Modulating Merging Strengths via Joint Loss Estimation for LoRA-based Continual Learning

Kun Gu[1] **De Cheng**[1,✉] **Zhipeng Xu**[1] **Lingfeng He**[1] **Di Xu**[2] **Nannan Wang**[1]

[1]Xidian University [2]Huawei Technologies Co., Ltd.

{kung@stu., dcheng@, xu_zhipeng@stu., lfhe@stu., nnwang@}xidian.edu.cn

xudi21@huawei.com

Abstract

Continual learning (CL) aims to adapt models to new tasks while preserving previously learned knowledge. Recently, Low-Rank Adaptation (LoRA), a representative Parameter-Efficient Fine-Tuning (PEFT) method, has gained increasing attention in CL due to its efficiency and scalability. Several LoRA-based merging methods maintain a single inference model by scaling the newly learned LoRA update with one coefficient and merging it into the previous weights after each task. However, they usually control the whole update with a shared scalar factor, although different update elements may affect previous and current tasks in different ways. To address this issue, we propose Merging Strength Modulation via Joint Loss Estimation (M^2LE). It consists of Hessian Information-guided Adaptive Merging (HIM) and Task Simulation Perturbation for Merging (TSP). Instead of searching for a global merge coefficient, HIM directly solves for the target merged update under a joint objective over previous and current tasks. Solving this objective gives an analytic solution that naturally takes the form of element-wise modulation over the current task LoRA update. To improve the stability of the learned update under subsequent merging, we introduce TSP which simulates different tasks interference. Experiments on multiple benchmarks show that M^2LE consistently outperforms existing PEFT-based continual learning methods. Code is available at https://github.com/bbdeeer/M2LE_NeurIPS2026.

1 Introduction

Continual learning (CL) aims to enable models to learn continuously from a stream of tasks without retraining from scratch or accessing data from previous tasks [11, 22], which is essential for evolving open-world scenarios. A central challenge in CL is catastrophic forgetting: models must preserve previously learned capabilities during adapting to new tasks [23]. In this work, we consider the challenging rehearsal-free Continual Learning setting, where the model must continually learn new tasks without storing past data and predict test samples over all seen tasks without task identities.

Recently, with the advancement of Pre-Trained Models (PTMs) [1], Parameter-Efficient Fine-Tuning (PEFT) methods have emerged as a promising solution for CL [28, 32, 39]. By updating a small number of paramters, PEFT-based methods enable efficient adaptation to new tasks while preserving general pre-trained knowledge, making it particularly suitable for rehearsal-free CL.

In the regime of PEFT-based CL, Low-Rank Adaptation (LoRA) [9] methods have gained increasing attention due to their efficiency and scalability. To maintain a single inference model, a common strategy is to merge the current task LoRA update into the accumulated model after each task [25], rather than storing a separate LoRA module per task. This controlled merging reduces interference with prior knowledge while enabling adaptation to the current task. Early studies mainly adopt EMA-style interpolation [17] or weight averaging [3], while recent approaches estimate the interpolation

✉Corresponding author

40th Conference on Neural Information Processing Systems (NeurIPS 2026).

coefficient from joint objectives [25] or Bayesian formulations [14]. Despite these different designs, most existing methods still use a shared scalar factor to control the whole LoRA update. This naturally raises a question: ***should all LoRA parameters share the same merge strength, or should different positions in the LoRA update be treated differently during merging?*** To answer this question, we conduct a preliminary analysis in Fig. 1, where the LoRA update is split into two groups according to different normalized magnitudes [10]. We then assign an independent merge coefficient to each group. The results show that this simple grouped merge already outperforms the best shared factor merge, suggesting that different parts of the LoRA update may prefer different merge strengths.

Building on this observation, we propose a Merging Strength Modulation via Joint Loss Estimation (M^2LE) framework for LoRA-based Continual Learning, consists of Hessian Information-guided Adaptive Merging (HIM) and Task Simulation Perturbation for Merging (TSP). Instead of searching for an interpolation coefficient, HIM directly formulates the optimization of the target merged update $\Delta\mathbf{W}_t^*$ under a joint objective that accounts for losses on all seen tasks. Through a local quadratic approximation to the task-wise losses, we obtain an analytic solution, which naturally takes the form of an element-wise modulation of the current task LoRA update. Specifically, each element is adaptively scaled by the associated Hessian information, allowing the merging process to account for the varying sensitivity of individual parameter elements across tasks, thereby achieving a more fine-grained balance between old and new task performance. However, once the current task update is merged into the inference model, it becomes part of the fixed parameters and will be affected by subsequent task updates. If the merged update lies in a sharp local region, later merging steps can easily increase the loss on old tasks, resulting in forgetting. Therefore, besides determining how to merge the update, it is also important to shape the update itself before merging. To this end, we introduce TSP, which simulates cross-task interference during current task training. TSP induces a local parameter change with one subset of the current mini-batch and evaluates the resulting loss variation on another subset, thereby encouraging the learned update to remain stable during subsequent merging.

Our main contributions can be summarized as follows:

- We propose Hessian Information-guided Adaptive Merging (HIM), a parameter-adaptive merging algorithm derived from an approximated joint loss objective, enabling more fine-grained stability-plasticity balancing beyond a shared scalar merging coefficient.
- We propose Task Simulation Perturbation (TSP), a training strategy that simulates future cross-task interference through local parameter perturbations induced from label-disjoint subset, thereby encouraging flat solutions and improving LoRA merging stability.
- Extensive experiments demonstrate the effectiveness of our proposed method, achieving superior performances over state-of-the-art PEFT-based CL methods on multiple benchmarks.

2 Related Work

PEFT-based and LoRA-based continual learning. PEFT-based CL typically keeps a pre-trained model frozen and introduces lightweight learnable modules for continual learning. L2P [39] retrieves instance specific prompts from a prompt pool, and SECA [6] uses textual semantic priors to support knowledge transfer and prototype refinement in CLIP-based continual learning. EWC [11] preserves previous knowledge by penalizing changes to important parameters, and EKPC [35] mitigates forgetting through elastic knowledge preservation and compensation. Among PEFT-based method, LoRA has become a representative choice for continual learning due to its parameter efficiency and compatibility with frozen pre-trained backbones [4, 9]. InfLoRA [15] constrains new LoRA updates to interference-free subspaces through orthogonality, PLAN [37] proactively allocates low-rank subspaces and optimizes them against worst-case perturbations, GainLoRA [16] expands a new LoRA for each task and introduces gating modules to adaptively weight the old and new branches, and MACIL [40] mitigates forgetting through semantic drift calibration. LoDA [5] continually decomposes LoRA updates into general and isolated subspaces for better stability-plasticity trade-off.

Model Merging. Model merging aims to integrate weights trained on different tasks into a single network [30]. RobustMerge [42] improves parameter-efficient merging by enhancing the robustness of low-rank update directions, while OPCM [33] enables sequential merging through orthogonal projection and adaptive scaling. This idea has also been used in continual learning, where the current model or update is merged into the inference model after each task [10]. ACMap [3] averages

the adapters learned from different tasks into a shared adapter, BECAME [14] derives an adaptive coefficient from a Bayesian formulation, and LoRA-P&M [25] combines the previous and current models using a shared merging factor computed in closed form. However, these methods mainly rely on a shared factor to interpolate updates, while our method derives the merged update from the loss changes of old and new tasks under a joint objective.

3 Preliminary Analysis

Research Question. *Should LoRA parameters at different positions be treated differently during merging, rather than being controlled by a shared factor?*

Experiment Setting. As a case study, we consider the 10 task setting on ImageNet-R and focus on the first merge step. Specifically, we take the model after learning task 0 and the update learned on task 1 for analysis. Let $\mathbf{W}_0$ denote the inference parameter after learning task 0, and let $\Delta\mathbf{W}_1$ denote the LoRA update learned on task 1. To this end, we build a simple merging setup. Specifically, we compute the normalized magnitude map of $\Delta\mathbf{W}_1$ as $\mathbf{R} = |\Delta\mathbf{W}_1| / \max_{i,j} |\Delta\mathbf{W}_1^{(i,j)}|$ [10], and divide the update into two groups: elements with normalized values below 0.5, and elements with normalized values at least 0.5. We then assign two independent coefficients, α_1 and α_2, to the two groups, and define the merged model as: $\mathbf{W}(\alpha_1, \alpha_2) = \mathbf{W}_0 + \alpha_1(\mathbf{M}_1 \odot \Delta\mathbf{W}_1) + \alpha_2(\mathbf{M}_2 \odot \Delta\mathbf{W}_1)$, where $\mathbf{M}_1$ and $\mathbf{M}_2$ denote the corresponding masks, and $\odot$ denotes element-wise multiplication. By varying α_1 and α_2, we evaluate the joint loss on task 0 and task 1.

Observation. Fig. 1 shows that the low loss region is not centered on the diagonal $\alpha_1 = \alpha_2$, but is shifted toward an off-diagonal region. Consistently, the green diamond marks the point with the best average accuracy on task0 and task1, which is located at $(0.40, 0.80)$, while the purple diamond marks the best scalar merge on the diagonal, located at $(0.60, 0.60)$. Compared with the scalar merge, the green-diamond point improves the average accuracy from 91.04% to 91.84%.

Insight. These results suggest that a shared scalar coefficient is overly restrictive: different parameter groups do not, in general, share the same preferred merge strength, and should therefore be treated differently during merging.

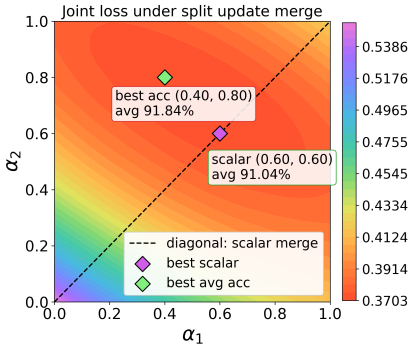


Figure 1: Joint loss landscape of the two group merge on 10S-ImageNet-R.

4 Methodology

4.1 Preliminaries

In CL, a model learns from a sequence of tasks $\{\mathcal{D}_t\}_{t=1}^{T}$. The dataset of task t is denoted by $\mathcal{D}_t = \{(x_t^i, y_t^i)\}_{i=1}^{n_t}$, where $x_t^i \in \mathbb{R}^d$ is the input and $y_t^i \in \mathcal{C}_t$ is the corresponding label. Each task introduces a set of new classes, and the label spaces of different tasks are mutually disjoint, i.e., $\mathcal{C}_i \cap \mathcal{C}_j = \varnothing$ for any $i \neq j$. Accordingly, the cumulative label space observed up to task t is $\mathcal{C}_{1:t} = \bigcup_{i=1}^{t} \mathcal{C}_i$. Under the standard rehearsal-free CL setting, only the current task data $\mathcal{D}_t$ is accessible when learning task t. The goal is to learn tasks sequentially while maintaining strong performance over all tasks observed so far.

4.2 Overall framework

Let $\mathbf{W}_{t-1} \in \mathbb{R}^{d_o \times d_i}$ denote the inference parameter after learning task $t-1$, where d_i and d_o are the input and output dimensions. At t-th task, for each ViT layers, we freeze the backbone weights $\mathbf{W}_{t-1}$ and attach a LoRA module. The LoRA update is parameterized by two low-rank matrices:

$$\Delta\mathbf{W}_t = \mathbf{B}_t\mathbf{A}_t, \quad \mathbf{A}_t \in \mathbb{R}^{r \times d_i}, \quad \mathbf{B}_t \in \mathbb{R}^{d_o \times r}, \tag{1}$$

where r is the LoRA rank. A common LoRA merging strategy directly updates the inference model by scaling the current task update with a shared coefficient $\mathbf{W}_t = \mathbf{W}_{t-1} + \alpha_t\Delta\mathbf{W}_t$, where the scalar α_t controls the merge strength of the whole update. However, this scalar merge applies the same

merge strength to all elements in $\Delta\mathbf{W}_t$, although different elements may have different effects on previous and current tasks.

To address this, M^{2}LE consists of two components: Hessian Information-guided Adaptive Merging (HIM) and Task Simulation Perturbation for Merging (TSP). During training, TSP regularizes the LoRA update by simulating cross task perturbations, encouraging the learned update $\Delta\mathbf{W}_t$ to improve the stability during subsequent merging. After training, HIM derives a target merged update $\Delta\mathbf{W}_t^*$ under a joint loss impact objective over previous and current tasks, instead of searching for a global merge coefficient. The inference parameter is then updated by $\mathbf{W}_t = \mathbf{W}_{t-1} + \Delta\mathbf{W}_t^*$. An overview of the framework and the complete training procedure are provided in Appendix A (Figure 5 and Algorithm 1).

4.3 Hessian Information-guided Adaptive Merging

Existing LoRA-based merging methods for CL always merge the current task LoRA update into the inference model using a shared scalar factor. This forces all parameter elements to be merged with the same strength, although they may affect previous and current tasks differently. This motivates us to account for the different preferred merge strengths of update parameter during merging.

