consider the knowledge of language modality to aid in modeling learning.

Model Merging

Model merging has recently gained significant attention as a practical technique for aggregating multiple models by performing linear interpolation in parameter space (Xu et al. 2024). The core idea traces back to ensemble learning methods such as Bagging Predictors (Breiman 1996), which improve generalization by averaging the outputs of diverse models. Stochastic Weight Averaging (Izmailov et al. 2018) aggregates gradients over training to yield wider optima and improved robustness. Subsequently, recent methods shift the focus from output space to a weight space combination. numerous methods (Wortsman et al. 2022; Jang, Yun, and Han 2024; Ilharco et al. 2023; Yadav et al. 2023) have explored weight combination to achieve model merging, and MagMax (Marczak et al. 2024) extends model merging to the continual learning, achieving excellent performance through the appropriate storage of model parameters.

Background and Motivation

Problem Setting

We consider a supervised continual learning based on pre-trained models, where a pre-trained model $g_{\theta}^{initial}$ parametrized by $\theta_{initial}$ is required to learn a sequence of T tasks in order. Each task dataset D^t contains different classes C^t and there is no overlapping between any two different tasks, i.e., $C^i \cap C^j = \emptyset$, $i \neq j$ and $(x, y) \in D^t$ denotes the training sample in task t. For each task t, training is strictly initialized from the model f_{θ}^{t-1} obtained after task $t-1$, and any previous models or data are permitted to be stored in any form except for $f_{\theta}^{initial}$. For simplicity, we will use model parameters θ_i to represent the model i in the following content for convenience.

Therefore, we investigate why using the merged model as the initialization for the next training stage has such a significant impact on overall performance.

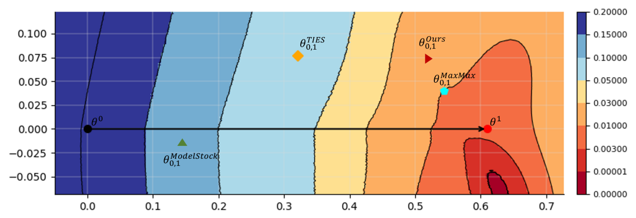

Figure 2: Loss landscape. We visualized the loss landscape along the trajectory from θ_0 to θ_1 and projected all merged points onto this surface based on their corresponding loss values.

Analysis

We randomly constructed three disjoint tasks T_0, T_1, and T_2 from the ImageNet-R, each task consists of 20 non-overlapping classes. The model θ_0 trained on task 0 is used as the initial model for subsequent training, and we choose the existing method MagMax (Marczak et al. 2024), TIES (Ilharco et al. 2023), and Model Stock (Jang, Yun, and Han 2024) as representative merging method for our experiments.

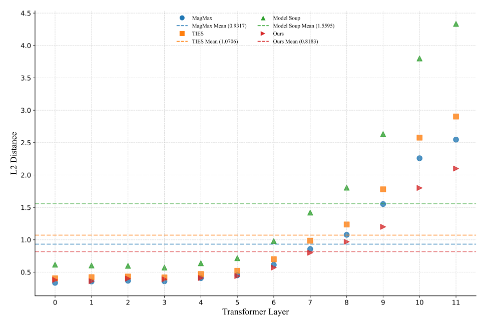

Figure 3: Output drift between θ_1 and $\theta_{0,1}^{f}$. We visualize the differences between different layers of the model for the same input.

Loss for Current Task. In order to investigate the retention capability of the merged model with respect to previously learned knowledge, we directly finetune θ_0 on T_1 to obtain θ_1, and then get the merged model $\theta_{0,1}^{f}$, where f indicates the different merging methods. We select θ_0 and θ_1 as the interpolation endpoints, plot the loss landscape over D_1, and superimpose the position of the $\theta_{0,1}^{f}$ on the same graph according to its loss. The results are presented in Figure 2. As shown, the trajectory of finetuning ultimately converges to a low loss region, whereas the merged model exhibits a substantially higher loss on the same training data, consistently falling into a suboptimal region, regardless of which merging method we used. We argue that existing merging methods primarily focus on optimizing global performance, often resulting in intermediate parameter states that deviate from the optimal solution for any individual task. Furthermore, task-specific errors tend to accumulate across tasks, ultimately leading to irreversible performance degradation, thereby hindering continual learning.

Output for Current Task. At the same time, we also measured the output differences between θ_1 and the merged model $\theta_{0,1}^{f}$ across corresponding layers on D^1. For any input x_j, the difference of layer j can be expressed as:

$$\Delta_{out}^{l} = \frac{1}{|D^1|}\sum_{j=1}^{|D^1|}\left\|\theta_{0,1}^{f}(x_j)_{[0]}^{l} - \theta_1(x_j)_{[0]}^{l}\right\|_2, \quad (1)$$

where l is the corresponding layer, $|D^1|$ is the number of samples of T^1 and $[0]$ indicates the class token. The results are shown in Figure 3. It can be observed that all merging methods alter the model's output under the same input, and this difference becomes increasingly pronounced in deeper layers. We believe that model parameters are highly interdependent after optimization, and the merging phase may introduce abrupt changes in certain parameters. This disruption affects the structured semantic representations encoded in the parameter space, ultimately leading to output drift. Meanwhile, the varying effects of such perturbations across layers support the conclusion previously presented in (Zheng et al. 2025), the lower layers of the model encode general information, while the higher layers encode task-specific information.

