S-LoRA: Scalable Low-Rank Adaptation for Class Incremental Learning

Anonymous authors
Paper under double-blind review

Abstract

Continual Learning (CL) with foundation models has recently emerged as a promising approach to harnessing the power of pre-trained models for sequential tasks. Existing prompt-based methods generally use a prompt selection mechanism to select relevant prompts aligned with the test query for further processing. However, the success of these methods largely depends on the precision of the selection mechanism, which also raises scalable issues with additional computational overhead as tasks increase. To overcome these issues, we propose a Scalable Low-Rank Adaptation (S-LoRA) method for class incremental learning, which incrementally decouples the learning of the direction and magnitude of LoRA parameters. S-LoRA supports efficient inference by employing the last-stage trained model for direct testing without the selection process. Our theoretical and empirical analysis demonstrates that S-LoRA tends to follow a low-loss trajectory that converges to an overlapped low-loss region, resulting in an excellent stability-plasticity trade-off in CL. Furthermore, based on our findings, we develop variants of S-LoRA with further improved scalability. Extensive experiments across multiple CL benchmarks and various foundation models consistently validate the effectiveness of S-LoRA.

1 Introduction

Continual Learning (CL) (Rolnick et al., 2019; Wang et al., 2024b; Zhou et al., 2024; Wang et al., 2022b) seeks to develop a learning system that can continually adapt to changing environments while retaining previously acquired knowledge. Unlike traditional supervised learning, which trains models on independent and identically distributed (i.i.d.) data, CL focuses on training models on non-stationary data distributions where tasks are presented sequentially. This deviation from the i.i.d assumption introduces the central challenge of mitigating catastrophic forgetting (French, 1999; McClelland et al., 1995; McCloskey & Cohen, 1989; Kirkpatrick et al., 2017), a phenomenon characterized by a significant decline in performance on previously learned tasks.

Over the last five years, large foundation models have demonstrated their effectiveness in enabling efficient knowledge transfer and showing greater resistance to catastrophic forgetting (Wang et al., 2022b;a; Smith et al., 2023a; Huang et al., 2024; Wang et al., 2024a; Liang & Li, 2024). As a result, leveraging these foundation models to further mitigate forgetting has emerged as a key research area recently. One popular line of research involves manipulating the *input and intermediate representations* (also referred to as the prompts) of foundation models. L2P (Wang et al., 2022b) and DualPrompt (Wang et al., 2022a) are pioneering methods that incrementally optimize a pool of prompts and selectively insert them into the model based on their match with the query task, specifically the input's encoding by the foundation model. Coda-Prompt (Smith et al., 2023a) builds on L2P and DualPrompt by optimizing the prompt selection module in an end-to-end manner. Although these methods do not require explicit task information, their effectiveness heavily relies on the ability to accurately identify the correct task-specific prompts from the pool. More recently, efforts to further enhance performance have led to HidePrompt (Wang et al., 2024a), which not only employs incremental learning of prompts but also requires the storage of extensive sample features. Similarly, InfLoRA (Liang & Li, 2024) requires the storage of large amounts of sample features, while it incrementally learns LoRA components instead of prompts. Consequently, due to their requirement for extensive sample features, neither method is rehearsal-free (Li & Hoiem, 2017; Smith et al., 2021; 2023b; Wang et al., 2022a;b), making them unsustainable.

Table 1: Comparisons of existing CL methods with foundation models. *Rehearsal-free* indicates that the methods do not require storing sample features from previous tasks. *End-to-end Optimization* means the model is trained in an end-to-end manner rather than through separate optimization processes. *Inference Efficiency* denotes the computational efficiency during the inference phase.

Method	**Rehearsal-free**	**End-to-end Optimization**	**Inference Efficiency**
L2P (Wang et al., 2022b)	✓	✗	✗
DualPrompt (Wang et al., 2022a)	✓	✗	✗
CodaPrompt (Smith et al., 2023a)	✓	✓	✗
HidePrompt (Wang et al., 2024a)	✗	✓	✗
InfLoRA (Liang & Li, 2024)	✗	✓	✓
S-LoRA(Ours)	✓	✓	✓

As shown in Table 1, an ideal CL method leveraging foundation models should possess three key properties, (1) *Rehearsal-free*: this represents the model does not need to store sample features from previously trained tasks, thereby ensuring sustainability during long continual training; (2) *End-to-end Optimization*: this property highlights that training the algorithm in an end-to-end manner, rather than with separate optimizations, typically leads to better performance; (3) *Inference Efficiency:* this property highlights the computation efficiency during the testing, where ideally no additional computational cost should be introduced. However, among current CL methods with foundation models, L2P (Wang et al., 2022b) and DualPrompt (Wang et al., 2022a) do not employ an end-to-end update strategy. Meanwhile, HidePrompt (Wang et al., 2024a) and InfLoRA (Liang & Li, 2024), in their pursuit of higher performance, require the storage of large amounts of sample features, which indicates that they are not rehearsal-free. With the exception of InfLoRA, all other prompt-based methods require the computation for prompt selection during testing. While this selection process allows for better utilization of task-specific information, it inevitably introduces extra computational costs during testing. Most importantly, there remains uncertainty about whether the correct task can be reliably identified from the query (Huang et al., 2024), particularly in the context of CL and considering the challenge of identifying query embeddings that are reliable yet resistant to forgetting.

To address these challenges faced by current CL methods, we propose S-LoRA, an effective **S**calable algorithm based on **Lo**w-**R**ank **A**daptation, which incrementally adds LoRA components while decoupling the learning of its direction and magnitude. By directly employing the final learned S-LoRA for testing, without the selective process of acquiring task-specific information, the model significantly increases its inference efficiency. Through in-depth theoretical and empirical analysis, we demonstrate that the proposed S-LoRA automatically learns a low-loss path converging to a shared low-loss region, thereby achieving excellent performance while eliminating the need to store sample features. Additionally, we observe that the importance of its incrementally learned LoRA directions gradually decreases during continual training. This insight has led us to develop enhanced, more efficient versions, which further optimize the parameter fine-tuning process. In summary, our contributions are as following four points:

- We propose the S-LoRA algorithm, which incrementally decouples the learning of LoRA's direction and magnitude. By avoiding any extra computational costs during testing, S-LoRA facilitates sample-independent inference and enhances inference efficiency.
- We conduct both theoretical and experimental analyses to investigate the underlying mechanism of the proposed S-LoRA, discovering that it automatically learns a low-loss path that converges to a shared low-loss region. This enables S-LoRA to achieve excellent performance without the need to store previous sample features, making it rehearsal-free.
- Based on the findings that the importance of its incrementally learned LoRA directions gradually decreases during continual training, we further develop enhanced and efficient versions of S-LoRA, referred to as ES-LoRA, to improve training efficiency and scalability.
- Our comprehensive experiments on various class incremental learning benchmarks across different backbones demonstrate that our S-LoRA and its efficient versions consistently outperform.

2 Related Work

Continual Learning. Continual learning aims to effectively acquire new task knowledge while preserving the knowledge learned from previous tasks during continual training. Traditional CL methods can be broadly divided into three categories: rehearsal-based, regularization-based, and architecture-based. Rehearsal-based methods (Chaudhry et al., 2019; Riemer et al., 2018; Tiwari et al., 2022) selectively retain samples or features from previous tasks to avoid catastrophic forgetting. In contrast, regularization-based methods (Kirkpatrick et al., 2017; Li & Hoiem, 2017; Lee et al., 2019) typically incorporate a quadratic regularization term to slow down the learning of weights that are important to prior tasks. Architecture-based methods (Mallya et al., 2018; Ebrahimi et al., 2020; Ramesh & Chaudhari, 2021) introduce extra task-specific parameters for each new task, thereby preventing the erasure of previously learned knowledge. This paper specifically concentrates on the highly impactful and challenging class-incremental learning (CIL), where the model must perform all tasks without access to the oracle of task identity during the testing process. In CIL, traditional CL methods often require learning from scratch and tuning many parameters, which can easily lead to overfitting and interference between tasks, ultimately resulting in significant catastrophic forgetting.