Specifically, we directly solve for the target merged update $\Delta\mathbf{W}_t^* \in \mathbb{R}^{d_o \times d_i}$ under a joint objective over previous and current tasks. Intuitively, we want this update to incorporate new knowledge as effectively as possible while reducing catastrophic forgetting on previous tasks. To this end, we first define $\mathcal{J}_{\text{old}}$ and $\mathcal{J}_{\text{new}}$ to measure how an ideal merged update $\Delta\hat{\mathbf{W}}_t \in \mathbb{R}^{d_o \times d_i}$ affects the losses on previous and current tasks:

$$\mathcal{J}_{\text{old}}(\Delta\hat{\mathbf{W}}_t) = \sum_{i=1}^{t-1} \Big(\mathcal{L}_i(\mathbf{W}_{t-1} + \Delta\hat{\mathbf{W}}_t) - \mathcal{L}_i(\mathbf{W}_{t-1}) \Big), \tag{2}$$

$$\mathcal{J}_{\text{new}}(\Delta\hat{\mathbf{W}}_t) = \mathcal{L}_t(\mathbf{W}_{t-1} + \Delta\hat{\mathbf{W}}_t) - \mathcal{L}_t(\mathbf{W}_{t-1} + \Delta\mathbf{W}_t). \tag{3}$$

Here, $\Delta\mathbf{W}_t \in \mathbb{R}^{d_o \times d_i}$ denotes the update learned for task t, $\mathcal{L}_i$ denotes the loss of task i. Therefore, we seek a merged update that jointly minimizes $\mathcal{J}_{\text{old}}$ on previous tasks and $\mathcal{J}_{\text{new}}$ on the current task, with a hyper-parameter λ controlling the strength of the current task term. The target merged update $\Delta\mathbf{W}_t^*$ is obtained by solving:

$$\Delta\mathbf{W}_t^* = \arg\min_{\Delta\hat{\mathbf{W}}_t} \ \mathcal{J}_{\text{old}}(\Delta\hat{\mathbf{W}}_t) + \lambda\mathcal{J}_{\text{new}}(\Delta\hat{\mathbf{W}}_t). \tag{4}$$

In the rehearsal-free setting, Eq. (4) cannot be optimized directly, since the old-task term $\mathcal{J}_{\text{old}}$ depends on previous task data that are no longer accessible when learning task t. To obtain a tractable approximation, we approximate Eq. (2) and Eq. (3) with local second-order Taylor expansions. For notational simplicity, we vectorize $\Delta\hat{\mathbf{W}}_t$ into $\Delta\hat{\mathbf{w}}_t = \text{vec}(\Delta\hat{\mathbf{W}}_t) \in \mathbb{R}^{d_o d_i \times 1}$. For the old task term $\mathcal{J}_{\text{old}}(\Delta\hat{\mathbf{w}}_t)$, expanding around $\mathbf{w}_{t-1}$ gives:

$$\mathcal{J}_{\text{old}}(\Delta\hat{\mathbf{w}}_t) \approx \sum_{i=1}^{t-1} \left(\mathbf{g}_i^\top \Delta\hat{\mathbf{w}}_t + \frac{1}{2} \Delta\hat{\mathbf{w}}_t{}^\top \mathbf{H}_i \Delta\hat{\mathbf{w}}_t \right), \tag{5}$$

where $\mathbf{g}_i \in \mathbb{R}^{d_o d_i \times 1}$ and $\mathbf{H}_i \in \mathbb{R}^{d_o d_i \times d_o d_i}$ denote the first and second terms of the i-th old task loss around $\mathbf{w}_{t-1}$, respectively. Similarly, for the current task, expanding around $\mathbf{w}_{t-1} + \Delta\mathbf{w}_t$ yields:

$$\mathcal{J}_{\text{new}}(\Delta\hat{\mathbf{w}}_t) \approx \mathbf{g}_t^\top (\Delta\hat{\mathbf{w}}_t - \Delta\mathbf{w}_t) + \frac{1}{2} (\Delta\hat{\mathbf{w}}_t - \Delta\mathbf{w}_t)^\top \mathbf{H}_t (\Delta\hat{\mathbf{w}}_t - \Delta\mathbf{w}_t), \tag{6}$$

where $\mathbf{g}_t$ and $\mathbf{H}_t$ denote the first- and second-order terms of the current task loss around $\mathbf{w}_{t-1} + \Delta\mathbf{w}_t$, respectively. Since the previous model $\mathbf{w}_{t-1}$ is obtained by recursively merging past updates and $\mathbf{w}_{t-1} + \Delta\mathbf{w}_t$ is the learned current-task solution. Under a local approximation, we treat the first-order terms in Eq. (5) and (6) as approximately zero. Substituting the remaining quadratic terms into Eq. (4), we obtain:

$$\Delta\mathbf{W}_t^* = \arg\min_{\Delta\hat{\mathbf{w}}_t} \quad \sum_{i=1}^{t-1} \frac{1}{2} \Delta\hat{\mathbf{w}}_t{}^\top \mathbf{H}_i \Delta\hat{\mathbf{w}}_t + \frac{\lambda}{2} (\Delta\hat{\mathbf{w}}_t - \Delta\mathbf{w}_t)^\top \mathbf{H}_t (\Delta\hat{\mathbf{w}}_t - \Delta\mathbf{w}_t). \tag{7}$$

Taking the derivative with respect to $\Delta\hat{\mathbf{w}}_t$ and setting it to zero gives:

$$\left(\sum_{i=1}^{t-1} \mathbf{H}_i + \lambda \mathbf{H}_t \right) \Delta\mathbf{w}_t^* = \lambda \mathbf{H}_t \Delta\mathbf{w}_t. \tag{8}$$

A detailed derivation is provided in Appendix B, and Appendix C.4 further analyzes the effectiveness of accumulated historical Fisher statistics. Following EWC [11], we approximate each Hessian in Eq. (8) with the diagonal empirical Fisher, reusing their historical running sum for the old task term. For task t, the i-th diagonal element of $\mathbf{F}_t$ at the learned parameters $\mathbf{W}_{t-1} + \Delta\mathbf{W}_t$ is defined as $\mathbf{F}_t^{i,i} = \mathbb{E}_{(x,y)\sim\mathcal{D}_t} \left[\left(\frac{\partial \mathcal{L}(f_{\mathbf{W}}(x), y)}{\partial w_i} \Big|_{\mathbf{W}=\mathbf{W}_{t-1}+\Delta\mathbf{W}_t} \right)^2 \right]$. Here, $\mathbf{F}_t \in \mathbb{R}^{d_o \times d_i}$ denotes the diagonal Fisher of the trainable update, reshaped to match $\Delta\mathbf{W}_t$. Under this diagonal approximation, Eq. (8) gives the following element-wise merged update:

$$\Delta\mathbf{W}_t^* = \frac{\lambda \mathbf{F}_t}{\mathbf{F}_{1:t-1} + \lambda \mathbf{F}_t + \varepsilon} \odot \Delta\mathbf{W}_t, \tag{9}$$

where ε is a small damping constant for numerical stability and after merging task t, we update the accumulated Fisher by $\mathbf{F}_{1:t} = \mathbf{F}_{1:t-1} + \mathbf{F}_t$, which is used in the next merging step.

4.4 Task Simulation Perturbation for Merging

According to Eq. (9), HIM obtains $\Delta\mathbf{W}_t^*$ by element-wise reweighting $\Delta\mathbf{W}_t$ and merge it into the inference model, where it becomes part of the fixed parameters for later tasks. If the merged update lies in a sharp region, subsequent new task merging can easily increase its loss and lead to forgetting. This naturally connects to the flat-minimum principle, making SAM [2] a straightforward way to improve the local stability of the current task update. However, SAM computes the perturbation from the overall current task loss and minimizes the perturbed loss on the same task, so its flatness constraint remains limited to the current task. Such a design makes it difficult to approximate the case in CL where the update from one task affects another task. To approximating cross-task interference when old task data are unavailable. We randomly split each mini-batch into two label-disjoint subsets $(\mathcal{S}_1, \mathcal{S}_2)$, using one subset to induce a local parameter change and the other to evaluate the resulting loss change.

For the $\mathcal{S}_1 \to \mathcal{S}_2$ direction, we first compute the gradient on the source subset $\mathcal{S}_1$:

$$\mathbf{g}_1 = \nabla\mathcal{L}(\mathcal{S}_1). \tag{10}$$

We then construct a probing perturbation along the induced descent direction:

$$\epsilon_1 = -\rho \frac{\mathbf{g}_1}{\|\mathbf{g}_1\|_2^2 + \varepsilon}, \tag{11}$$

where ρ controls the perturbation scale and ε is a small constant for numerical stability. The target subset $\mathcal{S}_2$ is evaluated under both the perturbed and original parameters:

$$\mathcal{L}_{\text{pert}}^{1\to2} = \mathcal{L}(\mathcal{S}_2; \mathbf{B}_t\mathbf{A}_t + \epsilon_1), \qquad \mathcal{L}_{\text{clean}}^{1\to2} = \mathcal{L}(\mathcal{S}_2; \mathbf{B}_t\mathbf{A}_t). \tag{12}$$

The perturbed loss evaluates whether the direction induced by $\mathcal{S}_1$ remains effective on the disjoint subset $\mathcal{S}_2$ under a local displacement. To prevent the perturbation from pulling training too far away from the normal update and degrading the clean loss on $\mathcal{S}_2$, we retain the clean loss as an anchor and define the TSP objective as:

$$\mathcal{L}_{\text{TSP}}^{1\to2} = \beta\mathcal{L}_{\text{pert}}^{1\to2} + (1-\beta)\mathcal{L}_{\text{clean}}^{1\to2}, \tag{13}$$

where β balances the perturbed and clean terms. We first update the parameters for the $\mathcal{S}_1 \to \mathcal{S}_2$ step by:

$$\mathbf{B}_t'\mathbf{A}_t' = \mathbf{B}_t\mathbf{A}_t - \eta\nabla\mathcal{L}_{\text{TSP}}^{1\to2}(\mathbf{B}_t\mathbf{A}_t), \tag{14}$$

Table 1: Experimental results for 10 incremental tasks on ImageNet-R, ImageNet-A, CIFAR100, and CUB. We report the averaged results over 3 random seeds. Within each group, the highest results are in **bold**, and the second highest results are underlined. "FR" denotes the use of feature replay.

Method	Venue	FR	10S-ImageNet-R		10S-ImageNet-A		10S-CIFAR100		10S-CUB	
			$\mathcal{A}_{Last}\uparrow$	$\mathcal{A}_{Avg}\uparrow$	$\mathcal{A}_{Last}\uparrow$	$\mathcal{A}_{Avg}\uparrow$	$\mathcal{A}_{Last}\uparrow$	$\mathcal{A}_{Avg}\uparrow$	$\mathcal{A}_{Last}\uparrow$	$\mathcal{A}_{Avg}\uparrow$
L2P [39]	CVPR'22	✗	$72.34_{\pm 0.17}$	$77.36_{\pm 0.64}$	$44.04_{\pm 0.93}$	$51.24_{\pm 2.26}$	$84.06_{\pm 0.88}$	$88.26_{\pm 1.34}$	$67.02_{\pm 1.90}$	$70.62_{\pm 1.60}$
CODAPrompt [28]	CVPR'23	✗	$73.31_{\pm 0.50}$	$78.47_{\pm 0.53}$	$52.08_{\pm 0.12}$	$63.92_{\pm 0.12}$	$83.21_{\pm 3.39}$	$87.71_{\pm 3.17}$	$77.23_{\pm 1.12}$	$81.90_{\pm 0.85}$
InfLoRA [15]	CVPR'24	✗	$74.75_{\pm 0.64}$	$80.67_{\pm 0.55}$	$49.20_{\pm 1.12}$	$60.92_{\pm 0.61}$	$86.75_{\pm 0.35}$	$91.72_{\pm 0.15}$	$70.82_{\pm 0.23}$	$81.39_{\pm 0.14}$
PLAN [37]	ICCV'25	✗	$75.25_{\pm 0.42}$	$80.41_{\pm 0.56}$	–	–	$87.54_{\pm 0.31}$	$92.21_{\pm 0.35}$	–	–
SEMA [36]	CVPR'25	✗	78.00	83.56	53.32	64.53	86.98	91.37	–	–
SD-LoRA [41]	ICLR'25	✗	$77.34_{\pm 0.35}$	$82.04_{\pm 0.24}$	$55.96_{\pm 0.73}$	$64.95_{\pm 1.63}$	$88.01_{\pm 0.31}$	$92.54_{\pm 0.18}$	$77.48_{\pm 0.20}$	$\underline{85.59}_{\pm 0.44}$
BiLoRA [48]	CVPR'25	✗	$77.95_{\pm 0.14}$	$81.52_{\pm 0.26}$	–	–	$87.46_{\pm 0.76}$	$92.50_{\pm 0.62}$	–	–
LoRA-P&M [25]	NIPS'25	✗	$\underline{79.95}_{\pm 0.18}$	$\underline{85.29}_{\pm 0.93}$	$56.57_{\pm 0.78}$	$65.35_{\pm 1.81}$	$\underline{88.45}_{\pm 0.35}$	$\underline{92.89}_{\pm 1.13}$	$\underline{78.29}_{\pm 0.50}$	$83.39_{\pm 0.61}$
EWC-LoRA [44]	ICLR'26	✗	$72.86_{\pm 0.79}$	$78.95_{\pm 0.86}$	$\underline{59.89}_{\pm 0.26}$	$\underline{68.33}_{\pm 0.67}$	$87.91_{\pm 0.57}$	$92.27_{\pm 0.39}$	–	–
M^{2}LE (Ours)	–	✗	$\mathbf{81.34_{\pm 0.16}}$	$\mathbf{86.53_{\pm 0.51}}$	$\mathbf{62.45_{\pm 1.34}}$	$\mathbf{71.02_{\pm 1.82}}$	$\mathbf{89.97_{\pm 0.04}}$	$\mathbf{93.31_{\pm 0.95}}$	$\mathbf{81.40_{\pm 0.21}}$	$\mathbf{89.23_{\pm 0.48}}$
SLCA [43]	ICCV'23	✓	$79.35_{\pm 0.28}$	$83.29_{\pm 0.46}$	$61.05_{\pm 0.63}$	$68.88_{\pm 2.31}$	$91.26_{\pm 0.37}$	$94.29_{\pm 0.92}$	$84.68_{\pm 0.09}$	$90.77_{\pm 0.79}$
SSIAT [32]	CVPR'24	✓	$79.38_{\pm 0.59}$	$83.63_{\pm 0.43}$	$62.43_{\pm 1.63}$	$70.83_{\pm 1.63}$	$91.35_{\pm 0.26}$	$94.35_{\pm 0.60}$	$88.75_{\pm 0.38}$	$93.00_{\pm 0.90}$
MOS [31]	AAAI'25	✓	$77.68_{\pm 0.41}$	$82.06_{\pm 0.53}$	$54.75_{\pm 1.09}$	$63.32_{\pm 2.38}$	$91.53_{\pm 0.35}$	$94.21_{\pm 0.91}$	$89.97_{\pm 0.32}$	$93.43_{\pm 0.60}$
MACIL [40]	ICML'25	✓	$\underline{81.88}_{\pm 0.07}$	$\underline{85.95}_{\pm 0.27}$	$64.14_{\pm 0.58}$	$71.45_{\pm 1.35}$	$\mathbf{91.94_{\pm 0.17}}$	$94.43_{\pm 0.79}$	$\mathbf{90.52_{\pm 0.13}}$	$\mathbf{93.93_{\pm 0.47}}$
SAEF [18]	CVPR'26	✓	80.02	84.54	$\underline{64.91}$	$\underline{73.39}$	90.60	$\underline{94.53}$	–	–
M^{2}LE (Ours) + CA	–	✓	$\mathbf{82.68_{\pm 0.30}}$	$\mathbf{87.05_{\pm 0.51}}$	$\mathbf{65.96_{\pm 0.90}}$	$\mathbf{74.02_{\pm 0.85}}$	$\underline{91.86}_{\pm 0.21}$	$\mathbf{94.62_{\pm 0.90}}$	$\underline{90.35}_{\pm 0.16}$	$\underline{93.92}_{\pm 0.38}$