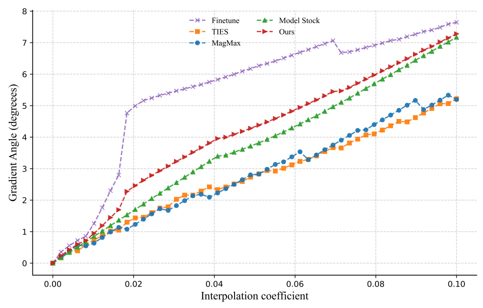

Figure 4: Gradient variations. We visualize the gradient variations between the initial point and neighboring points along the training trajectory.

Gradient Variations for Future Task. To explore the ability of the merge model to learn future knowledge, we use θ_1 and $\theta_{0,1}^{f}$ as the initialization, perform finetuning on task T^2 to obtain the corresponding θ_2, and then interpolate along the corresponding training trajectory, $\theta_1 \rightarrow \theta_2$ and $\theta_{0,1}^{f} \rightarrow \theta_2$ to calculate the gradient variation between the initial point and neighboring points on the future task,

$$\Delta_{\theta}(\delta) = argcos\left(\nabla_{\theta}\mathcal{L}(\theta), \nabla_{\theta}\mathcal{L}(\theta+\delta)\right), \quad (2)$$

where $\mathcal{L}$ indicates the loss function and δ is the perturbation on the training trajectory. The experiment results are shown in Figure 4. Within a small perturbation range, the gradient variations at neighboring points for all merged models are smaller than continuous finetuning. We believe that a larger

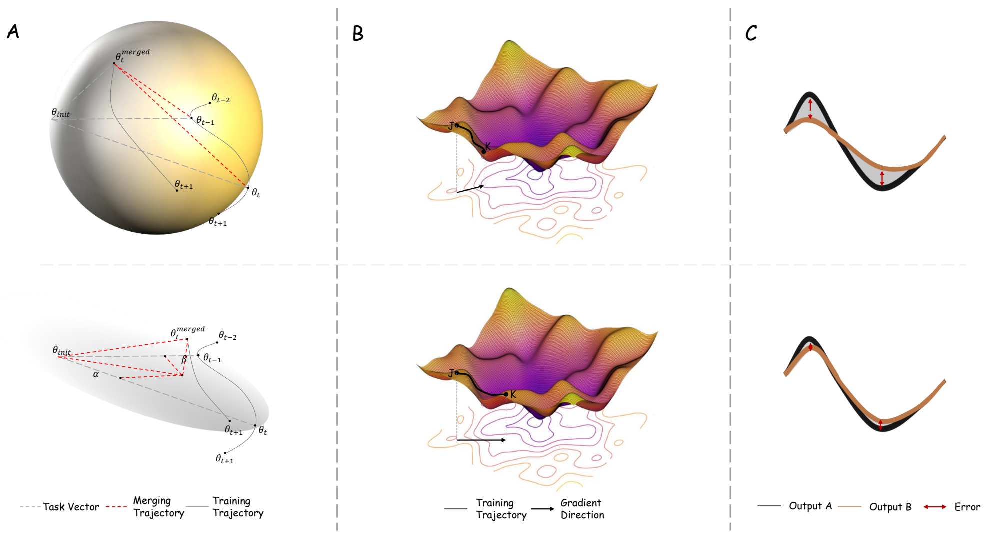

Figure 5: Visualization of our method. The bottom row illustrates the implementation of our method, while the top row presents the existing method for comparison. A depicts the merging phase, which we formulate as a search problem within a parameter plane. B represents the loss surface constraints, incorporating both loss and curvature. C shows the model outputs, where our objective is to maintain consistency.

gradient angle shift indicates that the model is highly sensitive to new data and actively adapting to new tasks. In contrast, a smaller gradient fluctuation suggests that the model is trapped in a flat or saturated region, updates slowly, lacks sensitivity to new tasks, and therefore exhibits limited plasticity with respect to future tasks.

Our experiments suggest that existing model merging methods primarily target global performance by locating a global optimum using all historical task vectors. However, this often comes at the cost of disrupting the structured organization of the parameter space. Even without considering the inaccessibility of past models in real-world continual learning scenarios, the lack of plasticity already poses significant challenges to continuous finetuning. Moreover, local errors tend to accumulate over time as tasks progress, eventually leading to a decline in overall performance.

To address these issues, we propose a trajectory regularized merging framework to mitigate the limitations of model merging in continual learning applications.