Continual Learning with Foundation Models. Foundation models have recently demonstrated their effectiveness in facilitating efficient knowledge transfer and reducing catastrophic forgetting in CL scenarios (Wang et al., 2022b;a; 2024a; Smith et al., 2023a; Liang & Li, 2024). And researchers have been exploring how to further mitigate forgetting for these models. For instance, methods like L2p (Wang et al., 2022b), Dual-Prompt (Wang et al., 2022a), Coda-Prompt (Smith et al., 2023a) utilize the Vistion Transformer and incorporate a prompt-tuning mechanism within CL. Building on this, Hide-Prompt (Wang et al., 2024a) not only implements incremental learning of prompts but also requires the storage of extensive sample features to enhance performance further. Similarly, InfLoRA (Liang & Li, 2024), a LoRA-based approach, also necessitates storing a considerable number of old sample features. However, none of the current CL methods using foundation models can simultaneously satisfy the three properties outlined in Table 1. Therefore, in this paper, we propose S-LoRA to address these limitations. Additionally, recent model-merging techniques (Chitale et al., 2023; Ilharco et al., 2023), while developed in different settings, provide valuable insights by combining task-specific vectors to mitigate forgetting, further complementing CL research.

Parameter Efficient Fine-Tuning (PEFT). Large-scale pre-trained foundation models have demonstrated exceptional adaptability across various downstream tasks. However, employing standard full fine-tuning for each task results in substantial computational and storage overhead, which often leads to overfitting. To address these challenges, researchers have turned to PEFT methods that can achieve comparable or even superior performance and generalizability by fine-tuning only a small set of parameters. For instance, adapter (Houlsby et al., 2019) incorporate additional modules into different layers of Transformer architectures. Similarly, prompt-tuning (Qin & Eisner, 2021; Jia et al., 2022) and prefix-tuning (Li & Liang, 2021) introduce learnable soft tokens at various layers of the Transformer input. Low-rank adaptation (LoRA) (Hu et al., 2021) adds low-rank branches to the pre-trained weights and tunes only these branches. Despite their advantages, common PEFT techniques are typically limited to static single-task and multi-task learning scenarios. In practical applications, however, models must continuously adapt and acquire new capabilities to navigate an ever-evolving environment, which necessitates that they can perform PEFT on an ongoing basis.

3 Preliminary

Problem Definition. Suppose there are N sequential tasks $\{\mathcal{T}_1, \mathcal{T}_2, ..., \mathcal{T}_N\}$, where each task $\mathcal{T}_i = \{\mathbf{x}_n^i, y_n^i\}_{n=1}^{N_i}$ consists of N_i training examples, with $\mathbf{x}_n^i$ representing the input image and y_n^i its corresponding label. Let $\mathcal{L}_i(\cdot)$ denote the empirical risk on i-th task $\mathcal{T}_i$, and f_θ represent the classification model. Then, the objective function of continual learning can be expressed as $\frac{1}{j}(\sum_{i=1}^{j} \mathcal{L}_i(\theta))$, where j is the index of current training task. The goal is for the model to perform well on both the current training task $\mathcal{T}_j$ and all previously learned tasks $\mathcal{T}_i, i \in \{1, 2, \ldots, j-1\}$. Following (Wang et al., 2022b; Liang & Li, 2024), in this paper, we primarily focus on class-incremental scenarios and adopt the pre-trained ViT as the initialized classification model f_θ.

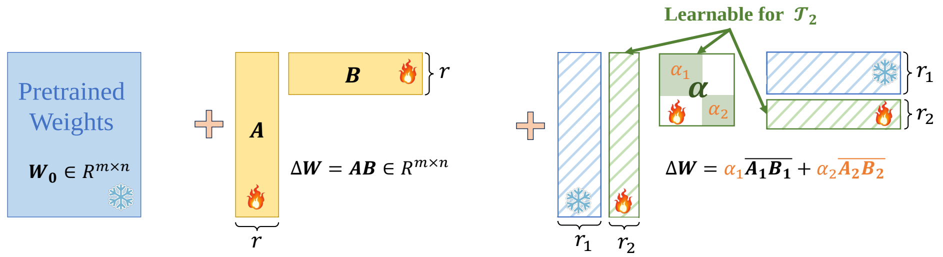
Low-Rank Adaptation. As illustrated in Fig. 1 (a), LoRA (Hu et al., 2022) assumes that changes in parameters $\Delta\mathbf{W}$ occur within a low-rank space when fine-tuning the layer weights $\mathbf{W_0} \in \mathbb{R}^{m\times n}$ of the model f_θ for a downstream task. Specifically, the parameter update is expressed as $\Delta\mathbf{W} = \mathbf{A} \times \mathbf{B}$, where $\mathbf{A} \in \mathbb{R}^{m\times r}$ and $\mathbf{B} \in \mathbb{R}^{r\times n}$ are two learnable matrices with $r \ll \min\{m, n\}$. For a given layer of the model f_θ, the LoRA update represented is,

$$h' = \mathbf{W}_0 x + \Delta\mathbf{W} x = (\mathbf{W}_0 + \mathbf{AB})x$$

where h' denotes the modified output and the original weights $\mathbf{W}_0$ is frozen during fine-tuning.

4 Method

In this section, we first delve into details of the proposed S-LoRA algorithm. Next, we examine the underlying mechanisms of S-LoRA and explain the reason why it can achieve excellent performance in the context of CL. Finally, building on the analyses, we propose several efficient versions of S-LoRA designed to improve training efficiency while maintaining performance.


Pretrained Weights
$\boldsymbol{W_0} \in R^{m\times n}$
$\boldsymbol{B}$
r
$\boldsymbol{A}$
$\Delta \boldsymbol{W} = \boldsymbol{AB} \in R^{m\times n}$
r
Learnable for $\mathcal{T}_2$
α_1
$\boldsymbol{\alpha}$
α_2
r_1
r_2
$\Delta \boldsymbol{W} = \alpha_1 \overline{\boldsymbol{A_1B_1}} + \alpha_2 \overline{\boldsymbol{A_2B_2}}$
r_1
r_2

(a) LoRA (b) S-LoRA

Figure 1: Illustration of weight updates in vanilla LoRA and the proposed S-LoRA within continual learning scenarios. (a) depicts vanilla low-rank adaptation with $r \ll \min\{m, n\}$. (b) demonstrates S-LoRA's learning process across two sequential tasks, where $\overline{\boldsymbol{A}_i\boldsymbol{B}_i} = \frac{\boldsymbol{A}_i\boldsymbol{B}_i}{\|\boldsymbol{A}_i\boldsymbol{B}_i\|}$ $(i = 1, 2)$ represents the normalized direction and the vector $\boldsymbol{\alpha}$ adjusts the magnitude, with $r_1, r_2 \ll \min\{m, n\}$.

4.1 The Proposed S-LoRA Algorithm

As demonstrated in Sec. 3, LoRA's parameter updates $\Delta\mathbf{W} = \|\mathbf{AB}\| \cdot \overline{\mathbf{AB}} = \|\mathbf{AB}\| \cdot \frac{\mathbf{AB}}{\|\mathbf{AB}\|}$ consists of two components: magnitude (i.e., $\|\mathbf{AB}\|$) and direction (i.e., $\overline{\mathbf{AB}}$). However, recent studies (Liu et al., 2024) have revealed that, compared to full fine-tuning, LoRA lacks the fine-grained capability to make precise adjustments to both components. This limitation hinders its performance on overly complex tasks requiring precise direction and magnitude control. Additionally, other research (Qiu et al., 2023) has indicated that direction is more critical than magnitude during fine-tuning. Motivated by these works and considering the inference efficiency of LoRA-based methods, we make an initial attempt to propose the S-LoRA algorithm within the context of continual learning. This approach seeks to incrementally decouple the learning of direction and magnitude within LoRA while maintaining the directions learned from previous tasks during the continual training.