Table 2: Additional results under different task settings, including 5 and 20 tasks on ImageNet-R, 20 tasks on ImageNet-A, and 10 tasks on DomainNet. Within each group, the highest results are in **bold**, and the second highest results are underlined. "FR" denotes the use of feature replay.

Method	Venue	FR	5S-ImageNet-R		20S-ImageNet-R		20S-ImageNet-A		10S-DomainNet	
			$\mathcal{A}_{Last}\uparrow$	$\mathcal{A}_{Avg}\uparrow$	$\mathcal{A}_{Last}\uparrow$	$\mathcal{A}_{Avg}\uparrow$	$\mathcal{A}_{Last}\uparrow$	$\mathcal{A}_{Avg}\uparrow$	$\mathcal{A}_{Last}\uparrow$	$\mathcal{A}_{Avg}\uparrow$
L2P [39]	CVPR'22	✗	$70.83_{\pm 0.58}$	$78.34_{\pm 0.47}$	$69.64_{\pm 0.42}$	$75.28_{\pm 0.57}$	$40.48_{\pm 1.78}$	$49.62_{\pm 1.46}$	$81.17_{\pm 0.83}$	$87.43_{\pm 0.95}$
CODAPrompt [28]	CVPR'23	✗	$74.91_{\pm 0.33}$	$79.25_{\pm 0.53}$	$69.96_{\pm 0.50}$	$75.34_{\pm 0.85}$	$44.62_{\pm 1.92}$	$54.86_{\pm 0.50}$	$80.04_{\pm 0.79}$	$86.27_{\pm 0.82}$
InfLoRA [15]	CVPR'24	✗	$77.52_{\pm 0.37}$	$82.01_{\pm 0.12}$	$71.01_{\pm 0.45}$	$77.28_{\pm 0.45}$	$46.81_{\pm 2.30}$	$57.13_{\pm 0.95}$	$81.45_{\pm 0.68}$	$87.55_{\pm 0.57}$
SEMA [36]	CVPR'25	✗	79.78	84.75	74.53	81.75	–	–	–	–
SD-LoRA [41]	ICLR'25	✗	$79.15_{\pm 0.20}$	$83.01_{\pm 0.42}$	$75.26_{\pm 0.37}$	$80.22_{\pm 0.72}$	$49.12_{\pm 1.78}$	$59.63_{\pm 1.57}$	$81.01_{\pm 0.42}$	$86.85_{\pm 0.40}$
BiLoRA [48]	CVPR'25	✗	$78.32_{\pm 0.37}$	$83.31_{\pm 0.52}$	$72.41_{\pm 0.65}$	$79.28_{\pm 0.76}$	–	–	$76.56_{\pm 0.41}$	$83.20_{\pm 0.35}$
LoRA-P&M [25]	NIPS'25	✗	$\underline{81.47}_{\pm 0.56}$	$\underline{85.96}_{\pm 0.52}$	$\underline{76.37}_{\pm 0.09}$	$\underline{82.77}_{\pm 0.71}$	$\underline{52.27}_{\pm 1.84}$	$\underline{62.15}_{\pm 1.33}$	$\underline{84.35}_{\pm 0.48}$	$\underline{89.43}_{\pm 0.27}$
EWC-LoRA [44]	ICLR'26	✗	$76.36_{\pm 0.21}$	$81.43_{\pm 0.13}$	$70.18_{\pm 1.06}$	$77.06_{\pm 0.54}$	–	–	–	–
M^{2}LE (Ours)	–	✗	$\mathbf{82.67_{\pm 0.18}}$	$\mathbf{87.31_{\pm 0.35}}$	$\mathbf{78.62_{\pm 0.43}}$	$\mathbf{84.62_{\pm 0.26}}$	$\mathbf{53.87_{\pm 1.45}}$	$\mathbf{63.69_{\pm 1.87}}$	$\mathbf{84.78_{\pm 0.13}}$	$\mathbf{89.74_{\pm 0.44}}$
SLCA [43]	ICCV'23	✓	$81.01_{\pm 0.11}$	$84.18_{\pm 0.28}$	$74.63_{\pm 1.55}$	$79.92_{\pm 1.29}$	$36.69_{\pm 21.31}$	$56.35_{\pm 7.09}$	$84.35_{\pm 0.72}$	$88.56_{\pm 0.21}$
SSIAT [32]	CVPR'24	✓	$80.52_{\pm 0.07}$	$84.25_{\pm 0.31}$	$75.67_{\pm 0.24}$	$82.30_{\pm 0.36}$	$59.16_{\pm 1.03}$	$\underline{68.45}_{\pm 1.92}$	$85.11_{\pm 0.56}$	$89.80_{\pm 0.34}$
MOS [31]	AAAI'25	✓	$78.76_{\pm 0.17}$	$82.37_{\pm 0.26}$	$75.16_{\pm 0.59}$	$80.53_{\pm 0.70}$	–	–	–	–
MACIL [40]	ICML'25	✓	$\mathbf{83.37_{\pm 0.26}}$	$\underline{87.00}_{\pm 0.35}$	$\underline{79.43}_{\pm 0.34}$	$\underline{84.34}_{\pm 0.32}$	$\underline{59.36}_{\pm 1.27}$	$68.03_{\pm 2.00}$	$\underline{86.62}_{\pm 0.35}$	$\underline{90.88}_{\pm 0.20}$
M^{2}LE (Ours) + CA	–	✓	$\underline{82.74}_{\pm 0.29}$	$\mathbf{87.25_{\pm 0.28}}$	$\mathbf{81.12_{\pm 0.14}}$	$\mathbf{86.12_{\pm 0.33}}$	$\mathbf{64.25_{\pm 0.56}}$	$\mathbf{72.54_{\pm 0.92}}$	$\mathbf{87.61_{\pm 0.13}}$	$\mathbf{91.51_{\pm 0.18}}$

where η is the learning rate. We then exchange the roles of the two subsets and perform the reverse update:

$$\mathbf{B}_t''\mathbf{A}_t'' = \mathbf{B}_t'\mathbf{A}_t' - \eta\nabla\mathcal{L}_{\text{TSP}}^{2\to1}(\mathbf{B}_t'\mathbf{A}_t'), \tag{15}$$

where $\mathcal{L}_{\text{TSP}}^{2\to1}$ is defined in the same way, with the roles of $\mathcal{S}_1$ and $\mathcal{S}_2$ exchanged. Thus, within each mini-batch, the optimization proceeds sequentially as $\mathbf{B}_t\mathbf{A}_t \xrightarrow{\mathcal{S}_1\to\mathcal{S}_2} \mathbf{B}_t'\mathbf{A}_t' \xrightarrow{\mathcal{S}_2\to\mathcal{S}_1} \mathbf{B}_t''\mathbf{A}_t''$. This dual training framework learns the current task update under task simulating perturbations, so that the subsequent merge better preserves previously learned knowledge.

5 Experiment

Datasets. We evaluate our method on five standard continual learning benchmarks: ImageNet-R [7], ImageNet-A [8], CIFAR100 [13], CUB200 [34], and DomainNet [24]. ImageNet-R contains 30,000 images from 200 classes with diverse styles, while ImageNet-A consists of 7,500 natural adversarial examples spanning 200 classes. CIFAR100 is a standard benchmark with 60,000 images from 100 classes, and CUB200 is a fine-grained bird classification dataset with 200 categories. For DomainNet, following prior works [5], we use the commonly adopted 200-class evaluation protocol.

Evaluation Metrics. Following prior PEFT-based continual learning works [32, 40], we adopt two standard evaluation metrics: (1) $\mathcal{A}_{Last}$, i.e., the final accuracy over all classes after the last task, and (2) $\mathcal{A}_{Avg}$, i.e., the average accuracy across all incremental tasks.

Implementation Details. We use ViT-B/16 [1] pre-trained on ImageNet-21K [27] as the backbone. All experiments are conducted on a single NVIDIA RTX 3090 GPU. The model is optimized with SGD and a Cosine Annealing scheduler following [40]. We train each task for 20 epochs with a batch size of 48. The initial learning rate is set to 0.02 on ImageNet-R and 0.01 on the other datasets. For

Table 3: Ablation studies on ImageNet-R and ImageNet-A under 10 and 20 incremental tasks.

Idx	HIM	TSP	10S-ImageNet-R		10S-ImageNet-A		20S-ImageNet-R		20S-ImageNet-A	
			$\mathcal{A}_{Last}\uparrow$	$\mathcal{A}_{Avg}\uparrow$	$\mathcal{A}_{Last}\uparrow$	$\mathcal{A}_{Avg}\uparrow$	$\mathcal{A}_{Last}\uparrow$	$\mathcal{A}_{Avg}\uparrow$	$\mathcal{A}_{Last}\uparrow$	$\mathcal{A}_{Avg}\uparrow$
1	–	–	$73.05_{\pm 0.35}$	$80.40_{\pm 0.60}$	$52.56_{\pm 2.08}$	$63.36_{\pm 1.32}$	$66.14_{\pm 0.20}$	$73.72_{\pm 0.25}$	$46.06_{\pm 2.14}$	$59.43_{\pm 1.77}$
2	✓	–	$80.36_{\pm 0.40}$	$85.78_{\pm 0.44}$	$61.35_{\pm 1.06}$	$70.23_{\pm 2.04}$	$78.04_{\pm 0.35}$	$84.31_{\pm 0.54}$	$52.80_{\pm 1.46}$	$62.64_{\pm 2.14}$
3	–	✓	$74.20_{\pm 0.91}$	$80.67_{\pm 0.47}$	$56.57_{\pm 2.31}$	$66.26_{\pm 1.46}$	$69.17_{\pm 0.10}$	$77.16_{\pm 0.39}$	$49.50_{\pm 0.72}$	$60.45_{\pm 1.52}$
4	✓	✓	$\mathbf{81.34_{\pm 0.16}}$	$\mathbf{86.53_{\pm 0.51}}$	$\mathbf{62.45_{\pm 1.34}}$	$\mathbf{71.02_{\pm 1.82}}$	$\mathbf{78.62_{\pm 0.43}}$	$\mathbf{84.62_{\pm 0.26}}$	$\mathbf{53.87_{\pm 1.45}}$	$\mathbf{63.69_{\pm 1.87}}$

Table 4: Comparison of different merge strategies on 10S-ImageNet-R and 10S-ImageNet-A.