Method

In our method, the training of each individual task $t, t > 0$ is divided into two phases, the training phase and the merging phase.

In the training phase, finetuning starts from θ_{t-1}, which obtained from the previous stage, and produces model θ_t under the supervision of the cross-entropy loss $\mathcal{L}_{ce}$.

In the merging phase, we first construct the task vectors v_{t-1} and v_t, where $v_i = \theta_i - \theta_{initial}$. Based on the above analysis, to preserve parameter dependencies and mitigate abrupt structural changes, we aim to keep the merged process as linear as possible, as illustrated in the lower half of Figure 5 A. Therefore, we apply simple linear incorporate to derive the merged task vector, $v_{merged} = \alpha \cdot v_t + (1-\alpha) \cdot v_{t-1}$, meanwhile, we aim not to restrict the merged point to the plane spanned by v_{t-1} and v_t completely, we randomly sampled unit vector P from the subspace orthogonal to the direction $P^\top = v_t - v_{t-1}$ and

$$P = Normalize\left(\tilde{r} - \frac{\langle \tilde{r}, P^\top \rangle}{\|P^\top\|^2} P^\top\right), \quad \tilde{r} \sim \mathcal{N}(0, I), \tag{3}$$

we introduce directional perturbations to allow the merging point to deviate from the plane spanned by the task vectors . Therefore, the resulting merged task vector is given by :

$$v_{merged} = \alpha \cdot v_t + (1-\alpha) \cdot v_{t-1} + \beta \cdot P, \tag{4}$$

where α and β are the interpolation coefficients. The merged model can be denoted as,

$$\theta_t := \theta_{t-1,t}^{merged} = \theta_0 + v_{merged}. \tag{5}$$

To evaluate the quality of the merged model, we propose an evaluation function composed of three constraints, which serving as the objective for coefficient determination.

Task Alignment

To alleviate the progressive accumulation of model errors over the continual data stream after merging, the most straightforward method is to minimize the model's loss on the current task. Therefore, one of the criteria for evaluating the quality of the merged model is,

$$\mathcal{L}_{task} = \mathcal{L}_{ce}(x, y), (x, y) \in D^t. \tag{6}$$

Prediction Consistency

The loss constraint helps preserve model performance on the current task. At the same time, to minimize representation drift caused by disruption of the parameter structure, we aim to maintain consistency in parameter relationships within the merged model. Therefore, we propose prediction consistency term that encourages the internal feature representations of the merged model to align with those of the task-specific model.

$$\begin{aligned}\mathcal{L}_{pre} = \sum_{l=1}^{L} \omega_l \Bigg\| \left(\theta_{t-1,t}^{\text{merged}}(x)\right)_{[0]}^{l} \\ -\frac{1}{2}\left(\theta_t(x)_{[0]}^{l} + \theta_{t-1}(x)_{[0]}^{l}\right) \Bigg\|_2^2,\end{aligned} \tag{7}$$

where $\omega_l = \frac{q(l)}{\sum_{l=1}^{L} q(l)}$ and $q(l) = e$ befor the layer b and $q(l) = e^{l-b}$ otherwise.

Curvature Balance

Finally, to enhance the model's gradient responsiveness to new knowledge, we aim to prevent the merged model from residing in flat or saturated regions of the loss landscape. In other words, we encourage it to lie outside regions of minimal curvature. To this end, we introduce a curvature balance constraint.

Ideally, the local curvature near the model parameters can be estimated using the trace of the Hessian of the loss function,

$$\mathcal{C}(\theta^{merged}) = Tr\left(\nabla^2_{\theta^{merged}} \mathcal{L}(\theta^{merged})\right). \tag{8}$$

However, due to the high dimensionality of the parameter space, directly computing the Hessian matrix is highly complex. To improve efficiency, we first perform a second-order Taylor expansion of the loss function around a local optimum θ^*.

$$\begin{aligned}\mathcal{L}(\theta) \approx \mathcal{L}(\theta^*) + \nabla_\theta \mathcal{L}(\theta^*)^\top (\theta - \theta^*) \\ + \frac{1}{2}(\theta - \theta^*)^\top \nabla^2_\theta \mathcal{L}(\theta^*)(\theta - \theta^*),\end{aligned} \tag{9}$$

we take the first derivative of both sides, and then square them simultaneously,

$$\|\nabla_\theta \mathcal{L}(\theta)\|_2^2 \approx (\theta - \theta^*)^\top \left(\nabla^2_\theta \mathcal{L}(\theta^*)\right)^2 (\theta - \theta^*). \tag{10}$$

Since the eigenvalue decomposition of the Hessian matrix can be expressed as

$$\nabla^2_\theta \mathcal{L}(\theta^*) = U \Lambda U^\top, \quad \Lambda = \mathrm{diag}(\lambda_1, \ldots, \lambda_n), \tag{11}$$

where λ_i is the eigenvalue, and U is the matrix of eigenvectors, Then, equation (10) can further equated after we indicate $z = U^\top(\theta - \theta^*)$

$$\|\nabla_\theta \mathcal{L}(\theta)\|_2^2 \approx z^\top \Lambda^2 z = \sum_{i=1}^{n} \lambda_i^2 z_i^2. \tag{12}$$