Concretely, let $\boldsymbol{\alpha} = \{\alpha_1, \alpha_2, \ldots, \alpha_j\}$ represent a list of scaling factors corresponding to the learnable magnitudes, and let $\overline{\mathbf{A}_i\mathbf{B}_i}, i \in \{1, 2, \ldots, j-1\}$ denote the previously learned directions for task $\mathcal{T}_i$. During the training of the current task $\mathcal{T}_j$, as illustrated in Fig. 1 (b), the output of a given layer using our proposed S-LoRA approach can be formulated as follows:

$$h' = (\mathbf{W_0} + \alpha_1 \overline{\mathbf{A}_1\mathbf{B}_1} + \alpha_2 \overline{\mathbf{A}_2\mathbf{B}_2} + ... + \alpha_j \overline{\mathbf{A}_j\mathbf{B}_j})x, \tag{1}$$

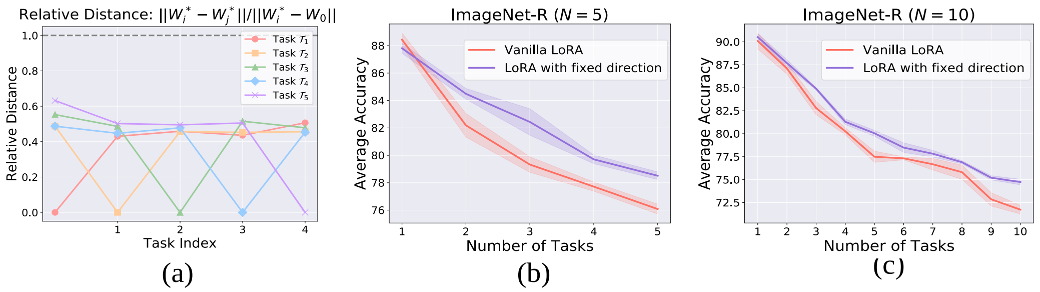
where the colored terms $\boldsymbol{\alpha} = \{\alpha_1, \alpha_2, \ldots, \alpha_j\}$ and $\overline{\mathbf{A}_j\mathbf{B}_j}$ are learnable, while the remaining parameters $\mathbf{W_0}$, $\overline{\mathbf{A}_i\mathbf{B}_i}, i \in \{1, 2, ..., j-1\}$ remain fixed during the training on the current task $\mathcal{T}_j$.

By incrementally decoupling magnitude and direction while preserving the directions of previously learned tasks, as shown in Eqn. (1), we were surprised to observe a significant performance improvement across various CL benchmarks. Detailed experimental results are provided in Sec. 6. To explore this intriguing phenonmenon, we will conduct a thorough analysis of why the simple decoupling approach used in S-LoRA leads to such excellent performance

4.2 Analysis of our Proposed S-LoRA

Although the proposed S-LoRA demonstrates impressive performance gains on CL benchmarks, the reasons behind this success—particularly how it alleviates the forgetting problem when the model is sequentially trained on different tasks—remain unclear. To shed light on this, we conducted comprehensive experiments and have summarized the key insights into the following three findings.

Finding 1: *The optimal weights for downstream tasks fine-tuned from the foundation model are closer to each other than to the foundation model's weights.* As shown in Fig. 2 (a), using the five

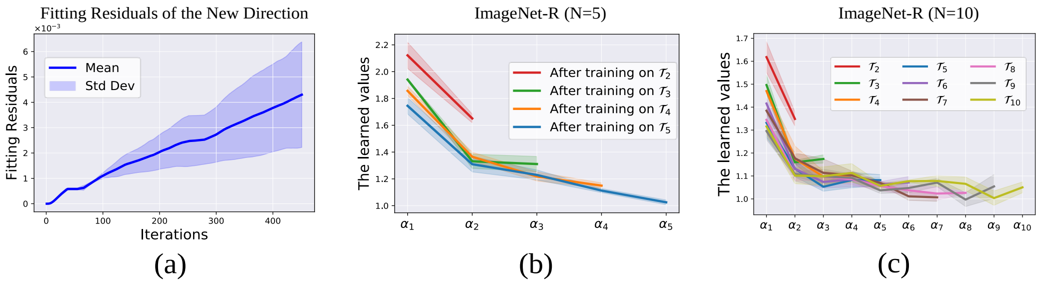


(a) (b) (c)

Figure 2: (a) illustrates the relative distances between the five optimal weights $\mathbf{W}_i^*$ on ImageNet-R ($N = 5$) and the foundation model weights $\mathbf{W}_0$. All values being largely less than 1 indicate that the five optimal weights are closer to each other than to the foundation model weights $\mathbf{W}_0$; (b) and (c) compare the performance of Vanilla LoRA and LoRA with the first learned direction fixed, evaluated on ImageNet-R across 5 tasks and 10 tasks, respectively.

tasks from ImageNet-R as an example, we computed the optimal weights for each task by fine-tuning directly from the foundation model. When comparing the distances between the task-specific weights and the foundation model weights, it is evident that the optimal weights for different tasks are closer to each other in parametric space than to the foundation model weights.

Additionally, we conducted an experiment where only the magnitude was learned while keeping the first learned direction fixed. In this setup, the modified output for a given layer is $h' = \mathbf{W}_0 + \alpha_1 \overline{\mathbf{A}_1\mathbf{B}_1}$ across all N sequentially trained tasks. The final average results across all sequential tasks, shown in Fig. 2 (b)(c), remarkably outperform vanilla LoRA (i.e., $h' = \mathbf{W}_0 + \mathbf{AB}$). This further indicates that the fine-tuned optimal weights for different tasks are close to each other. Even with one fixed learned update direction, we can achieve relatively good performance across all tasks. It is worth noting that this finding aligns with (Entezari et al., 2022; Gueta et al., 2023), which suggests that the fine-tuned model weights for similar tasks derived from a foundation model lie within a near low-loss region.



(a) (b) (c)

Figure 3: Analysis of the learning process of S-LoRA. (a) depicts the least squares fitting residuals of the newly learned direction $\overline{\mathbf{A}_j\mathbf{B}_j}$ relative to all previous $\overline{\mathbf{A}_i\mathbf{B}_i}$ during iterative training, while (b) and (c) illustrate the gradually learned $\boldsymbol{\alpha}$ values on ImageNet-R with 5 and 10 tasks, respectively.

Finding 2: *The preserved directions from previously learned tasks (i.e.,* $\mathbf{A}_i\mathbf{B}_i, i \in \{1, 2, \ldots, j-1\}$*) are reused, and the initial ones play a significant role in the learning process.* To investigate the effect of the learnable scaling factors $\boldsymbol{\alpha}$ and the newly learned direction of the current task $\overline{\mathbf{A}_j\mathbf{B}_j}$, we visualized the learned $\boldsymbol{\alpha}$ values and the linear correlations between the newly learned $\overline{\mathbf{A}_j\mathbf{B}_j}$ with previously learned $\overline{\mathbf{A}_i\mathbf{B}_i}, i \in \{1, 2, .., j-1\}$ in Fig. 3. It can be observed that the fitting residuals of $\overline{\mathbf{A}_j\mathbf{B}_j}$ using the set of $\overline{\mathbf{A}_i\mathbf{B}_i}$ increase as training progresses, starting small and gradually growing larger. This suggests that, in the early stages, the newly learned direction aligns closely with previous ones, enabling the model to primarily reuse earlier directions. However, as training progresses, the new direction diverges, incorporating subtle variations from the earlier learned ones.