Strategy	10S-ImageNet-R		10S-ImageNet-A	
	$\mathcal{A}_{Last}\uparrow$	$\mathcal{A}_{Avg}\uparrow$	$\mathcal{A}_{Last}\uparrow$	$\mathcal{A}_{Avg}\uparrow$
Seq-FT	$73.05_{\pm 0.35}$	$80.40_{\pm 0.60}$	$52.56_{\pm 2.08}$	$63.36_{\pm 1.32}$
EMA-style ($\theta = \frac{t-1}{t}\theta_{t-1} + \frac{1}{t}\theta_t$) [17]	$79.14_{\pm 0.10}$	$85.08_{\pm 0.41}$	$59.64_{\pm 1.22}$	$69.52_{\pm 1.82}$
LoRA-P&M [25]	$78.89_{\pm 0.44}$	$84.99_{\pm 0.45}$	$59.93_{\pm 1.57}$	$69.87_{\pm 2.05}$
HIM-last ($\mathcal{J}_{old} = \mathcal{L}_{t-1}(\mathbf{W}_{t-1} + \Delta\hat{\mathbf{W}}_t) - \mathcal{L}_{t-1}(\mathbf{W}_{t-1})$)	$76.23_{\pm 0.76}$	$83.81_{\pm 0.61}$	$58.17_{\pm 2.38}$	$68.84_{\pm 1.93}$
HIM (Ours)	$\mathbf{80.36_{\pm 0.40}}$	$\mathbf{85.78_{\pm 0.44}}$	$\mathbf{61.35_{\pm 1.06}}$	$\mathbf{70.23_{\pm 2.04}}$

the HIM, the hyper-parameter λ in Eq. (9) is set to 5 for all datasets. For the TSP, ρ in Eq. (11) is set to 0.3 on ImageNet-A and 0.5 on the other datasets, while β in Eq. (13) is set to 0.6 on ImageNet-A and 0.8 on the other datasets.

Variant with classifier alignment. For fair comparison with methods that use feature replay or classifier alignment (CA), we also evaluate an enhanced variant, denoted as M^{2}LE+CA. M^{2}LE follows the end-to-end continual learning protocol without feature replay or post-hoc classifier adjustment. M^{2}LE+CA further applies the widely used classifier alignment strategy after each task, following SLCA [43]. We report these two variants separately to distinguish the effect of our proposed method from the additional classifier alignment stage.

5.1 Comparison with the State-of-the-Arts

We compare our method with representative state-of-the-art continual learning approaches from three categories, including prompt-based methods [28, 39], adapter-based methods [18, 31, 32, 36] and LoRA-based methods [15, 25, 37, 40, 41, 44, 48]. For completeness, methods that rely on feature replay or classifier alignment are reported separately from strictly rehearsal-free methods.

As shown in Tables 1 and 2, our method demonstrates clear advantages over existing PEFT-based continual learning methods. Without classifier alignment, compared with the LoRA-P&M, M^{2}LE consistently improves $\mathcal{A}_{Last}$ across all settings, with gains of 1.39%–5.88% on the 10 task benchmarks and 0.43%–2.25% under the additional task settings. With classifier alignment, M^{2}LE+CA achieves highly competitive performance against feature-replay-based methods, outperforming MACIL in $\mathcal{A}_{Last}$ by 0.80%–1.82% on the 10 task ImageNet-R/ImageNet-A settings and by 1.69%–4.89% on their 20 task settings. The margins are relatively smaller on CIFAR100 and CUB, where strong pre-trained representations already provide discriminative features and thus reduce the need for further backbone adaptation.

5.2 Ablation Studies

We conduct ablation studies on the main components of our training pipeline, including the sequential fine-tuning baseline, the proposed HIM, and TSP. As shown in Tab. 3, HIM (Idx 2) brings the main improvements over the baseline (Idx 1), increasing $\mathcal{A}_{Last}$ by 6.74%–11.90% across all four settings. This shows that reweighting the current task update under the joint loss impact objective in Eq. (4) is more effective than directly adding it to the previous inference model. TSP (Idx 3) also consistently improves the baseline, with $\mathcal{A}_{Last}$ gains of 1.15%–4.01%. The improvement is particularly clear on the more challenging ImageNet-A benchmark, suggesting that TSP helps the current task update remain stable under task simulating perturbations, making it less sensitive to subsequent merge-induced changes. Notably, combining HIM and TSP (Idx 4) further improves $\mathcal{A}_{Last}$ by 0.58%–1.10% over HIM alone and achieves the best overall performance across all settings, indicating that the two components play complementary roles.

Table 5: Comparison of different perturbation strategies on 10S-ImageNet-R and 10S-ImageNet-A.

Strategy	10S-ImageNet-R		10S-ImageNet-A	
	$\mathcal{A}_{Last}\uparrow$	$\mathcal{A}_{Avg}\uparrow$	$\mathcal{A}_{Last}\uparrow$	$\mathcal{A}_{Avg}\uparrow$
w/o TSP	$80.36_{\pm 0.40}$	$85.78_{\pm 0.44}$	$61.35_{\pm 1.06}$	$70.23_{\pm 2.04}$
SAM [2]	$79.55_{\pm 0.17}$	$84.92_{\pm 0.44}$	$60.46_{\pm 1.51}$	$69.00_{\pm 1.98}$
TSP (Ours)	$\mathbf{81.34}_{\pm \mathbf{0.16}}$	$\mathbf{86.53}_{\pm \mathbf{0.51}}$	$\mathbf{62.45}_{\pm \mathbf{1.34}}$	$\mathbf{71.02}_{\pm \mathbf{1.82}}$

Table 6: Task-by-task results (%) on ImageNet-R and ImageNet-A under the 10 task setting. We compare the results with and without TSP. The forgetting metric is defined in Appendix C.1

Dataset	Metrics	Methods	Task 0	Task 1	Task 2	Task 3	Task 4	Task 5	Task 6	Task 7	Task 8	Task 9
ImageNet-R	Accuracy ↑	w/o TSP	$94.96_{\pm 1.23}$	$91.31_{\pm 1.22}$	$89.48_{\pm 0.20}$	$86.91_{\pm 0.67}$	$84.73_{\pm 0.64}$	$83.68_{\pm 0.20}$	$83.00_{\pm 0.61}$	$82.13_{\pm 0.53}$	$81.24_{\pm 0.56}$	$80.36_{\pm 0.40}$
		w/ TSP	$\mathbf{95.14}_{\pm 1.07}$	$\mathbf{92.14}_{\pm 0.86}$	$\mathbf{89.93}_{\pm 0.85}$	$\mathbf{87.58}_{\pm 0.56}$	$\mathbf{85.86}_{\pm 0.91}$	$\mathbf{84.49}_{\pm 0.61}$	$\mathbf{83.67}_{\pm 0.42}$	$\mathbf{82.98}_{\pm 0.40}$	$\mathbf{82.14}_{\pm 0.54}$	$\mathbf{81.34}_{\pm 0.16}$
		Δ	+0.18	+0.83	+0.45	+0.67	+1.13	+0.81	+0.67	+0.85	+0.90	+0.98
	Forgetting ↓	w/o TSP	–	$4.98_{\pm 0.90}$	$4.87_{\pm 0.60}$	$6.16_{\pm 0.58}$	$7.00_{\pm 0.59}$	$7.34_{\pm 0.20}$	$7.26_{\pm 0.67}$	$7.34_{\pm 0.49}$	$7.43_{\pm 0.44}$	$7.76_{\pm 0.38}$
		w/ TSP	–	$\mathbf{3.17}_{\pm 0.60}$	$\mathbf{3.79}_{\pm 0.86}$	$\mathbf{4.40}_{\pm 0.58}$	$\mathbf{4.89}_{\pm 0.78}$	$\mathbf{5.41}_{\pm 0.24}$	$\mathbf{5.46}_{\pm 0.62}$	$\mathbf{5.49}_{\pm 0.88}$	$\mathbf{5.52}_{\pm 0.82}$	$\mathbf{5.77}_{\pm 0.64}$
		Δ	–	−1.81	−1.08	−1.76	−2.11	−1.93	−1.80	−1.85	−1.91	−1.99
ImageNet-A	Accuracy ↑	w/o TSP	$86.56_{\pm 1.34}$	$80.06_{\pm 1.70}$	$74.96_{\pm 1.79}$	$71.95_{\pm 3.44}$	$69.65_{\pm 3.18}$	$67.65_{\pm 2.97}$	$64.79_{\pm 2.13}$	$63.60_{\pm 1.98}$	$61.75_{\pm 1.88}$	$61.35_{\pm 1.06}$
		w/ TSP	$\mathbf{86.70}_{\pm 0.30}$	$\mathbf{81.60}_{\pm 0.54}$	$\mathbf{75.84}_{\pm 2.67}$	$\mathbf{72.28}_{\pm 2.89}$	$\mathbf{70.06}_{\pm 2.83}$	$\mathbf{68.60}_{\pm 2.70}$	$\mathbf{65.83}_{\pm 2.63}$	$\mathbf{64.07}_{\pm 1.58}$	$\mathbf{62.79}_{\pm 1.64}$	$\mathbf{62.45}_{\pm 1.34}$
		Δ	+0.14	+1.54	+0.88	+0.33	+0.41	+0.95	+1.04	+0.47	+1.04	+1.10
	Forgetting ↓	w/o TSP	–	$13.54_{\pm 6.63}$	$11.43_{\pm 1.91}$	$12.49_{\pm 1.06}$	$12.71_{\pm 0.85}$	$12.27_{\pm 0.15}$	$13.88_{\pm 0.72}$	$13.47_{\pm 0.44}$	$14.19_{\pm 1.52}$	$14.01_{\pm 0.90}$
		w/ TSP	–	$\mathbf{10.56}_{\pm 0.60}$	$\mathbf{9.19}_{\pm 2.12}$	$\mathbf{9.98}_{\pm 1.49}$	$\mathbf{9.95}_{\pm 1.04}$	$\mathbf{9.24}_{\pm 0.91}$	$\mathbf{10.52}_{\pm 1.26}$	$\mathbf{10.77}_{\pm 0.69}$	$\mathbf{10.49}_{\pm 0.86}$	$\mathbf{10.04}_{\pm 1.29}$
		Δ	–	−2.98	−2.24	−2.51	−2.76	−3.03	−3.36	−2.70	−3.70	−3.97

5.3 In-Depth Analysis

Comparison of different merge strategies. We further compare different merge strategies on 10S-ImageNet-R and 10S-ImageNet-A to analyze how the choice of merge rule affects continual learning performance. Seq-FT is used as a without merge baseline that updates the inference model by directly adding the current task LoRA update without any merge rule. As shown in Tab. 4, EMA-style and LoRA-P&M both outperform Seq-FT, confirming the benefit of merging in rehearsal-free CL.

However, these methods still rely on a scalar factor shared by all parameters, matching the limitation discussed in Sec. 3. In contrast, HIM directly solves the target merged update under Eq. (4) and obtains the element-wise solution in Eq. (9). It achieves the best results, improving $\mathcal{A}_{Last}$ over EMA-style averaging by 1.22%/1.71% and over LoRA-P&M by 1.47%/1.42% on ImageNet-R/ImageNet-A, showing the assigning merge strengths at the element level is more effective than using a shared scalar factor. We further compare HIM with a HIM variant (HIM-last) that only uses task $t-1$ in Eq. (2); without explicit constraints from earlier tasks, this variant suffers from more severe forgetting, indicating that accumulated old task information in $\mathbf{F}_{1:t-1}$ is important for merging. More HIM variants are reported in Appendix C.3.

Analysis of perturbation training under cross-task merging. We further compare TSP with SAM [2] and without TSP to examine the effect of different perturbation training strategies during subsequent merging. As shown in Tab. 5, SAM performs worse than HIM alone. Compared with HIM alone, TSP improves $\mathcal{A}_{Last}$ by $0.98\%/1.10\%$ on ImageNet-R/ImageNet-A. To better understand this result, we visualize its behavior on both the current task and the old tasks after merging. Fig. 2 shows that SAM can still obtain a low loss solution on task 2, but the merged solution falls into a less favorable region for old tasks 0 and 1. This indicates that optimizing current task sharpness cannot well capture the cross-task interference caused by sequential LoRA updates. In contrast, TSP constructs the perturbation from one label-disjoint subset and evaluates it on the other subset. As a

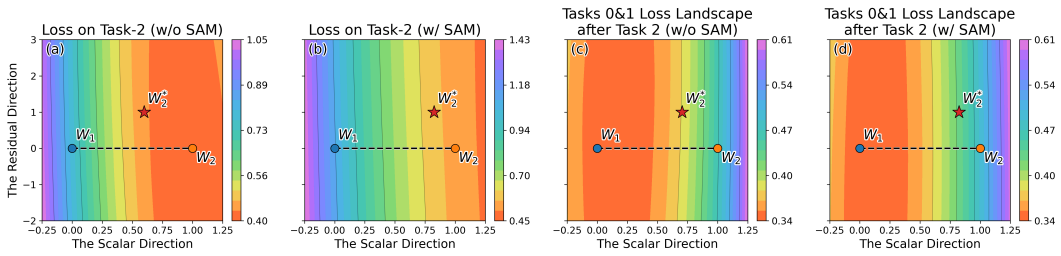


Figure 2: Analysis of the effect of SAM on 10S-ImageNet-R. Panels (a,b) visualize the task 2 loss landscapes for updates learned without/with SAM, while panels (c,d) show the total old task loss after merging. The dashed line denotes scalar interpolation, and the red star marks the merged solution.

Table 7: Computational and memory overhead on 10S-ImageNet-A.

Methods	GPU Memory (MiB)	Time/Batch (s)	Extra Memory (MB)	$\mathcal{A}_{Last} \uparrow$
Seq-FT	4560	0.12	0.0	$52.56_{\pm 2.08}$
TSP	2881	0.34	0.0	$56.57_{\pm 2.31}$
HIM	4560	0.12	54.0	$61.35_{\pm 1.06}$
HIM w/ SAM	4568	0.26	54.0	$60.46_{\pm 1.51}$
HIM w/ TSP	**2881**	0.34	54.0	$\mathbf{62.45_{\pm 1.34}}$

result, our method achieves the best performance, improving $\mathcal{A}_{Last}$ over SAM by 1.79%/1.99% on ImageNet-R/ImageNet-A.