We initially aimed to use the trace of the Hessian matrix to estimate the overall curvature at a given point on the loss surface. Since the Hessian matrix is a real symmetric matrix, its trace is equal to the sum of all eigenvalues. Additionally, the sum of the squares of these eigenvalues captures the squared magnitude of total curvature, amplifying sharp directions in the surface geometry in a nonlinear fashion, thereby providing a more sensitive measure of local sharpness variations. As shown in Equation (12), the square of the gradient norm correlates positively with the sum of squared Hessian eigenvalues. Therefore, in our experiments, we use the squared gradient norm, also referred to as gradient energy to approximate curvature.

$$\mathcal{L}_{curv}(\theta) = -\|\nabla_\theta \mathcal{L}(\theta)\|_2^2. \tag{13}$$

Finally, we evaluate the merged model using three complementary constraints:

$$\mathcal{L}_{tra} = \mathcal{L}_{task} + \lambda_1 \mathcal{L}_{pre} + \lambda_2 \mathcal{L}_{curv}. \tag{14}$$

In summary, we formulate the model fusion process as a search problem in the space spanned by the task vectors, where the quality of each candidate point is evaluated based on the proposed trajectory regularized objective.

During the merging phase, we treat the interpolation coefficient as the sole trainable parameter, which is optimized via gradient backpropagation of the Equation (14) on the corresponding dataset,

$$\min_{\alpha,\beta} \mathbb{E}_{(x,y)\sim D^t}\left[\mathcal{L}_{tra}(\theta^{merged})\right]. \tag{15}$$

In the actual merging phase, to prevent early convergence to a local optimum during optimization, we initialize θ_0 in Equation (5) with a randomly generated point in the search space by randomly place a part parameters in θ_0 from the corresponding positions in θ_{t-1} and θ_t.

Experiments

Experiments Setttings

Datasets. Following (Marczak et al. 2024), we selected three widely used datasets, CIFAR100 (Krizhevsky, Hinton et al. 2009), ImageNet-R (Hendrycks et al. 2021) and fine-grained Stanford Cars (Krause et al. 2013) for class incremental learning (CIL), and we divided each dataset into 5, 10, and 20 tasks. We selected ImageNet-R and DomainNet (Peng et al. 2019) for domain incremental learning (DIL). DomainNet is divided into 6 independent tasks based on domain categories, while ImageNet-R is split into 15 independent tasks. Similarly, we split these datasets into the corresponding number of tasks following the setting of CIL for a fair comparison of CIL and DIL performance.

Metrics. We use the standard metrics in the continual learning methods to measure performance: Task-agnostic Accuracy, which calculates all seen classes' accuracy after the final task.

Comparison Methods. We compared our method against current state-of-the-art methods, including traditional method, LwF (Li and Hoiem 2017) and EWC (Kirkpatrick et al. 2017), as well as methods based on PEFT, L2P (Wang et al. 2022c), DualPrompt (Wang et al. 2022b), CODAPrompt (Smith et al. 2023), RAPF (Huang et al. 2024) and CLG-CBM (Yu et al. 2025). We also include traditional model merging methods, Model Stock and TIES, and the model merged methods specifically designed for

	Method (Year)	CIFAR100			ImageNet-R			Stanford-Cars		
		5	10	20	5	10	20	5	10	20
	Zero-shot		66.7			72.2			64.4	
	Joint		89.4			86.5			84.4	
Conventional	LwF (2017)	68.2	67.4	67.0	66.3	60.9	55.6	65.2	61.7	57.5
	EWC (2017)	69.3	64.8	57.3	74.5	66.7	54.1	65.8	61.3	58.0
PEFT	L2P (2022)	77.2	73.1	68.5	77.9	75.7	72.5	67.8	66.7	65.0
	DualPrompt (2023)	77.4	73.6	70.4	78.6	75.9	73.4	68.9	67.1	65.2
	CODAPrompt (2023)	77.3	75.0	74.1	78.9	76.0	73.7	67.4	67.4	64.9
	RAPF (2024)	79.2	<u>79.0</u>	**79.2**	80.1	79.5	78.3	–	–	–
	CLG-CBM (2025)	78.1	76.8	75.9	79.2	78.8	77.5	<u>72.8</u>	<u>68.7</u>	64.2
Model Merging	Model Stock (2024)	76.5	70.7	68.3	79.6	77.1	76.5	66.6	64.5	64.1
	TIES (2023)	79.8	76.3	73.6	81.5	79.5	78.4	67.9	66.1	65.3
	MagMax (2024)	<u>80.4</u>	75.8	72.9	<u>82.2</u>	<u>80.1</u>	<u>79.3</u>	68.5	65.9	<u>65.7</u>
	PM (2025)	–	–	–	81.4	79.9	76.3	–	–	–
	Ours	**83.5**	**80.5**	<u>78.6</u>	**83.6**	**83.2**	**82.7**	**73.2**	**70.4**	**66.9**

Table 1: Comparison experiments on different benchmarks, **bolded** indicates optimal, <u>underlined</u> indicates sub-optimal.