To further analyze the importance of the learned directions, we visualized the learned magnitude $\boldsymbol{\alpha}$ values and observed a downward trend in $\{\alpha_1, \alpha_2, \ldots, \alpha_j\}$, as shown in Fig. 3 (b)(c). For more results, please refer to Appendix A.2. This trend suggests that, as the continual training progresses, the

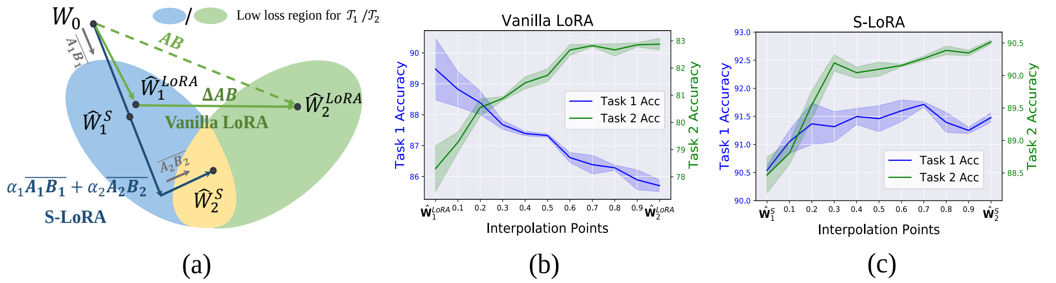


Figure 4: Comparison of SE-LoRA and Vanilla LoRA Learning Pathways: (a) A simplified illustration of the learning trajectories for S-LoRA and Vanilla LoRA across two sequential tasks. $\mathbf{W}_0$ represents the foundation model, while $\hat{\mathbf{W}}_i^S$ and $\hat{\mathbf{W}}_i^{LoRA}$ denote the learned weights after training on $\mathcal{T}_i$; (b) The interpolation accuracy along the vanilla LoRA path shows improved performance on $\mathcal{T}_2$ but a decline in $\mathcal{T}_1$, indicating it fails to reach the shared low-loss region; (c) In contrast, the interpolation accuracy along the S-LoRA path shows it finds a low-loss route to a shared region.

model increasingly relies on directions learned from earlier tasks, while the newly learned direction primarily serves as a subtle adjustment for the new task.

Finding 3: *The learning process of S-LoRA essentially identifies a low-loss path by leveraging the fixed directions from previously learned tasks and the learnable* $\boldsymbol{\alpha}$*, ultimately reaching a low-loss region shared by different tasks.* Based on the aforementioned two findings, we hypothesize that S-LoRA adjusts the magnitude of the learned fixed directions to identify a low-loss path (Doan et al., 2023) that ultimately converges to a shared low-loss region for different tasks. To test this hypothesis, we conduct interpolation experiments to examine the linear low-loss path among the two sequentially learned model weights of S-LoRA. As shown in Fig. 4, along the linear path from S-LoRA-learned $\hat{\mathbf{W}}_1^S$ to $\hat{\mathbf{W}}_2^S$, the performance on $\mathcal{T}_2$ improves without any degradation in accuracy on $\mathcal{T}_1$. In contrast, vanilla LoRA enhances performance on $\mathcal{T}_2$ at the cost of $\mathcal{T}_1$. This indicates that S-LoRA adjusts the magnitude of updates in the learned directions, allowing the model to find a low-loss path that ultimately leads to a shared low-loss region across tasks, as illustrated in Fig. 4 (a).

Based on the above findings, we can outline the reasons why S-LoRA is effective in the continual learning setting. First, it makes significant updates along the key directions learned from earlier tasks, rapidly approaching the shared low-loss region for multiple tasks. Then, by incrementally introducing LoRA, it fine-tunes these directions, allowing the model to accurately converge on the shared low-loss region for different tasks. This strategy of seeking a low-loss path enables S-LoRA to effectively identify the shared low-loss region, eliminating the need to store features from the trained samples.

4.3 The Efficient Versions of S-LoRA

From the analysis in Sec. 4.2, we find that during sequential training, the first few directions learned from previous tasks are particularly important and are consistently reused, while later directions mainly serve as subtle adjustments. As tasks are learned sequentially, the set of learned directions (i.e., $\{\overline{\mathbf{A}_i\mathbf{B}_i}\}, i \in \{1, 2, ..., j-1\}$) expands significantly. However, this expansion increasingly comprises directions that function mainly as minor adjustments, thereby contributing less to the overall performance. Consequently, the necessity for introducing entirely new directions diminishes, as the existing ones can effectively act as substitutes. With this insight, we propose the following two efficient versions, ES-LoRA, which adopt the *dynamic rank* and *knowledge distillation*, respectively.

ES-LoRA1. (*Dynamic Rank*) As shown in Fig. 1 (b), let r_i denotes the lower dimension of matrices $\mathbf{A}_i$ and $\mathbf{B}_i$, i.e., $\mathbf{A}_i \in \mathbb{R}^{m\times r_i}, \mathbf{B}_i \in \mathbb{R}^{r_i\times n}$. To enhance the efficiency of incremental training, we employ a piecewise reduction to decrease the rank of the later learned direction matrices during training. Specifically, we establish a dynamic rank for the sequential matrices $\mathbf{A}_i$ and $\mathbf{B}_i$, with b_1 and b_2 as critical task indices where the rank decreases, and r_i as a set of hyperparameters.

$$r_1 = r_2 = ... > r_{b_1} = r_{b_1+1} = ... > r_{b_2} = r_{b_2+1} = ... = r_N.$$

ES-LoRA2. (*Knowledge Distillation*) While the *Dynamic Rank* strategy helps reduce model expansion, it still necessitates an increase in parameters with each new task. To further address this issue,

Algorithm 1 The Algorithm of the proposed S-LoRA and its Efficient Variants.

```
Input: Foundation model $\mathbf{W}_0$; current training task $\mathcal{T}_j$, $j \in \{1, ..., N\}$; scaling factors list $\boldsymbol{\alpha} = \{\alpha_1, ..., \alpha_{j-1}\}$; previously learned directions $\{\overline{\mathbf{A}_i\mathbf{B}_i}\}, i = 1, 2, ..., j-1$; threshold $\tau$; epochs $M$.
Output: The updated $\{\overline{\mathbf{A}_i\mathbf{B}_i}\}$ and scaling factor lists $\boldsymbol{\alpha}$.
Sample examples $\{x_b, y_b\}_{b=1}^{B}$ from the current training task $\mathcal{T}_j$, i.e., $\{x_b, y_b\}_{b=1}^{B} \sim \mathcal{T}_j$
Compute the Cross-Entropy loss $\mathcal{L}$ on the sampled $\{x_b, y_b\}_{b=1}^{B}$ with the ViT model $f_\theta$ with SE-LoRA as shown in Eqn. (1) (i.e., $\mathbf{W}_0 + \alpha_1\overline{\mathbf{A}_1\mathbf{B}_1} + \alpha_2\overline{\mathbf{A}_2\mathbf{B}_2} + ... + \alpha_j\overline{\mathbf{A}_j\mathbf{B}_j}$ ).
if $j = b_1$ / $b_2$ then                                          / Only for the ES-LoRA1
    Reduce the lower dimension $r$ of $\mathbf{A}_j$ and $\mathbf{B}_j$ to $r_1$ / $r_2$, respectively.
end if
for epoch $= 0$ to $M$ do
    Update the learnable parameters $\boldsymbol{\alpha}$ and $\overline{\mathbf{A}_j\mathbf{B}_j}$
end for
if the fitting residual of $\overline{\mathbf{A}_j\mathbf{B}_j}$ using $\{\overline{\mathbf{A}_i\mathbf{B}_i}\} < \tau$ then     / Only for the ES-LoRA2
    Add the corresponding fitting coefficients to $\boldsymbol{\alpha}$ and discard $\overline{\mathbf{A}_j\mathbf{B}_j}$.
else
    Add $\overline{\mathbf{A}_j\mathbf{B}_j}$ to the set $\{\overline{\mathbf{A}_i\mathbf{B}_i}\}$
```

we employ least squares fitting to assess whether the newly learned direction $\overline{\mathbf{A}_j\mathbf{B}_j}$ can be substituted by previously learned directions $\{\overline{\mathbf{A}_i\mathbf{B}_i}\}, i \in \{1, 2, ..., j-1\}$. If a relatively high correlation is found, the coefficients from this fitting will be added to the corresponding $\boldsymbol{\alpha}$ values.