Effect of TSP on forgetting. TSP aims to reduce the sensitivity of the current task update to the parameter change introduced by the subsequent merge. As shown in Tab. 6, adding TSP consistently improves total accuracy and reduces forgetting across incremental stages on both ImageNet-R and ImageNet-A. The final forgetting decreases from 7.76% to 5.77% on ImageNet-R and from 14.01% to 10.04% on ImageNet-A. This supports our motivation that stabilizing the current task update can alleviate forgetting during continual merging. More hyper-parameter analyses and results with different backbones are provided in Appendix C.5 and Appendix C.6.

Computational and Memory Overhead. Table 7 compares the overhead of HIM and TSP on 10S-ImageNet-A. HIM requires 54.0 MB of additional Fisher storage, independent of the number of tasks, and 3.8 s per task for Fisher estimation. TSP increases the training time per batch from 0.12 s to 0.34 s, but reduces GPU memory usage from 4560 MiB to 2881 MiB. These costs are limited to training, with no additional inference overhead. Further results on the effect of LoRA on overhead and accuracy are provided in Appendix C.2.

5.4 Further Analysis

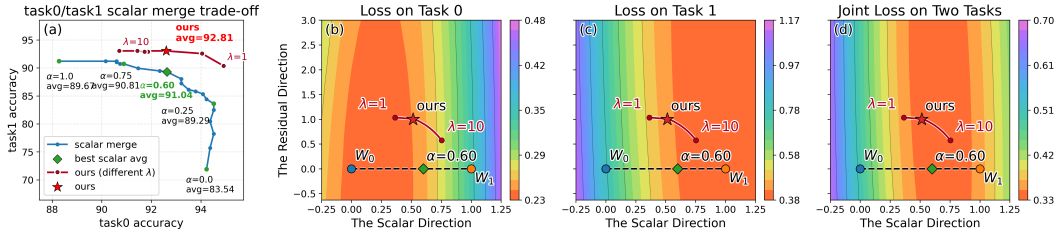


Figure 3: Analysis of scalar merge and HIM without TSP on 10S-ImageNet-R. Panel (a) compares task 0/1 accuracy under different merge choices, and panels (b–d) show the task 0, task 1, and joint loss landscapes. The dashed line denotes scalar interpolation; the green diamond and red star mark the best scalar merge and selected HIM solution, respectively.

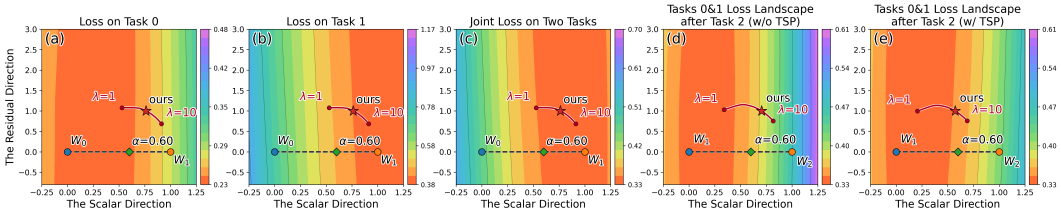


Figure 4: Analysis of the effect of TSP on 10S-ImageNet-R. Panels (a–c) show the task 0, task 1, and joint loss landscapes after introducing TSP, corresponding to Fig. 3(b–d). Panels (d,e) isolate TSP at the task 2 merge step: from the same task 1 model, we learn and merge the task 2 update without/with TSP and visualize the joint loss over old tasks 0 and 1. The dashed line denotes scalar interpolation, and the red star marks the merged solution.

Further analysis of HIM and TSP. We further analyze the behaviors of HIM and TSP using the task 1 accuracy against task 0 accuracy and the corresponding loss landscape visualizations. For the loss landscapes, we construct a 2D plane to compare the scalar interpolation path with our merged

solution. Let $\mathbf{W}_1$ denote the model obtained by directly adding the task 1 update to $\mathbf{W}_0$, and define the scalar interpolation direction as $\mathbf{u} = \mathbf{W}_1 - \mathbf{W}_0$. Let $\mathbf{d} = \mathbf{W}_{\text{ours}} - \mathbf{W}_0$ denote the displacement from $\mathbf{W}_0$ to our selected merged solution. We then remove from $\mathbf{d}$ its component along $\mathbf{u}$ and obtain the residual direction $\mathbf{r} = \mathbf{d} - \frac{\langle \mathbf{d}, \mathbf{u} \rangle}{\langle \mathbf{u}, \mathbf{u} \rangle} \mathbf{u}$ and $\mathbf{v} = \mathbf{r}/\|\mathbf{r}\|_2$, where $\mathbf{v}$ is orthogonal to the scalar interpolation direction. The loss is visualized on the plane $\mathbf{W}(x, y) = \mathbf{W}_0 + x\mathbf{u} + y\mathbf{v}$.

Observation 1: HIM reaches a better solution beyond scalar interpolation. Fig. 3(a) shows that scalar merge is restricted to a interpolation path, while varying λ in our method produces solutions outside this path. This indicates that HIM provides a more flexible search space than a shared scalar factor. More importantly, the joint loss landscape in Fig. 3(d) shows that the red curve produced by varying λ mostly remains in the low loss region, rather than being restricted to the scalar path. This suggests that the merged update under the joint objective can provide more flexible balances between previous task and current task.

Observation 2: TSP improves the local stability of the merged solution. Compared with Fig. 3(b)–(d), Fig. 4(a)–(c) show that introducing TSP makes the low loss regions around the merged solution visibly wider for task 0, task 1, and the joint loss. This suggests that TSP makes the learned update less sensitive to local parameter changes before merging. To further isolate the effect of TSP in the next merge step, Fig. 4(d) and Fig. 4(e) start from the same model before learning task 2 and visualize the joint loss over old tasks 0 and 1 after merging the task 2 update without and with TSP, respectively. With TSP, the old task low loss region becomes wider, and the merged solution is located within more stable region, indicating better robustness to the parameter change introduced by the task 2 merge.

6 Conclusion

In this work, we show that different parameters in LoRA update may prefer different merge strengths. Motivated by this, we propose M^{2}LE, which directly solves for the target merged update, yielding an analytic element-wise modulation of the current LoRA update. M^{2}LE first derives Hessian Information-guided Adaptive Merging (HIM), an analytic element-wise modulation rule under a joint loss objective over old and new tasks. It further introduces Task Simulation Perturbation (TSP) to simulate cross-task interference during training and improve the stability during subsequent merging. Overall, M^{2}LE offers a new view for continual learning with LoRA, showing that adapting merge strengths for different LoRA elements provides an effective way to balance stability and plasticity.

Acknowledgments and Disclosure of Funding

This work was supported in part by the National Key R&D Program of China under Grant No.2023YFA1008600, in part by the National Natural Science Foundation of China under Grants 62576262, in part by the Natural Science Foundation Basic Research Program of Shaanxi Province under grant 2026JC-YXQN-202, in part by Scientific and Technological Innovation Teams in Shaanxi Province under Grant 2025RS-CXTD-011, in part by the Shaanxi Province Core Technology Research and Development Project under Grant 2024QY2-GJHX-11, in part by the Natural Science Basic Research Program of Shaanxi under Grant 2026SYS-SYSZD-105, in part by the Key Research and Development Program of Shaanxi under Grant 2026JC-ZDGG-42, in part by the Establishment Fund of the State Key Laboratory of Wakefulness/Sleep and Cognition (Chongqing Institute for Brain and Intelligence), in part by the Fundamental Research Funds for the Central Universities under Grant QTZX25083 and QTZX26138.

References

[1] 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. *arXiv preprint arXiv:2010.11929*, 2020.

[2] Pierre Foret, Ariel Kleiner, Hossein Mobahi, and Behnam Neyshabur. Sharpness-aware minimization for efficiently improving generalization. *arXiv preprint arXiv:2010.01412*, 2020.

[3] Takuma Fukuda, Hiroshi Kera, and Kazuhiko Kawamoto. Adapter merging with centroid prototype mapping for scalable class-incremental learning. In *Proceedings of the computer vision and pattern recognition conference*, pages 4884–4893, 2025.

[4] Yufei Guo, Yinan Wu, Haoran Duan, Guiguang Ding, and Jungong Han. Phaselora: Control-regime-conditioned low-rank adaptation for continuous-action vision-language-action policies. *arXiv preprint arXiv:2608.15285*, 2026.

[5] Lingfeng He, De Cheng, Huaijie Wang, Xiaofeng Zhu, Xi Yang, Nannan Wang, and Xinbo Gao. Task-driven subspace decomposition for knowledge sharing and isolation in lora-based continual learning. In *Forty-third International Conference on Machine Learning*, 2026.

[6] Lingfeng He, De Cheng, Di Xu, Huaijie Wang, and Nannan Wang. Harnessing textual semantic priors for knowledge transfer and refinement in clip-driven continual learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 40, pages 21645–21653, 2026.

[7] Dan Hendrycks, Steven Basart, Norman Mu, Saurav Kadavath, Frank Wang, Evan Dorundo, Rahul Desai, Tyler Zhu, Samyak Parajuli, Mike Guo, et al. The many faces of robustness: A critical analysis of out-of-distribution generalization. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 8340–8349, 2021.

[8] Dan Hendrycks, Kevin Zhao, Steven Basart, Jacob Steinhardt, and Dawn Song. Natural adversarial examples. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 15262–15271, 2021.

[9] Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Liang Wang, Weizhu Chen, et al. Lora: Low-rank adaptation of large language models. *Iclr*, 1(2):3, 2022.

[10] Linlan Huang, Xusheng Cao, Haori Lu, and Xialei Liu. Class-incremental learning with clip: Adaptive representation adjustment and parameter fusion. In *European Conference on Computer Vision*, pages 214–231. Springer, 2024.

[11] James Kirkpatrick, Razvan Pascanu, Neil Rabinowitz, Joel Veness, Guillaume Desjardins, Andrei A Rusu, Kieran Milan, John Quan, Tiago Ramalho, Agnieszka Grabska-Barwinska, et al. Overcoming catastrophic forgetting in neural networks. *Proceedings of the national academy of sciences*, 114(13):3521–3526, 2017.

[12] 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*, pages 554–561, 2013.

[13] Alex Krizhevsky, Geoffrey Hinton, et al. Learning multiple layers of features from tiny images. 2009.

[14] Mei Li, Yuxiang Lu, Qinyan Dai, Suizhi Huang, Yue Ding, and Hongtao Lu. Became: Bayesian continual learning with adaptive model merging. *arXiv preprint arXiv:2504.02666*, 2025.

[15] Yan-Shuo Liang and Wu-Jun Li. Inflora: Interference-free low-rank adaptation for continual learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 23638–23647, 2024.

[16] Yan-Shuo Liang and Wu-Jun Li. Gated integration of low-rank adaptation for continual learning of language models. 2025.

[17] Guoliang Lin, Hanlu Chu, and Hanjiang Lai. Towards better plasticity-stability trade-off in incremental learning: A simple linear connector. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 89–98, 2022.

[18] Ruiqi Liu, Boyu Diao, Hangda Liu, Zhulin An, Fei Wang, and Yongjun Xu. From isolation to integration: Building an adaptive expert forest for pre-trained model-based class-incremental learning. *arXiv preprint arXiv:2602.20911*, 2026.

[19] Yue Lu, Shizhou Zhang, De Cheng, Yinghui Xing, Nannan Wang, Peng Wang, and Yanning Zhang. Visual Prompt Tuning in Null Space for Continual Learning. In *Conference on Neural Information Processing Systems*, 2024.

[20] Yue Lu, Shizhou Zhang, De Cheng, Guoqiang Liang, Yinghui Xing, Nannan Wang, and Yanning Zhang. Training consistent mixture-of-experts-based prompt generator for continual learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 39, pages 19152–19160, 2025.

[21] Subhransu Maji, Esa Rahtu, Juho Kannala, Matthew Blaschko, and Andrea Vedaldi. Fine-grained visual classification of aircraft. *arXiv preprint arXiv:1306.5151*, 2013.

[22] Marc Masana, Xialei Liu, Bartłomiej Twardowski, Mikel Menta, Andrew D Bagdanov, and Joost Van De Weijer. Class-incremental learning: survey and performance evaluation on image classification. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 45(5): 5513–5533, 2022.

[23] Michael McCloskey and Neal J Cohen. Catastrophic interference in connectionist networks: The sequential learning problem. In *Psychology of learning and motivation*, volume 24, pages 109–165. Elsevier, 1989.

[24] Xingchao Peng, Qinxun Bai, Xide Xia, Zijun Huang, Kate Saenko, and Bo Wang. Moment matching for multi-source domain adaptation. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 1406–1415, 2019.

[25] Haomiao Qiu, Miao Zhang, Ziyue Qiao, and Liqiang Nie. Train with perturbation, infer after merging: A two-stage framework for continual learning. *arXiv preprint arXiv:2505.22389*, 2025.

[26] 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 *International conference on machine learning*, pages 8748–8763. PmLR, 2021.

[27] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, et al. Imagenet large scale visual recognition challenge. *International journal of computer vision*, 115(3):211–252, 2015.