CL, MagMax and PM (Qiu et al. 2025). In the experiments, the backbones of all compared methods are kept consistent with ours, and all results are obtained by executing all methods in the same environment and strictly adhering to the source code parameters.

Implementation Details. The image encoder is CLIP version of ViT-B/16, and the training batch size is 128, the learning rate is $1e-5$, and cosine annealing learning rate schedule and AdamW optimizer with weight decay 0.1. The training epoch is 15 and merging epoch is 5 for all dataset. We set $\lambda_1 = 0.1$, $\lambda_2 = 0.01$, sensitivity analysis is provided in the supplementary material. We use the CLIP's text encoder to encode labels and use its output as a classifier. Except for the image encoder, all other components are kept frozen throughout training.

Method	**DomainNet**		**ImageNet-R**	
	CIL	**DIL**	**CIL**	**DIL**
LwF	57.3	59.5	58.2	61.7
EWC	57.9	59.2	63.5	65.1
Model Stock	63.9	70.2	77.0	82.6
TIES	66.6	66.2	78.8	83.5
MagMax	68.4	68.7	80.0	83.9
Ours	**69.5**	**70.3**	**83.1**	**84.9**

Table 2: Comparison of different methods on DomainNet and ImageNet-R under CIL and DIL settings.

Experimental Results

Class Incremental Learning. Experimental results for CIL are shown in Table 1. On the CIFAR100 dataset, our method achieves accuracy of 83.5% and 80.5% in the 5 and 10 tasks respectively, representing improvements of 3.1% and 1.5% over the previous SOTA method. After learning 20 consecutive tasks, the performance is slightly lower than the optimal RAPF. On the ImageNet-R, our method achieves the best performance across all settings, with accuracies of 83.6%, 83.2%, and 82.7% for 5, 10, and 20 tasks, corresponding to improvements of 1.4%, 3.1%, and 3.4% over previous SOTA methods. On the fine-grained Stanford Cars dataset, our method achieves 73.2%, 70.4%, and 66.9% for all settings, representing gains of 0.4%, 1.7%, and 1.2%, respectively. In summary, the proposed method effectively mitigates catastrophic forgetting under continuous data streams without storing any previous model or data distribution.

Domain Incremental Learning. In Table 2, for both CIL and DIL settings, we report results only on the complete test set as a comparison and do not evaluate performance on individual domains. As Table shown, our proposed method consistently outperforms all other comparison methods across all scenarios. On the DomainNet, our method achieved performance of 69.5% and 70.3% under the CIL and DIL settings, respectively. On the ImageNet-R, the corresponding results were 83.1% and 84.9%. This demonstrates that our method exhibits strong compatibility and adaptability across different training scenarios.

Ablation Study

In this section, we examine the effectiveness of each component within our proposed method. The experimental setting is ImageNet-R with 10 tasks, and the results are shown in Table 3. As shown, randomly replacing the parameters of θ_{init} with those at the corresponding positions of θ_{t-1} and θ_t achieves results comparable to those obtained using a standard merged methods, achieving only 79.3% accuracy. Our three proposed constraints, task alignment, prediction consistency, and curvature balance, when used individually, fail to balance plasticity and stability, which skews the search

Table 3: Ablation study on ImageNet-R, 10 tasks. a) indicates randomly mix.

	Components			Accuracy
	$\mathcal{L}_{init}$	$\mathcal{L}_{rep}$	$\mathcal{L}_{curv}$	
a)				79.3
b)	✓			71.8
c)		✓		72.4
d)			✓	76.5
e)	✓	✓		70.7
f)	✓		✓	81.9
g)		✓	✓	80.6
k)	✓	✓	✓	83.2

process, resulting in performance decreases of 7.5%, 6.9%, and 2.8%, respectively. When the two stability constraints are used together, performance drops by 8.6%. When we fix the curvature balance and pair it with either the task alignment or prediction consistency, performance improves by 2.6% and 1.3% compared to random merged. When all three constraints are used simultaneously, performance reaches 83.2%.

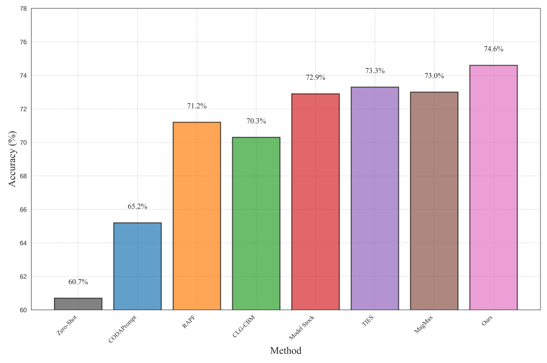

Figure 6: Experiments on ImageNet1K are conducted by evenly dividing the 1000 classes into 100 distinct training tasks.