Concretely, after training the current tasks as shown in Eqn. (1), we approximate $\overline{\mathbf{A}_j\mathbf{B}_j}$ using previously learned $\{\overline{\mathbf{A}_i\mathbf{B}_i}\}, i \in \{1, 2, ..., j-1\}$. The residual is calculated as follows:

$$r = \|\overline{\mathbf{A}_j\mathbf{B}_j} - \sum_{i=1}^{j-1} \hat{\alpha}_i \overline{\mathbf{A}_i\mathbf{B}_i}\|$$

where $\hat{\alpha}_i$ represents the fitting coefficients. If the residual r is less than the threshold τ, we will add $\hat{\alpha}_i$ to the corresponding α_i in Eqn. (1). That is,

$$h' = \{\mathbf{W_0} + (\alpha_1 + \hat{\alpha}_1)\overline{\mathbf{A}_1\mathbf{B}_1} + (\alpha_2 + \hat{\alpha}_2)\overline{\mathbf{A}_2\mathbf{B}_2} + ... + (\alpha_{j-1} + \hat{\alpha}_{j-1})\overline{\mathbf{A}_{j-1}\mathbf{B}_{j-1}}\}x,$$

The pseudocode for S-LoRA, ES-LoRA1, and ES-LoRA2 is presented in Algorithm 1.

5 Theoretical Analysis

As empirically demonstrated in Sec. 4.2, the initially learned directions of S-LoRA are not only reused but also play a pivotal role in sequential training. In this section, we present a theoretical analysis to explain why these first several learned directions are so critical during S-LoRA training.

Let $\Delta\mathbf{W}^* \in \mathbb{R}^{m\times n}$ denote the optimal weights that lie in the shared low-loss region among all N sequential tasks. Let $\Delta\mathbf{W}_i^*, i \in \{1, 2, 3, ..., N\}$ denote the optimal weights specific to the low-loss region of $\mathcal{T}_i$. The singular values of $\Delta\mathbf{W}^*$ are represented by $\sigma_1 \geq ... \geq \sigma_{min\{m,n\}} \geq 0$. The matrices $\mathbf{A} \in \mathbb{R}^{m\times r}$, $\mathbf{B} \in \mathbb{R}^{r\times n}$ are iteratively updated, where the initial values $(\mathbf{A}_0, \mathbf{B}_0) = \frac{\rho}{3\sqrt{m+n+r}}(\tilde{\mathbf{A}}_0, \tilde{\mathbf{B}}_0)$, with $\tilde{\mathbf{A}}_0, \tilde{\mathbf{B}}_0$ having i.i.d. entries drawn from $N(0, \sigma_1)$. Let $k \in [0, \min\{k, m, n\}]$ and define the k-th condition number $\kappa_k := \frac{\sigma_1}{\sigma_k}$. The notation $\|\cdot\|_o$ refers to the operator norm.

Theorem 1. *Suppose the assumptions in Appendix A.1 hold, where ϵ_1 is a small number and let $\delta \in (0, 1)$ such that $\delta \leq \min_{1\leq s\leq k}\{\frac{\sigma_s - \sigma_{s+1}}{\sigma_s}\}$. Fix any tolerance $\epsilon_2 \leq \frac{1}{m+n+r}$, and let η denote the learning rate to update the matrix $\mathbf{A}$ and $\mathbf{B}$. $\Delta\mathbf{W}_{(s)}^*$ and $\Delta\mathbf{W}_{i(s)}^*$ are the s-th principle component of $\Delta\mathbf{W}^*$ and $\Delta\mathbf{W}_i^*$, respectively. Then, there exist some numerical constants c, c' and a sequence of iteration indices*

$$\mathbf{T}^{(1)} \leq \mathbf{T}^{(2)} \leq ... \leq \mathbf{T}^{(k)} \leq \frac{c'}{\delta\eta\sigma_k}\log(\frac{\kappa_k}{\delta\epsilon_2})$$

such that with high probability, the gradient descent with stepsize $\eta \leq c\min\{\delta, 1-\delta\}\frac{\sigma_k^2}{\sigma_1^3}$ and initialization size $\rho \leq (\frac{c\delta\epsilon_2}{\kappa_k})^{\frac{1}{c\delta}}$ satisfy

$$\|\mathbf{A}_{\mathbf{T}^{(s)}}\mathbf{B}_{\mathbf{T}^{(s)}} - \Delta\mathbf{W}_{(s)}^*\|_o \leq \epsilon_2\sigma_1 + \epsilon_1, \quad \forall s = 1, 2, ..., k, \tag{2}$$

Table 2: Average performance comparison on ImageNet-R across different task lengths, including standard deviation. All reported values are the mean of 5 runs.

Method	ImageNet-R ($N = 5$)		ImageNet-R ($N = 10$)		ImageNet-R ($N = 20$)	
	Acc ↑	AAA ↑	Acc ↑	AAA ↑	Acc ↑	AAA ↑
Fine-Tuning	64.92$_{(0.87)}$	75.57$_{(0.50)}$	60.57$_{(1.06)}$	72.31$_{(1.09)}$	49.95$_{(1.31)}$	65.32$_{(0.84)}$
L2P	73.04$_{(0.71)}$	76.94$_{(0.41)}$	71.26$_{(0.44)}$	76.13$_{(0.46)}$	68.97$_{(0.51)}$	74.16$_{(0.32)}$
Dual-Prompt	69.99$_{(0.57)}$	72.24$_{(0.41)}$	68.22$_{(0.20)}$	73.81$_{(0.39)}$	65.23$_{(0.45)}$	71.30$_{(0.16)}$
Coda-Prompt	76.63$_{(0.27)}$	80.30$_{(0.28)}$	74.05$_{(0.41)}$	78.14$_{(0.39)}$	69.38$_{(0.33)}$	73.95$_{(0.63)}$
Hide-Prompt	74.77$_{(0.25)}$	78.15$_{(0.24)}$	74.65$_{(0.14)}$	78.46$_{(0.18)}$	73.59$_{(0.19)}$	77.93$_{(0.19)}$
InfLoRA	76.95$_{(0.23)}$	81.81$_{(0.14)}$	74.75$_{(0.64)}$	80.67$_{(0.55)}$	69.89$_{(0.56)}$	76.68$_{(0.57)}$
S-LoRA	**79.15**$_{(0.20)}$	**83.01**$_{(0.42)}$	**77.34**$_{(0.35)}$	**82.04**$_{(0.24)}$	**75.26**$_{(0.37)}$	80.22$_{(0.72)}$
ES-LoRA1	79.01$_{(0.26)}$	82.50$_{(0.38)}$	77.18$_{(0.39)}$	81.74$_{(0.24)}$	74.05$_{(0.51)}$	**80.65**$_{(0.35)}$
ES-LoRA2	78.85$_{(0.29)}$	82.47$_{(0.58)}$	77.03$_{(0.67)}$	81.52$_{(0.26)}$	74.12$_{(0.66)}$	80.11$_{(0.75)}$

In Theorem 1, we interpret the learning process of S-LoRA as a matrix factorization problem and prove that gradient descent with small initialization leads the learned $\mathbf{AB}$ to sequentially approximate the principal components of $\Delta\mathbf{W}^*$ (i.e., $\Delta\mathbf{W}^*_{(1)}, \Delta\mathbf{W}^*_{(2)}, ..., \Delta\mathbf{W}^*_{(k)}$) sequentially. This provides a theoretical explanation for the decreasing trend in the learned $\boldsymbol{\alpha}$, as observed in Sec. 4.2, and also illustrates the feasibility of our proposed efficient versions, ES-LoRA.