[28] James Seale Smith, Leonid Karlinsky, Vyshnavi Gutta, Paola Cascante-Bonilla, Donghyun Kim, Assaf Arbelle, Rameswar Panda, Rogerio Feris, and Zsolt Kira. Coda-prompt: Continual decomposed attention-based prompting for rehearsal-free continual learning. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 11909–11919, 2023.

[29] Khurram Soomro, Amir Roshan Zamir, and Mubarak Shah. Ucf101: A dataset of 101 human actions classes from videos in the wild. *arXiv preprint arXiv:1212.0402*, 2012.

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

[31] Hai-Long Sun, Da-Wei Zhou, Hanbin Zhao, Le Gan, De-Chuan Zhan, and Han-Jia Ye. Mos: Model surgery for pre-trained model-based class-incremental learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 39, pages 20699–20707, 2025.

[32] Yuwen Tan, Qinhao Zhou, Xiang Xiang, Ke Wang, Yuchuan Wu, and Yongbin Li. Semantically-shifted incremental adapter-tuning is a continual vitransformer. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 23252–23262, 2024.

[33] Anke Tang, Enneng Yang, Li Shen, Yong Luo, Han Hu, Lefei Zhang, Bo Du, and Dacheng Tao. Merging on the fly without retraining: A sequential approach to scalable continual model merging. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2025.

[34] Catherine Wah, Steve Branson, Peter Welinder, Pietro Perona, and Serge Belongie. The caltech-ucsd birds-200-2011 dataset. 2011.

[35] Huaijie Wang, De Cheng, Lingfeng He, Yan Li, Jie Li, Nannan Wang, and Xinbo Gao. Ekpc: Elastic knowledge preservation and compensation for class-incremental learning: H. wang et al. *International Journal of Computer Vision*, 134(5):238, 2026.

[36] Huiyi Wang, Haodong Lu, Lina Yao, and Dong Gong. Self-expansion of pre-trained models with mixture of adapters for continual learning. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pages 10087–10098, 2025.

[37] Xiequn Wang, Zhan Zhuang, and Yu Zhang. Plan: Proactive low-rank allocation for continual learning. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 2909–2918, 2025.

[38] Zifeng Wang, Zizhao Zhang, Sayna Ebrahimi, Ruoxi Sun, Han Zhang, Chen-Yu Lee, Xiaoqi Ren, Guolong Su, Vincent Perot, Jennifer Dy, et al. Dualprompt: Complementary prompting for rehearsal-free continual learning. In *European conference on computer vision*, pages 631–648. Springer, 2022.

[39] Zifeng Wang, Zizhao Zhang, Chen-Yu Lee, Han Zhang, Ruoxi Sun, Xiaoqi Ren, Guolong Su, Vincent Perot, Jennifer Dy, and Tomas Pfister. Learning to prompt for continual learning. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 139–149, 2022.

[40] Fangwen Wu, Lechao Cheng, Shengeng Tang, Xiaofeng Zhu, Chaowei Fang, Dingwen Zhang, and Meng Wang. Navigating semantic drift in task-agnostic class-incremental learning. *arXiv preprint arXiv:2502.07560*, 2025.

[41] Yichen Wu, Hongming Piao, Long-Kai Huang, Renzhen Wang, Wanhua Li, Hanspeter Pfister, Deyu Meng, Kede Ma, and Ying Wei. Sd-lora: Scalable decoupled low-rank adaptation for class incremental learning. *arXiv preprint arXiv:2501.13198*, 2025.

[42] Fanhu Zeng, Haiyang Guo, Fei Zhu, Li Shen, and Hao Tang. Robustmerge: Parameter-efficient model merging for mllms with direction robustness. *arXiv preprint arXiv:2502.17159*, 2025.

[43] Gengwei Zhang, Liyuan Wang, Guoliang Kang, Ling Chen, and Yunchao Wei. Slca: Slow learner with classifier alignment for continual learning on a pre-trained model. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 19148–19158, 2023.

[44] Yaoyue Zheng, Yin Zhang, Joost van de Weijer, Gido M van de Ven, Shaoyi Du, Xuetao Zhang, and Zhiqiang Tian. Revisiting weight regularization for low-rank continual learning. *arXiv preprint arXiv:2602.17559*, 2026.

[45] Da-Wei Zhou, Zi-Wen Cai, Han-Jia Ye, De-Chuan Zhan, and Ziwei Liu. Revisiting class-incremental learning with pre-trained models: Generalizability and adaptivity are all you need. *International Journal of Computer Vision*, pages 1–21, 2024.

[46] Da-Wei Zhou, Yuanhan Zhang, Yan Wang, Jingyi Ning, Han-Jia Ye, De-Chuan Zhan, and Ziwei Liu. Learning without forgetting for vision-language models. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2025.

[47] 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.

[48] Hao Zhu, Yifei Zhang, Junhao Dong, and Piotr Koniusz. Bilora: almost-orthogonal parameter spaces for continual learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 25613–25622, 2025.

A Training Pipeline

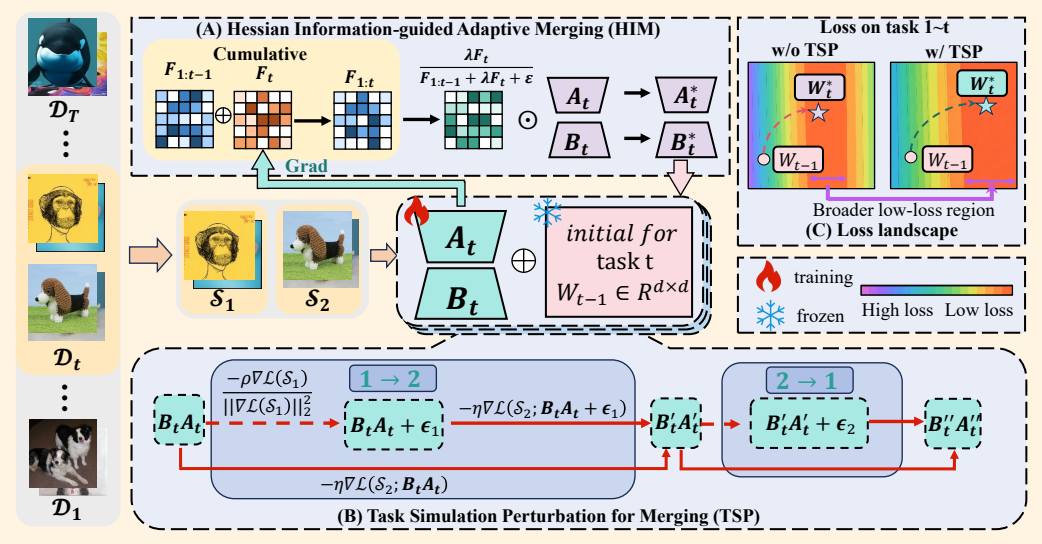


Figure 5: Overall framework of M^{2}LE. (B) During training, TSP simulates cross task interference between label-disjoint subsets. (A) After training, HIM adaptively merges the LoRA update using historical and current-task Fisher information.

Figure 5 illustrates the overall framework of M^{2}LE, and the training pipeline of M^{2}LE is outlined in Algorithm 1. At task t, we start from the inference parameter $\mathbf{W}_{t-1}$ and adapt the frozen backbone on the current data $\mathcal{D}_t$ with LoRA, where the current update is $\Delta\mathbf{W}_t = \mathbf{B}_t\mathbf{A}_t$. During current task training, TSP simulates cross-task interference with label-disjoint subsets, encouraging the learned update to remain stable for subsequent merging. After convergence, HIM estimates the current task Fisher $\mathbf{F}_t$ and obtains the target merged update $\Delta\mathbf{W}_t^*$ by Eq. (9). The merged update is then integrated into the inference model, and the Fisher statistics are accumulated for later tasks.

Algorithm 1 Training pipeline of the proposed Merging Strength Modulation via Joint Loss Estimation (M^{2}LE).

Input: Task sequence $\{\mathcal{D}_1, \ldots, \mathcal{D}_T\}$, where $\mathcal{D}_t = \{(x_t^i, y_t^i)\}_{i=1}^{n_t}$; ViT model $f(\cdot)$ with weight $\mathbf{W}_0$; LoRA $\Delta\mathbf{W}_t = \mathbf{B}_t\mathbf{A}_t$ in each layer; accumulated Fisher statistics $\mathbf{F}_{1:t-1}$ in each layer.
Output: The adapted inference parameters $\mathbf{W}_t$ for each layer in the ViT model.

Initialization: $\mathbf{F}_{1:t-1} \leftarrow \mathbf{0}^{d_o \times d_i}$.
for task $t = 1$ to T **do**
  Freeze the inference weight $\mathbf{W}_{t-1}$ and attach current-task LoRA modules.
  for each LoRA layer **do**
    Initialize current-task LoRA parameters $\mathbf{A}_t$ and $\mathbf{B}_t$.
  // During Training: Task Simulation Perturbation for Merging
  repeat
    Sample a mini-batch $\mathcal{S}$ from $\mathcal{D}_t$ and split it into two label-disjoint subsets $\mathcal{S}_1$ and $\mathcal{S}_2$;
    Update trainable parameters with the dual TSP objective in Eq. (14) and Eq. (15).
  until Convergence on $\mathcal{D}_t$
  // After Training: Hessian Information-guided Adaptive Merging
  for each LoRA layer **do**
    Given the trained update $\Delta\mathbf{W}_t$, estimate $\mathbf{F}_t$ on $\mathcal{D}_t$ and compute the HIM merged update $\Delta\mathbf{W}_t^*$ by Eq. (9).
    Update the inference model weight $\mathbf{W}_t \leftarrow \mathbf{W}_{t-1} + \Delta\mathbf{W}_t^*$;
    Update accumulated Fisher $\mathbf{F}_{1:t} \leftarrow \mathbf{F}_{1:t-1} + \mathbf{F}_t$.;

B Theoretical Analysis

Starting from Eq. (7), we solve the following quadratic problem over the candidate merged update $\Delta\hat{\mathbf{w}}_t$:

$$\begin{aligned}\Delta\mathbf{w}_t^* = \arg\min_{\Delta\hat{\mathbf{w}}_t} \quad & \sum_{i=1}^{t-1}\frac{1}{2}\Delta\hat{\mathbf{w}}_t^\top\mathbf{H}_i\Delta\hat{\mathbf{w}}_t \\ & + \frac{\lambda}{2}(\Delta\hat{\mathbf{w}}_t - \Delta\mathbf{w}_t)^\top\mathbf{H}_t(\Delta\hat{\mathbf{w}}_t - \Delta\mathbf{w}_t).\end{aligned} \tag{B.0.1}$$

We first expand the current task term and separate the parts that depend on $\Delta\hat{\mathbf{w}}_t$:

$$\begin{aligned}&\frac{\lambda}{2}(\Delta\hat{\mathbf{w}}_t - \Delta\mathbf{w}_t)^\top\mathbf{H}_t(\Delta\hat{\mathbf{w}}_t - \Delta\mathbf{w}_t) \\ =&\frac{\lambda}{2}\Delta\hat{\mathbf{w}}_t^\top\mathbf{H}_t\Delta\hat{\mathbf{w}}_t - \lambda\Delta\hat{\mathbf{w}}_t^\top\mathbf{H}_t\Delta\mathbf{w}_t + \frac{\lambda}{2}\Delta\mathbf{w}_t^\top\mathbf{H}_t\Delta\mathbf{w}_t.\end{aligned} \tag{B.0.2}$$

The last term in Eq. (B.0.2) is constant with respect to $\Delta\hat{\mathbf{w}}_t$ and can be ignored when solving the minimization problem. Thus, the derivative of Eq. (B.0.1) with respect to $\Delta\hat{\mathbf{w}}_t$ is:

$$\begin{aligned}&\frac{\partial}{\partial\Delta\hat{\mathbf{w}}_t}\left[\sum_{i=1}^{t-1}\frac{1}{2}\Delta\hat{\mathbf{w}}_t^\top\mathbf{H}_i\Delta\hat{\mathbf{w}}_t + \frac{\lambda}{2}\Delta\hat{\mathbf{w}}_t^\top\mathbf{H}_t\Delta\hat{\mathbf{w}}_t - \lambda\Delta\hat{\mathbf{w}}_t^\top\mathbf{H}_t\Delta\mathbf{w}_t\right] \\ &= \sum_{i=1}^{t-1}\mathbf{H}_i\Delta\hat{\mathbf{w}}_t + \lambda\mathbf{H}_t\Delta\hat{\mathbf{w}}_t - \lambda\mathbf{H}_t\Delta\mathbf{w}_t.\end{aligned} \tag{B.0.3}$$

By setting Eq. (B.0.3) to zero and denoting the solution by $\Delta\mathbf{w}_t^*$, we obtain:

$$\sum_{i=1}^{t-1}\mathbf{H}_i\Delta\mathbf{w}_t^* + \lambda\mathbf{H}_t\Delta\mathbf{w}_t^* - \lambda\mathbf{H}_t\Delta\mathbf{w}_t = \mathbf{0}. \tag{B.0.4}$$

Rearranging the terms gives the linear system, which corresponds to Eq. (8):

$$\left(\sum_{i=1}^{t-1}\mathbf{H}_i + \lambda\mathbf{H}_t\right)\Delta\mathbf{w}_t^* = \lambda\mathbf{H}_t\Delta\mathbf{w}_t. \tag{B.0.5}$$

C More Detailed Experimental Results

C.1 Forgetting metric

We measure forgetting after task $t \geq 1$ as $\text{Forgetting}_t = \frac{1}{t}\sum_{i=0}^{t-1}\left(\max_{j\in\{i,\ldots,t-1\}} a_{j,i} - a_{t,i}\right)$, where $a_{j,i}$ denotes the accuracy on task i after learning task j.