Further Analysis

Large Scale Dataset. We conducted experiments on the large-scale ImageNet1K dataset by evenly dividing its 1000 classes into 100 non-overlapping tasks. The experimental results are presented in the Figure 6. As the number of tasks increases, the difficulty of learning on continue data stream grows significantly; nevertheless, our method consistently achieves the best performance, demonstrating its scalability and robustness to data scale.

The Ration of Replaced Parameters. At the end of the methods section, we noted that during training, we randomly replaced a portion of the parameters in the initial model θ_{init} with parameters from the corresponding positions in models θ_{t-1} and θ_t to ensure optimization stability. To validate this, we conducted experiments on 10 tasks from the ImageNet-R dataset. In experiments, the position of the replaced parameters and the source of the replacement (from θ_{t-1} or θ_t) are completely random, only the overall replacement ratio was controlled. For each ratio, we performed 10 full training and recorded the maximum, minimum, and average results. The experiments results are shown in the Figure 7. As observed, using the original parameters as the initialization led to slightly lower performance and exhibited a wider range between the best and worst results compared to experiments with partial parameter replacement. As the replacement ratio increased, the variance in performance initially decreased and then gradually increased. Based on these findings, we set the parameter replacement ratio to 0.6 in our experiments.

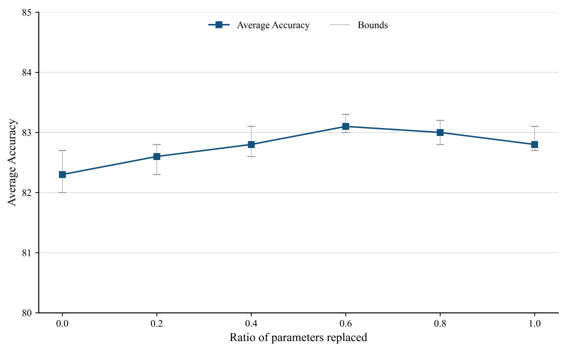

Figure 7: Gradient variations. We visualize the gradient variations between the initial point and neighboring points along the training trajectory. The horizontal axis represents the interpolation coefficient along the trajectory.

Others. In the analysis section, we presented experiments on loss, output, and gradient changes, with our results already highlighted in Figures 2-4. Due to space limitations, detailed analyses and additional experiments, such as time consumption, different backbone are all provided in the supplementary material.

Conclusion

In this paper, we investigate the limitations of existing model merging methods in realistic continual learning scenarios, where only the pre-task and post-task models are accessible, and the merged model must serve as the initialization for future training. Our analysis reveals that existing model merging method often prioritize global optimization, while neglecting local task-specific adaptation, resulting in accumulated errors and slow gradient evolution during training.

To address these challenges, we propose a trajectory regularized merging framework, reformulating model merging as an optimal point search within the subspace spanned by the task vectors of the current and previous tasks, guided by three complementary objectives, task alignment, prediction consistency, and curvature balances, to improve the stability and plasticity of the merged model. Extensive experiments across multiple benchmarks confirm the effectiveness of our method under strict continual learning constraints, achieving state-of-the-art performance.

References

Breiman, L. 1996. Bagging predictors. *Machine learning*, 24(2): 123–140.

Dziadzio, S.; Udandarao, V.; Roth, K.; Prabhu, A.; Akata, Z.; Albanie, S.; and Bethge, M. 2025. How to Merge Your Multimodal Models Over Time? In *CVPR*, 20479–20491. Computer Vision Foundation / IEEE.

Gao, X.; Dong, S.; He, Y.; Wang, Q.; and Gong, Y. 2024. Beyond prompt learning: Continual adapter for efficient rehearsal-free continual learning. In *ECCV*, 89–106. Springer.

Hendrycks, D.; Basart, S.; Mu, N.; Kadavath, S.; Wang, F.; Dorundo, E.; Desai, R.; Zhu, T.; Parajuli, S.; Guo, M.; et al. 2021. The many faces of robustness: A critical analysis of out-of-distribution generalization. In *ICCV*, 8340–8349.

Huang, L.; Cao, X.; Lu, H.; and Liu, X. 2024. Class-incremental learning with clip: Adaptive representation adjustment and parameter fusion. In *ECCV*, 214–231. Springer.

Huang, L.; Cao, X.; Lu, H.; and Liu, X. 2025. Class-incremental learning with clip: Adaptive representation adjustment and parameter fusion. In *ECCV*, 214–231.

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

Izmailov, P.; Podoprikhin, D.; Garipov, T.; Vetrov, D.; and Wilson, A. G. 2018. Averaging weights leads to wider optima and better generalization. *arXiv preprint arXiv:1803.05407*.