6 Experiments

6.1 Experimental Settings

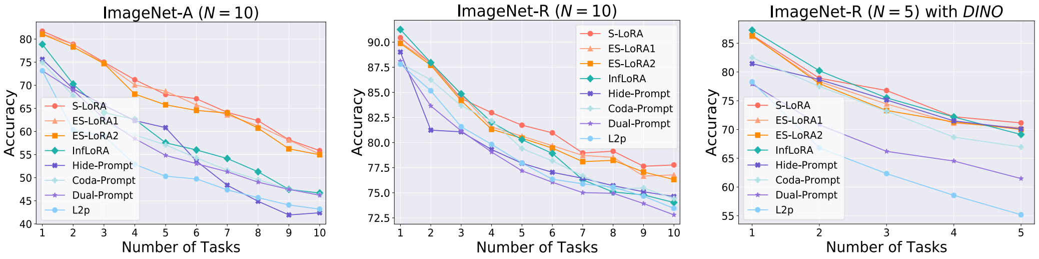
Datasets and Evaluation Metrics. To verify the effectiveness of the proposed SE-LoRA and its efficient variants, following (Gao et al., 2023; Liang & Li, 2024), we conduct comprehensive experiments on several benchmarks: ImageNet-R (Boschini et al., 2022), ImageNet-A (Hendrycks et al., 2021) and DomainNet (Peng et al., 2019). Specifically, ImageNet-R consists of 200 classes from ImageNet (Deng et al., 2009), with images altered using various artistic styles. Similarly, ImageNet-A also contains 200 classes but focuses on natural adversarial examples, which are often misclassified by standard ImageNet-trained models. DomainNet includes 345 classes across six distinct domains, making it a widely used benchmark in CL tasks. In line with recent CL studies (Liang & Li, 2024; Huang et al., 2024), we split ImageNet-R into 5, 10, 20 tasks, with each task containing 40, 20 and 10 classes, respectively. For ImageNet-A, we divide the dataset into 10 tasks, each containing 20 classes. For DomainNet, we create 5 tasks, each comprising 69 classes. Additionally, we also conducted experiments on other datasets, including CIFAR100 (Krizhevsky et al., 2009) and CUB200 (Wah et al., 2011a) which details and results can be found in Appendix A.3.

Following established CL methods (Wang et al., 2024a; Liang & Li, 2024), we adopt two widely used metrics: final averaged accuracy $Acc = AA_N$ and average anytime accuracy $AAA = \frac{1}{N}(\sum_{j=1}^{N}(AA_j)$. Here, AA_i denotes the accuracy on all seen tasks after completing training on task $\mathcal{T}_i$, and N is the total number of tasks. The final averaged accuracy (Acc) reflects the overall performance across all N tasks, while the average anytime accuracy (AAA) captures the model's average performance throughout the entire sequential learning process.

Baselines and Training Details. To emphasize the superior performance of S-LoRA and ES-LoRA, we compare it against state-of-the-art ViT-based CL methods, including L2p (Wang et al., 2022b), Dual-Prompt (Wang et al., 2022a), Coda-Prompt (Smith et al., 2023a), Hide-Prompt (Wang et al., 2024a), and InfLoRA (Liang & Li, 2024). Following prior work (Gao et al., 2023; Huang et al., 2024), we adopt the ViT-B/16 (Dosovitskiy, 2020) backbone, pre-trained on ImageNet-21k and fine-tuned on ImageNet-1K. To maintain generality, we also experimented with DINO (Caron et al., 2021), a self-supervised model trained under the ViT-B/16 framework. For all methods, following (Wang et al., 2022b; Liang & Li, 2024), we use the Adam (Kingma, 2014) optimizer with $\beta_1 = 0.9, \beta_2 = 0.999$. Training is conducted for 30 epochs on ImageNet-R, 10 epochs on DomainNet, and 20 epochs on the remaining datasets, with a batch size of 128 across all experiments. To ensure a fair comparison, we limit the number of class prototypes stored by Hide-Prompt, which typically retains a large amount of sample features, while our approach does not require such storage. The S-LoRA modules are inserted into all transformer blocks, and we set the rank $r = 10$. For the efficient variants ES-LoRA,

Table 3: Average performance comparison on ImageNet-A and DomainNet datasets, including standard deviation. All reported values are the mean of 5 runs.

Method	ImageNet-A ($N = 10$)		DomainNet($N = 5$)	
	Acc↑	AAA ↑	Acc ↑	AAA ↑
Fine-Tuning	$16.31_{(7.89)}$	$30.04_{(13.18)}$	$51.46_{(0.47)}$	$67.08_{(1.13)}$
L2P (Wang et al., 2022b)	$42.94_{(1.27)}$	$51.40_{(1.95)}$	$70.26_{(0.25)}$	$75.83_{(0.98)}$
Dual-Prompt (Wang et al., 2022a)	$45.49_{(0.96)}$	$54.68_{(1.24)}$	$68.26_{(0.90)}$	$73.84_{(0.45)}$
Coda-Prompt (Smith et al., 2023a)	$45.36_{(0.78)}$	$57.03_{(0.94)}$	$70.58_{(0.53)}$	$76.68_{(0.44)}$
Hide-Prompt (Wang et al., 2024a)	$42.70_{(0.60)}$	$56.32_{(0.40)}$	$72.20_{(0.08)}$	$77.01_{(0.04)}$
InfLoRA (Liang & Li, 2024)	$49.20_{(1.12)}$	$60.92_{(0.61)}$	$71.59_{(0.23)}$	$78.29_{(0.50)}$
S-LoRA	**55.96**$_{(0.73)}$	**64.95**$_{(1.63)}$	**72.82**$_{(0.37)}$	**78.89**$_{(0.50)}$
ES-LoRA1	$55.59_{(1.08)}$	$64.59_{(1.91)}$	$72.58_{(0.40)}$	$78.79_{(0.78)}$
ES-LoRA2	$54.24_{(1.12)}$	$63.89_{(0.58)}$	$72.15_{(0.50)}$	$78.44_{(0.66)}$



(a) (b) (c)

Figure 5: The detailed average accuracy during sequential training: (a) presents the Acc over 10 tasks sequentially on ImageNet-A, while (b) displays the corresponding Acc on ImageNet-R. (c) illustrates the performance using an alternative backbone, ViT-B/16, pre-trained with DINO.

we set the hyperparameters as follows: $b_1 = 4$, $b_2 = 8$, the dynamic rank as $\{10, 8, 6\}$. Moreover, the threshold value τ for the residual part is set to 9e-4.

6.2 Experimental Results

Performance on different CL benchmarks and backbones. To demonstrate the effectiveness of the proposed S-LoRA, we perform experiments across various CL benchmarks, as summarized in Table 2 and Table 3. The results clearly show that S-LoRA surpasses InfLoRA by nearly 7.68% in Acc 4.62% in AAA on ImageNet-R($N = 20$) and outperforms Hide-prompt by about 31.05% in Acc 15.32% in AAA on ImageNet-A. Even on the more complex DomainNet dataset, which spans six distinct domains, S-LoRA consistently achieves the best performance across both Acc and AAA metrics. Furthermore, unlike InfLoRA and Hide-Prompt, which store sample features from previous tasks, S-LoRA does not require feature storage, further highlighting its effectiveness across different continual learning benchmarks. To demonstrate the generalizability of ESLoRA, we also evaluate it using the ViT-B/16 backbone pre-trained with DINO, a self-supervised learning method. The results, as shown in Fig. 5 (c), reveal that S-LoRA continues to deliver the best performance in terms of both AAA and Acc on various backbones. Results on CIFAR100 and CUB please refer to Appendix.

Performance with different task length. To verify the performance of S-LoRA across different task length settings, we follow (Liang & Li, 2024; Huang et al., 2024) and split ImageNet-R into 5, 10, and 20 tasks, with each task containing 40, 20, and 10 classes, respectively. The results, shown in Table 2, demonstrate that S-LoRA consistently achieves the best performance on both Acc and AAA metrics across all task lengths. Even in the challenging 20-task setting, S-LoRA outperforms InfLoRA by approximately 7.68% in Acc and 4.62% in AAA. Notably, although the efficient variants of S-LoRA reduce the number of trainable parameters, their performance only slightly decreases while still surpassing other methods. For instance, ES-LoRA1 outperforms InfLoRA by 3.30% in

Table 4: The ablation study of the proposed S-LoRA. All reported values with standard deviation are from 5 independent runs. The colored notations indicate the trainable parameters during training.