C.2 Effect of Different LoRA Ranks on M^{2}LE Performance

We study the effect of rank in M^{2}LE on 20S-ImageNet-R in Tab. 8. We vary the LoRA rank $r \in \{4, 8, 16, 32, 64\}$ and denote each variant as M^{2}LE-r=r. Across all ranks, M^{2}LE achieves competitive performance with only a small number of trainable parameters, ranging from 0.15M to 2.36M. It also keeps the inference overhead unchanged, with 0.00M extra parameters and 17.60 GFLOPs, since the learned components are merged into the backbone after training. For each LoRA layer, the maintained Fisher statistic $\mathbf{F}_{1:t}$ has size $d_o \times d_i$, so the extra storage remains 54.0MB for all ranks once the layer dimensions are fixed.

As r increases from 4 to 32, $\mathcal{A}_{Last}$ improves from 76.92 to 78.62, while $\mathcal{A}_{Avg}$ increases from 83.28 to 84.62. Further increasing the rank to 64 brings only marginal improvement in $\mathcal{A}_{Avg}$ (84.66) but slightly decreases $\mathcal{A}_{Last}$ to 78.20, together with higher training cost. Thus, we adopt $r = 32$ as a practical trade-off between accuracy and efficiency in the remaining experiments.

Table 8: Analysis of the effect of rank on additional parameters, storage and computational overhead, and performance on 20S-ImageNet-R. We compare our M^{2}LE with different ranks against representative CL methods.

Methods	Venue	Training Stage (Per Sample)			Inference Stage (Per Sample)		20S-ImageNet-R	
		Extra Memory	Trainable Params	Training GFLOPs	Extra Params	Inference GFLOPs	$\mathcal{A}_{Last}\uparrow$	$\mathcal{A}_{Avg}$
Methods based on the ViT-21K model (86M).								
CODA [28]	CVPR-23	15.4MB	3.84M (4.46%)	35.17	3.84M (4.46%)	35.17	$69.96_{\pm 0.50}$	$75.34_{\pm 0.85}$
SLCA [43]	ICCV-23	450.6MB	86M (100%)	17.60	0.00M (0.00%)	17.60	$74.63_{\pm 1.55}$	$79.92_{\pm 1.29}$
SSIAT [32]	CVPR-24	450.6MB	1.19M (1.38%)	17.81	1.19M (1.38%)	17.81	$75.67_{\pm 0.24}$	$82.30_{\pm 0.36}$
InfLoRA [15]	CVPR-24	11.3MB	0.37M (0.43%)	17.67	0.00M (0.00%)	17.67	$71.01_{\pm 0.45}$	$77.28_{\pm 0.45}$
VPT-NSP [19]	NIPS-24	28.4MB	0.09M (0.10%)	17.72	0.09M (0.10%)	17.72	$75.69_{\pm 0.61}$	$81.87_{\pm 0.59}$
SD-LoRA [41]	ICLR-25	14.1MB	0.37M (0.43%)	20.51	0.00M (0.00%)	17.72	$75.26_{\pm 0.37}$	$80.22_{\pm 0.72}$
VPT-CPG [20]	AAAI-25	28.4MB	1.66M (1.93%)	17.72	1.66M (1.93%)	17.72	$75.33_{\pm 0.37}$	$80.22_{\pm 0.72}$
M^{2}LE–r=4 (Ours)	–	54.0MB	0.15M (0.17%)	17.59	0.00M (0.00%)	17.60	$\mathbf{76.92_{\pm 0.20}}$	$\mathbf{83.28_{\pm 0.25}}$
M^{2}LE–r=8 (Ours)	–	54.0MB	0.29M (0.34%)	17.62	0.00M (0.00%)	17.60	$\mathbf{77.32_{\pm 0.55}}$	$\mathbf{83.68_{\pm 0.51}}$
M^{2}LE–r=16 (Ours)	–	54.0MB	0.59M (0.69%)	17.68	0.00M (0.00%)	17.60	$\mathbf{78.01_{\pm 0.13}}$	$\mathbf{84.07_{\pm 0.43}}$
M^{2}LE–r=32 (Ours)	–	54.0MB	1.18M (1.37%)	17.80	0.00M (0.00%)	17.60	$\mathbf{78.62_{\pm 0.43}}$	$\mathbf{84.62_{\pm 0.26}}$
M^{2}LE–r=64 (Ours)	–	54.0MB	2.36M (2.74%)	18.03	0.00M (0.00%)	17.60	$\mathbf{78.20_{\pm 0.30}}$	$\mathbf{84.66_{\pm 0.70}}$

Table 9: Comparison of different HIM variants on 10S-ImageNet-R and 10S-ImageNet-A.

Variant	10S-ImageNet-R		10S-ImageNet-A	
	$\mathcal{A}_{Last}\uparrow$	$\mathcal{A}_{Avg}\uparrow$	$\mathcal{A}_{Last}\uparrow$	$\mathcal{A}_{Avg}\uparrow$
HIM-last ($k=1$)	$76.23_{\pm 0.76}$	$83.81_{\pm 0.61}$	$58.17_{\pm 2.38}$	$68.84_{\pm 1.93}$
HIM-$k=3$	$78.88_{\pm 0.62}$	$85.37_{\pm 0.50}$	$60.81_{\pm 1.10}$	$69.97_{\pm 2.28}$
HIM-$k=5$	$80.04_{\pm 0.35}$	$85.73_{\pm 0.47}$	$60.92_{\pm 0.97}$	$70.14_{\pm 2.48}$
HIM-$k=7$	$80.21_{\pm 0.45}$	$85.77_{\pm 0.45}$	$61.16_{\pm 1.15}$	$70.17_{\pm 2.42}$
HIM-full (Ours)	$\mathbf{80.36_{\pm 0.40}}$	$\mathbf{85.78_{\pm 0.44}}$	$\mathbf{61.35_{\pm 1.06}}$	$\mathbf{70.23_{\pm 2.04}}$

C.3 Analysis of HIM Variants

We further analyze several HIM variants that retain different numbers of past tasks in the old task term. For a more explicit comparison, we define a truncated HIM variant by retaining only the most recent k past tasks in the old task loss impact:

$$\mathcal{J}_{\text{old}}^{(k)}(\Delta\hat{\mathbf{W}}_t) = \sum_{i=\max(1,t-k)}^{t-1} \Big(\mathcal{L}_i(\mathbf{W}_{t-1} + \Delta\hat{\mathbf{W}}_t) - \mathcal{L}_i(\mathbf{W}_{t-1})\Big). \tag{C.3.1}$$

Here, $k \in \{1,3,5,7\}$ controls the number of past tasks used in the old task term. Correspondingly, this variant replaces the accumulated Fisher statistic in Eq. (9) with $\mathbf{F}_{t-k:t-1} = \sum_{i=\max(1,t-k)}^{t-1} \mathbf{F}_i$, while full HIM uses $\mathbf{F}_{1:t-1} = \sum_{i=1}^{t-1} \mathbf{F}_i$. When $k=1$, the variant uses only task $t-1$, corresponding to the HIM-last setting discussed in the main text.

As shown in Tab. 9, HIM-last gives much lower performance, i.e., $76.23/83.81$ on 10S-ImageNet-R and $58.17/68.84$ on 10S-ImageNet-A. When more past tasks are retained, the performance consistently improves, e.g., HIM-$k=5$ reaches $80.04/85.73$ and $60.92/70.14$ on the two datasets, respectively. Full HIM achieves the best results of $80.36/85.78$ and $61.35/70.23$, indicating that using a broader set of past tasks provides more complete constraints for preserving previously learned knowledge during merging. These results also validate the use of $\mathbf{F}_{1:t-1}$ in HIM, which aggregates information from all past tasks rather than relying only on recent ones.

C.4 Analysis of Historical Fisher Reuse

We further analyze the accumulated Fisher statistics in HIM on 10S-ImageNet-R. HIM estimates each task's Fisher at its pre-merge solution while the data are available, and retains only the accumulated statistic $F_{1:t-1}$. For comparison, Recomputed Fisher revisits previous task data before each merge and re-estimates their accumulated Fisher at its current merged model. The recomputed statistic is used for merging and subsequent learning, so the two methods follow their respective model trajectories. We measure their Fisher similarity by $S_t = \frac{1}{|\mathcal{L}|}\sum_{\ell\in\mathcal{L}} \cos(F_{1:t-1}^{\ell}, \widetilde{F}_{1:t-1}^{\ell})$, where $\mathcal{L}$

Table 10: Task-by-task comparison of accumulated historical Fisher and recomputed Fisher on 10S-ImageNet-R. Results are obtained with a single random seed.

Metric	Task 0	Task 1	Task 2	Task 3	Task 4	Task 5	Task 6	Task 7	Task 8	Task 9
$S_t \uparrow$	–	1.000	0.988	0.986	0.986	0.987	0.988	0.987	0.987	0.986
HIM $\mathcal{A}_{Last} \uparrow$	96.66	92.81	89.41	87.39	84.69	83.84	83.18	82.62	81.68	80.67
Recomputed Fisher $\mathcal{A}_{Last} \uparrow$	96.66	92.81	89.36	87.11	84.76	83.70	83.15	82.70	81.68	80.67

denotes the set of layers with Fisher statistics, and $F^{\ell}_{1:t-1}$ and $\widetilde{F}^{\ell}_{1:t-1}$ are the accumulated statistics from HIM and Recomputed Fisher, respectively.

As shown in Tab. 10, S_t remains at least 0.986 across the evaluated stages, indicating similar directions of the accumulated Fisher statistics. Recomputed Fisher achieves similar accuracy to HIM, with a maximum difference of 0.28 performance and the same final $\mathcal{A}_{Last}$ of 80.67%. These results support using the accumulated historical Fisher in HIM, avoiding repeated access to previous task data with little impact on performance over the evaluated task sequence.

C.5 Hyper-Parameter Analysis

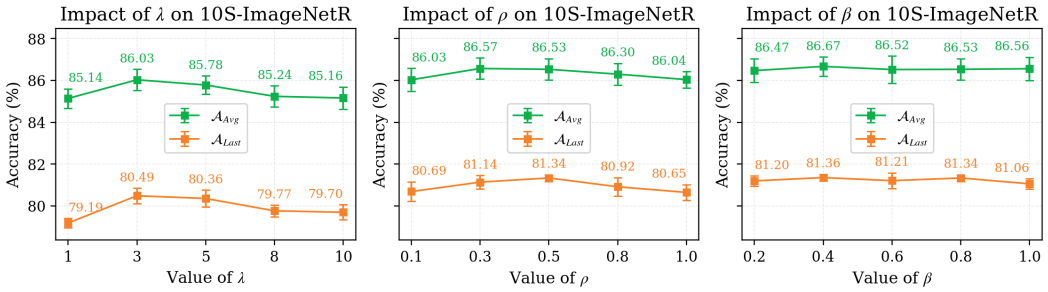


Figure 6: Hyper-parameter analysis on 10S-ImageNet-R. From left to right, λ, ρ, and β are varied while the other two are fixed. All results are averaged over three random seeds, and error bars denote the standard deviation.

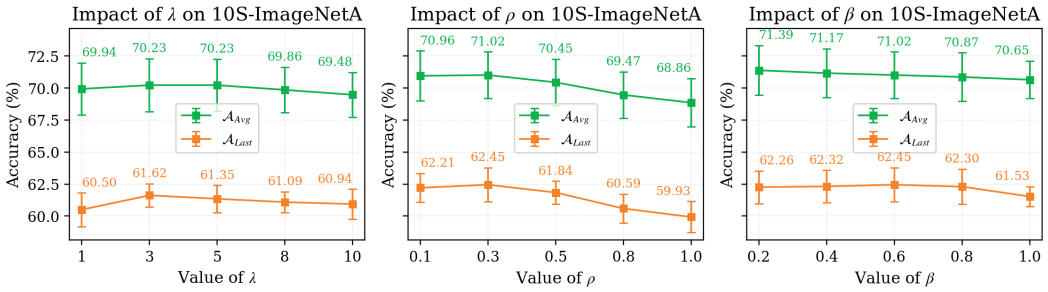


Figure 7: Same as Fig. 6, but on 10S-ImageNet-A.

We conduct experiments to investigate how the key hyper-parameters λ, ρ, and β affect performance on 10S-ImageNet-R and 10S-ImageNet-A. Here, λ is the trade-off coefficient in the merge objective in Eq. (9), ρ controls the perturbation magnitude in Eq. (11), and β balances the perturbed and clean losses in Eq. (13). We use the default settings $\lambda = 5$, $\rho = 0.5$, and $\beta = 0.8$ on ImageNet-R, and $\lambda = 5$, $\rho = 0.3$, and $\beta = 0.6$ on ImageNet-A. For each analysis, we vary only one hyper-parameter while fixing the other two at their default values. The results are shown in Fig. 6 and Fig. 7. All results are averaged over three random seeds, and we report the mean with error bars.