Jang, D.-H.; Yun, S.; and Han, D. 2024. Model stock: All we need is just a few fine-tuned models. In *ECCV*, 207–223. Springer.

Khan, M. G. Z. A.; Naeem, M. F.; Van Gool, L.; Stricker, D.; Tombari, F.; and Afzal, M. Z. 2023. Introducing language guidance in prompt-based continual learning. In *ICCV*, 11463–11473.

Kirkpatrick, J.; Pascanu, R.; Rabinowitz, N.; Veness, J.; Desjardins, G.; Rusu, A. A.; Milan, K.; Quan, J.; Ramalho, T.; Grabska-Barwinska, A.; et al. 2017. Overcoming catastrophic forgetting in neural networks. *Proceedings of the national academy of sciences*, 114(13): 3521–3526.

Krause, J.; Stark, M.; Deng, J.; and Fei-Fei, L. 2013. 3d object representations for fine-grained categorization. In *Proceedings of the IEEE international conference on computer vision workshops*, 554–561.

Krizhevsky, A.; Hinton, G.; et al. 2009. Learning multiple layers of features from tiny images.

Le, M.; Nguyen, H.; Nguyen, T.; Pham, T.; Ngo, L.; Ho, N.; et al. 2024. Mixture of experts meets prompt-based continual learning. *NeurIPS*, 37: 119025–119062.

Li, Z.; and Hoiem, D. 2017. Learning without forgetting. *IEEE Trans. Pattern Anal. Mach. Intell.*, 40(12): 2935–2947.

Liu, Z.; Zhang, J.; Asadi, K.; Liu, Y.; Zhao, D.; Sabach, S.; and Fakoor, R. 2023. Tail: Task-specific adapters for imitation learning with large pretrained models. *arXiv preprint arXiv:2310.05905*.

Marczak, D.; Twardowski, B.; Trzciński, T.; and Cygert, S. 2024. Magmax: Leveraging model merging for seamless continual learning. In *ECCV*, 379–395. Springer.

Peng, X.; Bai, Q.; Xia, X.; Huang, Z.; Saenko, K.; and Wang, B. 2019. Moment matching for multi-source domain adaptation. In *ICCV*, 1406–1415.

Qiao, J.; Tan, X.; Chen, C.; Qu, Y.; Peng, Y.; Xie, Y.; et al. 2024. Prompt gradient projection for continual learning. In *ICLR*.

Qiu, H.; Zhang, M.; Qiao, Z.; and Nie, L. 2025. Train with Perturbation, Infer after Merging: A Two-Stage Framework for Continual Learning. *arXiv preprint arXiv:2505.22389*.

Rebuffi, S.-A.; Kolesnikov, A.; Sperl, G.; and Lampert, C. H. 2017. icarl: Incremental classifier and representation learning. In *CVPR*, 2001–2010.

Smith, J. S.; Karlinsky, L.; Gutta, V.; Cascante-Bonilla, P.; Kim, D.; Arbelle, A.; Panda, R.; Feris, R.; and Kira, Z. 2023. Coda-prompt: Continual decomposed attention-based prompting for rehearsal-free continual learning. In *CVPR*, 11909–11919.

Tan, Y.; Zhou, Q.; Xiang, X.; Wang, K.; Wu, Y.; and Li, Y. 2024. Semantically-shifted incremental adapter-tuning is a continual vitransformer. In *CVPR*, 23252–23262.

Wang, X.; Yang, X.; Wei, K.; Gu, Y.; and Deng, C. 2025. Class Incremental Learning via Contrastive Complementary Augmentation. *IEEE Trans. Image Process.*

Wang, Z.; Shen, L.; Fang, L.; Suo, Q.; Duan, T.; and Gao, M. 2022a. Improving task-free continual learning by distributionally robust memory evolution. In *ICML*, 22985–22998. PMLR.

Wang, Z.; Zhang, Z.; Ebrahimi, S.; Sun, R.; Zhang, H.; Lee, C.-Y.; Ren, X.; Su, G.; Perot, V.; Dy, J.; et al. 2022b. Dual-prompt: Complementary prompting for rehearsal-free continual learning. In *ECCV*, 631–648. Springer.

Wang, Z.; Zhang, Z.; Lee, C.-Y.; Zhang, H.; Sun, R.; Ren, X.; Su, G.; Perot, V.; Dy, J.; and Pfister, T. 2022c. Learning to prompt for continual learning. In *CVPR*, 139–149.

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

Xu, Z.; Yuan, K.; Wang, H.; Wang, Y.; Song, M.; and Song, J. 2024. Training-free pretrained model merging. In *CVPR*, 5915–5925.

Yadav, P.; Tam, D.; Choshen, L.; Raffel, C. A.; and Bansal, M. 2023. Ties-merging: Resolving interference when merging models. *NeurIPS*, 36: 7093–7115.

Yu, J.; Zhuge, Y.; Zhang, L.; Hu, P.; Wang, D.; Lu, H.; and He, Y. 2024. Boosting continual learning of vision-language models via mixture-of-experts adapters. In *CVPR*, 23219–23230.