Different Training Ways	ImageNet-R ($N = 5$)		ImageNet-R ($N = 10$)	
	Acc ↑	AAA ↑	Acc ↑	AAA ↑
$\mathbf{W}_0 + \alpha\overline{\mathbf{A}_1\mathbf{B}_1}$	$78.17_{(0.27)}$	$81.93_{(0.51)}$	$74.82_{(0.96)}$	$80.63_{(0.63)}$
$\mathbf{W}_0 + \alpha\overline{\mathbf{A}\mathbf{B}}$	$73.24_{(0.31)}$	$78.80_{(0.13)}$	$70.62_{(0.78)}$	$76.32_{(0.16)}$
$\mathbf{W}_0 + \mathbf{A}_1\mathbf{B}_1 + ... + \mathbf{A}_{j-1}\mathbf{B}_{j-1} + \alpha\overline{\mathbf{A}_j\mathbf{B}_j}$	$78.28_{(0.59)}$	$82.02_{(0.71)}$	$74.29_{(0.32)}$	$79.74_{(0.71)}$
S-LoRA	**79.15**$_{(0.20)}$	**83.01**$_{(0.42)}$	**77.34**$_{(0.35)}$	**82.04**$_{(0.24)}$

Table 5: Comparison of the number of trainable parameters and FLOPs for ImageNet-R($N = 20$).

Method	FLOPs (G) ↓	Trainable Parameters (M) ↓	Stored Features (M) ↓
L2P	70.14	0.48	0
Dual-Prompt	70.26	0.06	0
Coda-Prompt	70.61	0.38	0
Hide-Prompt	70.36	0.08	0.15
InfLoRA	35.12	0.37	0.10
S-LoRA	35.12	0.37	0
ES-LoRA	35.12	0.23	0

Acc and 1.32% in AAA on ImageNet-R ($N = 10$). Similarly, ES-LoRA2 surpasses InfLoRA by 3.05% in Acc and 1.05% in AAA on the same dataset.

Ablation study. To verify the significance of each component of the proposed S-LoRA, we conducted detailed experiments, as shown in Table 4. The colored notations indicate the trainable parameters during training. (1) **Fixing the First Learned Direction:** In this setup, we fix the first learned direction and only learn its magnitude. Although this approach yields worse performance compared to S-LoRA, it still outperforms some state-of-the-art methods listed in Table 2. This further emphasizes the importance of the first learned direction, aligning with the findings in Sec. 4.2 and Theorem 1. (2) **Decoupling Direction and Magnitude:** When we decouple the direction and magnitude of LoRA while only learning a single LoRA, the performance is inferior to that of S-LoRA. This result indicates that the significant performance gains are not solely attributed to the decoupling strategy. (3) **Fixing the Previously Learned LoRA:** By fixing all previously learned LoRA components without scaling their magnitudes, we observe a decline in performance compared to S-LoRA. This outcome highlights the importance of scaling the previously learned direction (i.e., $\boldsymbol{\alpha}$), as the scaling process is essential for identifying a low-loss path, as discussed in Sec. 4.2.

Analysis of the FLOPs and the learnable parameters. In Table 5, we compare the FLOPs, trainable parameters, and stored features across different continual learning methods. Since both InfLoRA (Liang & Li, 2024) and S-LoRA bypass the prompt selection process and can directly add the learned LoRA weights to the original model weights during testing, they achieve the highest inference efficiency. Additionally, because our method can further reduce the LoRA parameters in subsequent learning stages while not requiring the storage of sample features, ES-LoRA achieves excellent performance when considering both trainable parameters and stored features.

7 Conclusion

In this paper, we revisited existing methods for continual learning (CL) that leverage foundation models and proposed three ideal properties: *rehearsal-free*, *end-to-end optimization* and enhanced *inference efficiency*. To address the challenges associated with these properties, we introduce the Scalable Low-Rank Adaptation (S-LoRA) algorithm, which incrementally decouples the learning of the direction and magnitude of LoRA parameters. Our theoretical and empirical analyses demonstrate that S-LoRA typically follows a low-loss trajectory that converges to a shared low-loss region, resulting in an excellent balance between stability and plasticity in CL. Furthermore, based on our findings, we develop variants of S-LoRA that offer even greater scalability. Extensive experiments across multiple CL benchmarks and backbones consistently validate the effectiveness of S-LoRA.

REFERENCES

Matteo Boschini, Lorenzo Bonicelli, Angelo Porrello, Giovanni Bellitto, Matteo Pennisi, Simone Palazzo, Concetto Spampinato, and Simone Calderara. Transfer without forgetting. In *European Conference on Computer Vision*, pp. 692–709. Springer, 2022.

Mathilde Caron, Hugo Touvron, Ishan Misra, Hervé Jégou, Julien Mairal, Piotr Bojanowski, and Armand Joulin. Emerging properties in self-supervised vision transformers. In *Proceedings of the IEEE/CVF international conference on computer vision*, pp. 9650–9660, 2021.

Arslan Chaudhry, Marcus Rohrbach, Mohamed Elhoseiny, Thalaiyasingam Ajanthan, Puneet K Dokania, Philip HS Torr, and Marc'Aurelio Ranzato. On tiny episodic memories in continual learning. *arXiv preprint arXiv:1902.10486*, 2019.

Rajas Chitale, Ankit Vaidya, Aditya Kane, and Archana Ghotkar. Task arithmetic with lora for continual learning. *arXiv preprint arXiv:2311.02428*, 2023.

Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pp. 248–255. Ieee, 2009.

Thang Doan, Seyed Iman Mirzadeh, and Mehrdad Farajtabar. Continual learning beyond a single model. In *Conference on Lifelong Learning Agents*, pp. 961–991. PMLR, 2023.

Alexey Dosovitskiy. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*, 2020.

Sayna Ebrahimi, Franziska Meier, Roberto Calandra, Trevor Darrell, and Marcus Rohrbach. Adversarial continual learning. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XI 16*, pp. 386–402. Springer, 2020.

Rahim Entezari, Hanie Sedghi, Olga Saukh, and Behnam Neyshabur. The role of permutation invariance in linear mode connectivity of neural networks. In *International Conference on Learning Representations*, 2022.

Robert M French. Catastrophic forgetting in connectionist networks. *Trends in cognitive sciences*, 3 (4):128–135, 1999.

Qiankun Gao, Chen Zhao, Yifan Sun, Teng Xi, Gang Zhang, Bernard Ghanem, and Jian Zhang. A unified continual learning framework with general parameter-efficient tuning. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 11483–11493, 2023.

Almog Gueta, Elad Venezian, Colin Raffel, Noam Slonim, Yoav Katz, and Leshem Choshen. Knowledge is a region in weight space for fine-tuned language models. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pp. 1350–1370, 2023.

Dan Hendrycks, Steven Basart, Mantas Mazeika, Andy Zou, Christina Kwon, and Jacob Steinhardt. Natural adversarial examples. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 15262–15271, 2021.

Neil Houlsby, Andrei Giurgiu, Stanislaw Jastrzebski, Bruna Morrone, Quentin De Laroussilhe, Andrea Gesmundo, Mona Attariyan, and Sylvain Gelly. Parameter-efficient transfer learning for nlp. In *International conference on machine learning*, pp. 2790–2799. PMLR, 2019.

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

Edward J Hu, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, Weizhu Chen, et al. Lora: Low-rank adaptation of large language models. In *International Conference on Learning Representations*, 2022.

Wei-Cheng Huang, Chun-Fu Chen, and Hsiang Hsu. Ovor: Oneprompt with virtual outlier regularization for rehearsal-free class-incremental learning. 2024.

Gabriel Ilharco, Marco Tulio Ribeiro, Mitchell Wortsman, Ludwig Schmidt, Hannaneh Hajishirzi, and Ali Farhadi. Editing models with task arithmetic. In *The Eleventh International Conference on Learning Representations*, 2023.

Menglin Jia, Luming Tang, Bor-Chun Chen, Claire Cardie, Serge Belongie, Bharath Hariharan, and Ser-Nam Lim. Visual prompt tuning. In *European Conference on Computer Vision*, pp. 709–727. Springer, 2022.

Liwei Jiang, Yudong Chen, and Lijun Ding. Algorithmic regularization in model-free overparametrized asymmetric matrix factorization. *SIAM Journal on Mathematics of Data Science*, 5 (3):723–744, 2023.

Diederik P Kingma. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

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.

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

Kibok Lee, Kimin Lee, Jinwoo Shin, and Honglak Lee. Overcoming catastrophic forgetting with unlabeled data in the wild. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 312–321, 2019.