Impact of λ. The hyper-parameter λ controls the relative weight of the current task term in the merge objective, and thus adjusts the balance between old task retention and current task adaptation. A too-small λ (e.g., $\lambda = 1$) underweights the current task term and limits new task adaptation, whereas an overly large λ (e.g., $\lambda \geq 10$) makes the merged update overly biased toward the current task and

weakens old task retention. Overall, the results remain stable for $1 \leq \lambda \leq 10$, with the best trade-off achieved around $\lambda = 3$–5. Based on this observation, we set $\lambda = 5$.

Impact of ρ. The hyper-parameter ρ controls the perturbation magnitude in the TSP. A too-small ρ (e.g., $\rho = 0.1$) leads to a weak cross-subset probing effect, whereas a too-large ρ (e.g., $\rho \geq 0.8$) over-amplifies the perturbation. Overall, the results remain stable for $0.1 \leq \rho \leq 1.0$, with the best performance achieved at moderate values. Accordingly, we set $\rho = 0.5$ on ImageNet-R and $\rho = 0.3$ on ImageNet-A.

Impact of β. The hyper-parameter β controls the relative weight of the perturbed loss and the clean loss in the TSP. A too-small β (e.g., $\beta = 0.2$) underweights the perturbation signal, whereas a too-large β (e.g., $\beta = 1.0$) makes the perturbation effect overly strong and is less favorable overall. Overall, the results remain stable for $0.2 \leq \beta \leq 1.0$, and the best results are obtained at moderate values. Based on the results, we set $\beta = 0.8$ on ImageNet-R and $\beta = 0.6$ on ImageNet-A.

C.6 Experimental Results Using the CLIP Model as the Backbone

Table 11: Experimental results on ImageNet-R, CIFAR100 and CUB datasets using CLIP as the backbone. All methods are initialized with the CLIP pre-trained on LAION-400M ***without experience replay*** for fair comparisons. PROOF† represents our reproduced results of PROOF [46] without experience replay. The best results are in **bold** and the second highest results are underlined.

Method	ImageNet-R				CIFAR100				CUB			
	B0 Inc10		B50 Inc10		B0 Inc10		B50 Inc10		B0 Inc20		B100 Inc20	
	$\mathcal{A}_{Last}$	$\mathcal{A}_{Avg}$	$\mathcal{A}_{Last}$	$\mathcal{A}_{Avg}$	$\mathcal{A}_{Last}$	$\mathcal{A}_{Avg}$	$\mathcal{A}_{Last}$	$\mathcal{A}_{Avg}$	$\mathcal{A}_{Last}$	$\mathcal{A}_{Avg}$	$\mathcal{A}_{Last}$	$\mathcal{A}_{Avg}$
CoOp [47]	37.52	60.73	39.77	54.20	24.24	47.00	24.12	41.23	8.57	27.61	10.14	24.03
ZS-CLIP [26]	77.17	83.37	77.17	79.57	71.38	81.81	71.38	76.49	63.06	74.38	63.06	67.96
L2P [39]	66.52	75.97	66.77	72.82	73.03	82.74	73.61	81.14	57.93	70.87	66.12	75.64
DualPrompt [38]	66.65	76.21	67.58	73.22	72.44	81.63	72.57	80.12	57.46	69.89	64.84	71.40
CODA-Prompt [28]	68.95	77.69	68.05	73.71	73.43	82.43	71.58	78.69	62.98	73.12	62.21	73.95
SimpleCIL [45]	74.48	81.06	74.48	76.84	76.63	84.15	76.63	80.20	77.52	83.81	77.52	79.75
RAPF [10]	79.62	86.28	79.21	84.03	79.04	86.19	78.25	82.35	62.77	79.09	62.93	72.82
PROOF† [46]	77.25	82.69	77.03	80.56	76.29	84.88	75.93	81.85	72.60	79.94	71.80	76.43
M^{2}LE (Ours)	**82.05**	**87.16**	**82.65**	**85.24**	**80.52**	**87.88**	**83.12**	**87.80**	**80.03**	**86.92**	**80.41**	**84.25**

Table 12: Experimental results on Aircraft, Cars and UCF datasets using CLIP as the backbone. All methods are initialized with the CLIP pre-trained on LAION-400M ***without experience replay*** for fair comparisons. PROOF† represents our reproduced results of PROOF [46] without experience replay. The best results are in **bold** and the second highest results are underlined.

Method	Aircraft				Cars				UCF			
	B0 Inc10		B50 Inc10		B0 Inc10		B50 Inc10		B0 Inc20		B100 Inc20	
	$\mathcal{A}_{Last}$	$\mathcal{A}_{Avg}$	$\mathcal{A}_{Last}$	$\mathcal{A}_{Avg}$	$\mathcal{A}_{Last}$	$\mathcal{A}_{Avg}$	$\mathcal{A}_{Last}$	$\mathcal{A}_{Avg}$	$\mathcal{A}_{Last}$	$\mathcal{A}_{Avg}$	$\mathcal{A}_{Last}$	$\mathcal{A}_{Avg}$
CoOp [47]	7.14	14.54	7.77	13.05	21.65	36.46	20.87	37.40	33.46	47.85	24.74	42.02
ZS-CLIP [26]	17.22	26.66	17.22	21.70	76.37	82.60	76.37	78.32	67.64	75.50	67.64	71.44
L2P [39]	28.29	47.19	32.13	44.07	61.82	76.63	65.64	76.37	76.43	86.34	76.62	83.95
DualPrompt [38]	25.83	44.30	33.57	46.07	62.94	76.26	67.55	76.88	75.82	85.21	76.35	84.31
CODA-Prompt [28]	27.69	45.98	32.28	45.14	66.47	80.21	64.19	75.06	80.14	87.76	75.03	83.04
SimpleCIL [45]	48.09	59.24	48.09	53.05	86.85	92.04	86.85	88.96	85.68	90.44	85.68	88.12
RAPF [10]	23.61	50.38	25.44	40.47	62.85	82.89	63.19	75.87	80.33	92.28	81.55	90.31
PROOF† [46]	39.03	47.33	45.36	54.35	84.06	86.87	82.66	88.68	81.47	87.04	81.85	86.15
M^{2}LE (Ours)	**60.43**	**70.63**	**63.85**	**69.50**	**90.57**	**94.39**	**90.91**	**92.40**	**90.53**	**94.38**	**91.28**	**94.81**

To examine whether M^{2}LE can be transferred beyond a single visual backbone, we further conduct experiments with CLIP as the pre-trained model. The results on six benchmarks are summarized in Tab. 11 and Tab. 12.

Datasets and Evaluation Metrics. Following existing CLIP-based continual learning studies [10, 46], we evaluate M^{2}LE on ImageNet-R [7], CIFAR100 [13], CUB [34], Aircraft [21], Cars [12], and UCF [29]. ImageNet-R contains 200 ImageNet categories with different artistic renditions, such as sketches, cartoons, and paintings. CIFAR100 includes 100 object categories with 60,000 natural

images. CUB, Aircraft, and Cars are fine-grained recognition datasets, covering bird species, aircraft model variants, and car categories, respectively. UCF101 is an action recognition benchmark with 101 human action classes. For Aircraft, Cars, and UCF, we follow prior protocols and sample 100 classes for continual learning evaluation. The split notation "B-x Inc-y" denotes that the first stage contains x classes and each later stage introduces y new classes. When $x = 0$, all tasks contain the same number of classes.

Implementation Details of M^{2}LE based on CLIP. We build M^{2}LE upon SimpleCIL [45], a strong training-free baseline that freezes the pre-trained model and constructs a prototypical classifier from downstream features. Specifically, we keep one frozen CLIP branch $f_{\mathrm{pt}}(\cdot)$ together with its prototype classifier $\mathbf{W}_{\mathrm{pt}}$, and sequentially adapt another CLIP branch $f_{\mathrm{ft}}(\cdot)$ using M^{2}LE. The adapted branch is trained with the CLIP-style image–text matching objective [46], where the text classifier is denoted as $\mathbf{W}_{\mathrm{text}}$. At inference time, we combine the predictions from the frozen branch and the adapted branch:

$$\mathbf{p} = \alpha \mathbf{W}_{\mathrm{pt}}^{\top} f_{\mathrm{pt}}(\mathbf{x}) + (1-\alpha)\mathbf{W}_{\mathrm{text}}^{\top} f_{\mathrm{ft}}(\mathbf{x}). \tag{C.6.1}$$

Comparison Methods. We compare M^{2}LE with a wide range of CLIP-based continual learning baselines, including prompt-based methods CoOp [47], L2P [39], DualPrompt [38], and CODA-Prompt [28]; CLIP-oriented methods RAPF [10] and PROOF [46]; the training-free SimpleCIL [45]; and the zero-shot CLIP baseline [26]. For fair comparison, all methods use the CLIP backbone pre-trained on LAION-400M, and no experience replay is used.

Results and Analysis. As shown in Tab. 11 and Tab. 12, M^{2}LE achieves the best performance across all CLIP-based settings, showing that it can generalize well to vision-language pre-trained models. Some adaptation baselines even perform worse than ZS-CLIP or SimpleCIL, indicating that directly updating CLIP may disturb its pre-trained knowledge. In contrast, M^{2}LE enables effective layer-wise continual adaptation through adaptive LoRA merging: HIM assigns element-wise merge strengths to better balance old and new knowledge, while TSP improves the stability of the learned update for later merging. Moreover, M^{2}LE is replay-free and does not require class-wise feature statistics, making it efficient and scalable as the label space grows.

D Limitations

Although M^{2}LE shows strong performance across diverse continual learning benchmarks, its effectiveness still depends on the quality and adaptability of the underlying pre-trained backbone. When the backbone provides less transferable representations or is not well suited to low-rank updates, the benefit of LoRA-based continual adaptation may be reduced.

E Potential Impact

M^{2}LE achieves efficient continual adaptation by updating only low-rank trainable parameters, which makes it suitable for long task sequences. However, this efficiency-oriented design may not always be optimal for accuracy-critical applications, such as medical imaging or other safety-sensitive scenarios. In such cases, practitioners should carefully evaluate whether the resulting performance meets the precision requirements of the target domain.

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: The abstract and introduction clearly state the research objectives, methods, and contributions, all of which are consistently elaborated and supported throughout the paper and supplementary materials.

   Guidelines:

   - The answer [N/A] means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A [No] or [N/A] answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: Please see Appendix D.

   Guidelines:

   - The answer [N/A] means that the paper has no limitation while the answer [No] means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

Answer: [Yes]

Justification: The paper provides clearly stated assumptions and complete derivations for the theoretical results. The corresponding formulas and proofs are presented with proper references, and additional details are included in Appendix when necessary.

Guidelines:

- The answer [N/A] means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

Answer: [Yes]

Justification: Our method and experimental setup are thoroughly described in Sec.5 to ensure reproducibility. Code for reproducing the main results is provided in the supplementary material.

Guidelines:

- The answer [N/A] means that the paper does not include experiments.
- If the paper includes experiments, a [No] answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
- If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
- Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
- While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
  (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
  (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
  (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
  (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

   Answer: [Yes]

   Justification: We provide the code in the supplementary material.

   Guidelines:

   - The answer [N/A] means that paper does not include experiments requiring code.
   - Please see the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - While we encourage the release of code and data, we understand that this might not be possible, so [No] is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
   - The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
   - The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
   - At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
   - Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer) necessary to understand the results?

   Answer: [Yes]

   Justification: Please see **Datasets**, **Evaluation Metrics** and **Implementation Details** in Sec.5.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [Yes]

   Justification: Our experiments use three random seeds, except for the analysis in Table 10, which uses a single seed. We report mean performance for the three seed experiments, and we report the average performance with standard deviation.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The authors should answer [Yes] if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.

- The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
- The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
- The assumptions made should be given (e.g., Normally distributed errors).
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g., negative error rates).
- If error bars are reported in tables or plots, the authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: We provide the information of our compute resources in Appendix.C.2.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification: Our research complies the NeurIPS Code of Ethics in all aspects.

   Guidelines:

   - The answer [N/A] means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer [No], they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [Yes]

    Justification: Please see Appendix E.

    Guidelines:

    - The answer [N/A] means that there is no societal impact of the work performed.

- If the authors answer [N/A] or [No], they should explain why their work has no societal impact or why the paper does not address societal impact.
- Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate Deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pre-trained language models, image generators, or scraped datasets)?

    Answer: [N/A]

    Justification: Our paper poses no such risks.

    Guidelines:

    - The answer [N/A] means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
    - Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
    - We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [Yes]

    Justification: The existing assets covered in the paper are subject to their license and terms of use, and are explicitly cited and described.

    Guidelines:

    - The answer [N/A] means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.
    - The name of the license (e.g., CC-BY 4.0) should be included for each asset.
    - For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.

- If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
- For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [Yes]

    Justification: We provide the code in the supplementary material.

    Guidelines:

    - The answer [N/A] means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [N/A]

    Justification: This paper does not involve crowdsourcing nor research with human subjects.

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
    - According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [N/A]

    Justification: This paper does not involve crowdsourcing nor research with human subjects.

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
    - Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.

- We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
- For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

    Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does *not* impact the core methodology, scientific rigor, or originality of the research, declaration is not required.

    Answer: [N/A]

    Justification: No, our paper does not involve the use of large language models (LLMs) as a component of the core methodology.

    Guidelines:

    - The answer [N/A] means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
    - Please refer to our LLM policy in the NeurIPS handbook for what should or should not be described.