Yu, L.; Han, H.; Tao, Z.; Yao, H.; and Xu, C. 2025. Language Guided Concept Bottleneck Models for Interpretable Continual Learning. In *CVPR*, 14976–14986.

Zhao, H.; Zhou, T.; Long, G.; Jiang, J.; and Zhang, C. 2023. Does continual learning equally forget all parameters? In *ICML*, 42280–42303. PMLR.

Zheng, J.; Cai, X.; Qiu, S.; and Ma, Q. 2025. Spurious Forgetting in Continual Learning of Language Models. In *ICLR*. OpenReview.net.

Zhou, D.-W.; Wang, Q.-W.; Qi, Z.-H.; Ye, H.-J.; Zhan, D.-C.; and Liu, Z. 2024. Class-incremental learning: A survey. *IEEE Trans. Pattern Anal. Mach. Intell.*

Zhu, F.; Cheng, Z.; Zhang, X.-Y.; and Liu, C.-l. 2021. Class-incremental learning via dual augmentation. *NeurIPS*, 34: 14306–14318.

Reproducibility Checklist

1. General Paper Structure

1.1. Includes a conceptual outline and/or pseudocode description of AI methods introduced (yes/partial/no/NA) partial

1.2. Clearly delineates statements that are opinions, hypothesis, and speculation from objective facts and results (yes/no) yes

1.3. Provides well-marked pedagogical references for less-familiar readers to gain background necessary to replicate the paper (yes/no) yes

2. Theoretical Contributions

2.1. Does this paper make theoretical contributions? (yes/no) no

If yes, please address the following points:

2.2. All assumptions and restrictions are stated clearly and formally (yes/partial/no) NA

2.3. All novel claims are stated formally (e.g., in theorem statements) (yes/partial/no) NA

2.4. Proofs of all novel claims are included (yes/partial/no) NA

2.5. Proof sketches or intuitions are given for complex and/or novel results (yes/partial/no) NA

2.6. Appropriate citations to theoretical tools used are given (yes/partial/no) NA

2.7. All theoretical claims are demonstrated empirically to hold (yes/partial/no/NA) NA

2.8. All experimental code used to eliminate or disprove claims is included (yes/no/NA) NA

3. Dataset Usage

3.1. Does this paper rely on one or more datasets? (yes/no) yes

If yes, please address the following points:

3.2. A motivation is given for why the experiments are conducted on the selected datasets (yes/partial/no/NA) yes

3.3. All novel datasets introduced in this paper are included in a data appendix (yes/partial/no/NA) NA

3.4. All novel datasets introduced in this paper will be made publicly available upon publication of the paper with a license that allows free usage for research purposes (yes/partial/no/NA) NA

3.5. All datasets drawn from the existing literature (potentially including authors' own previously published work) are accompanied by appropriate citations (yes/no/NA) yes

3.6. All datasets drawn from the existing literature (potentially including authors' own previously published work) are publicly available (yes/partial/no/NA) yes

3.7. All datasets that are not publicly available are described in detail, with explanation why publicly available alternatives are not scientifically satisficing (yes/partial/no/NA) NA

4. Computational Experiments

4.1. Does this paper include computational experiments? (yes/no) yes

If yes, please address the following points:

4.2. This paper states the number and range of values tried per (hyper-) parameter during development of the paper, along with the criterion used for selecting the final parameter setting (yes/partial/no/NA) yes

4.3. Any code required for pre-processing data is included in the appendix (yes/partial/no) no

4.4. All source code required for conducting and analyzing the experiments is included in a code appendix (yes/partial/no) no

4.5. All source code required for conducting and analyzing the experiments will be made publicly available upon publication of the paper with a license that allows free usage for research purposes (yes/partial/no) yes

4.6. All source code implementing new methods have comments detailing the implementation, with references to the paper where each step comes from (yes/partial/no) yes

4.7. If an algorithm depends on randomness, then the method used for setting seeds is described in a way sufficient to allow replication of results (yes/partial/no/NA) yes

4.8. This paper specifies the computing infrastructure used for running experiments (hardware and software), including GPU/CPU models; amount of memory; operating system; names and versions of relevant software libraries and frameworks (yes/partial/no) yes

4.9. This paper formally describes evaluation metrics used and explains the motivation for choosing these metrics (yes/partial/no) yes

4.10. This paper states the number of algorithm runs used to compute each reported result (yes/no) yes

4.11. Analysis of experiments goes beyond single-dimensional summaries of performance (e.g., average; median) to include measures of variation, confidence, or other distributional information (yes/no) yes

4.12. The significance of any improvement or decrease in performance is judged using appropriate statistical tests (e.g., Wilcoxon signed-rank) (yes/partial/no) partial

4.13. This paper lists all final (hyper-)parameters used for each model/algorithm in the paper's experiments (yes/partial/no/NA) yes