Xiang Lisa Li and Percy Liang. Prefix-tuning: Optimizing continuous prompts for generation. *arXiv preprint arXiv:2101.00190*, 2021.

Zhizhong Li and Derek Hoiem. Learning without forgetting. *IEEE transactions on pattern analysis and machine intelligence*, 40(12):2935–2947, 2017.

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*, pp. 23638–23647, 2024.

Shih-yang Liu, Chien-Yi Wang, Hongxu Yin, Pavlo Molchanov, Yu-Chiang Frank Wang, Kwang-Ting Cheng, and Min-Hung Chen. Dora: Weight-decomposed low-rank adaptation. In *Forty-first International Conference on Machine Learning*, 2024.

Arun Mallya, Dillon Davis, and Svetlana Lazebnik. Piggyback: Adapting a single network to multiple tasks by learning to mask weights. In *Proceedings of the European conference on computer vision (ECCV)*, pp. 67–82, 2018.

James L McClelland, Bruce L McNaughton, and Randall C O'Reilly. Why there are complementary learning systems in the hippocampus and neocortex: insights from the successes and failures of connectionist models of learning and memory. *Psychological review*, 102(3):419, 1995.

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

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*, pp. 1406–1415, 2019.

Guanghui Qin and Jason Eisner. Learning how to ask: Querying lms with mixtures of soft prompts. *arXiv preprint arXiv:2104.06599*, 2021.

Zeju Qiu, Weiyang Liu, Haiwen Feng, Yuxuan Xue, Yao Feng, Zhen Liu, Dan Zhang, Adrian Weller, and Bernhard Schölkopf. Controlling text-to-image diffusion by orthogonal finetuning. *Advances in Neural Information Processing Systems*, 36:79320–79362, 2023.

Rahul Ramesh and Pratik Chaudhari. Model zoo: A growing" brain" that learns continually. *arXiv preprint arXiv:2106.03027*, 2021.

Matthew Riemer, Ignacio Cases, Robert Ajemian, Miao Liu, Irina Rish, Yuhai Tu, and Gerald Tesauro. Learning to learn without forgetting by maximizing transfer and minimizing interference. *arXiv preprint arXiv:1810.11910*, 2018.

David Rolnick, Arun Ahuja, Jonathan Schwarz, Timothy Lillicrap, and Gregory Wayne. Experience replay for continual learning. *Advances in neural information processing systems*, 32, 2019.

James Smith, Yen-Chang Hsu, Jonathan Balloch, Yilin Shen, Hongxia Jin, and Zsolt Kira. Always be dreaming: A new approach for data-free class-incremental learning. In *Proceedings of the IEEE/CVF international conference on computer vision*, pp. 9374–9384, 2021.

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*, pp. 11909–11919, 2023a.

James Seale Smith, Junjiao Tian, Shaunak Halbe, Yen-Chang Hsu, and Zsolt Kira. A closer look at rehearsal-free continual learning. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 2410–2420, 2023b.

Rishabh Tiwari, Krishnateja Killamsetty, Rishabh Iyer, and Pradeep Shenoy. Gcr: Gradient coreset based replay buffer selection for continual learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 99–108, 2022.

Catherine Wah, Steve Branson, Peter Welinder, Pietro Perona, and Serge Belongie. The caltech-ucsd birds-200-2011 dataset. Technical Report CNS-TR-2011-001, California Institute of Technology, 2011a.

Catherine Wah, Steve Branson, Peter Welinder, Pietro Perona, and Serge Belongie. The caltech-ucsd birds-200-2011 dataset. Technical report, California Institute of Technology, 2011b. URL `https://authors.library.caltech.edu/27452/`.

Liyuan Wang, Jingyi Xie, Xingxing Zhang, Mingyi Huang, Hang Su, and Jun Zhu. Hierarchical decomposition of prompt-based continual learning: Rethinking obscured sub-optimality. *Advances in Neural Information Processing Systems*, 36, 2024a.

Liyuan Wang, Xingxing Zhang, Hang Su, and Jun Zhu. A comprehensive survey of continual learning: theory, method and application. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2024b.

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*, pp. 631–648. Springer, 2022a.

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*, pp. 139–149, 2022b.

Da-Wei Zhou, Qi-Wei Wang, Zhi-Hong Qi, Han-Jia Ye, De-Chuan Zhan, and Ziwei Liu. Class-incremental learning: A survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2024.

A Appendix

A.1 Proof of Theorem 1

In this section, we will prove Theorem 1, which is a specific instance of Theorem 2. This derivation relies on applying Theorem 2 for each $r = 1, 2, ...$ in conjunction with Assumption A. Therefore, to establish the validity of Theorem 1, it is sufficient to prove Theorem 2. For clarity, we will directly prove Theorem 3, the detailed version of Theorem 2.

Assumption 1. The optimal $\Delta\mathbf{W}_i^*, i \in \{1, 2, 3, ..., N\}$ are near in the loss region, and there exists a small number $\epsilon_1 > 0, s.t., \|\mathbf{AB} - \Delta\mathbf{W}^*\| - \|\mathbf{AB} - \Delta\mathbf{W}_i^*\| < \epsilon_1$.

Assumption 2. The first $k + 1$ singular values of $\Delta\mathbf{W}^*$ are distinct, i.e., $\sigma_1 > \cdots > \sigma_k > \sigma_{k+1}$.

Theorem 2. *Suppose the k-th and $(k+1)$-th singular values are distinct, i.e., $\sigma_k > \sigma_{k+1}$. Let $\underline{\delta}$ be any number in $(0, 1)$ such that $\underline{\delta} \leq \delta := \frac{\sigma_k - \sigma_{k+1}}{\sigma_k}$. Fix any tolerance $\epsilon_2 \leq \frac{1}{m+n+r}$. The following hods with high probability for some numerical constants c, c′. Consider gradient descent with stepsize $\eta \leq c \min\{\underline{\delta}, 1 - \underline{\delta}\} \frac{\sigma_k^2}{\sigma_1^3}$ and initialization size $\rho \leq (\frac{c\underline{\delta}\epsilon_2}{\kappa_k})^{\frac{1}{c\delta}}$. Let $T = \lfloor \frac{\log(\rho^{\frac{\delta}{2(2-\sigma)}}/\rho)}{\log(1+(1-\underline{\delta})\eta\sigma_k)} \rfloor$, which is $O(\frac{1}{\underline{\delta}\eta\sigma_k} \log(\frac{\kappa_k}{\underline{\delta}_{\epsilon_2}}))$ for $\rho = (\frac{c\underline{\delta}\epsilon_2}{\kappa_k})^{\frac{1}{c\delta}}$. Then, for all t such that $(1 - c'\underline{\delta})T \leq t \leq T$, we have*

$$\|\mathbf{A}_t\mathbf{B}_t - \Delta\mathbf{W}_{(k)}^*\| \leq \epsilon_2\sigma_1 + \epsilon_1 \tag{A.3}$$

Theorem 3 (The Detailed Version of Theorem 2)**.** *Fix any $k \leq r$. Suppose $\sigma_{k+1} < \sigma_k$, choose any $\gamma \in (0, 1)$ such that $\frac{\sigma_{k+1}}{\sigma_k} \leq \gamma$. Pick any stepsize $\eta \leq \min\{\frac{\gamma\sigma_k^2}{600\sigma_1^3}, \frac{(1-\gamma)\sigma_k}{20\sigma_1^2}\}$. For any $c_\rho < 1$, let the initialization size ρ satisfy*

$$\rho \leq \min\{\frac{1}{3}, \frac{1-\gamma}{24}, \frac{c_\rho\sigma_1}{12(m+n+r)\sqrt{\frac{1-\gamma}{24}\sqrt{\sqrt{\sigma_k}}}}\},$$

and

$$\rho \leq \min\{(\frac{(1-\gamma)c_\rho\sigma_k}{1200(m+n+r)k\sigma_1})^{\frac{2(1+\gamma)}{1-\gamma}}, (\frac{\gamma\sigma_k^2}{16000r\sigma_1^2})^{\frac{1+\gamma}{1-\gamma}}, \frac{\gamma\sigma_k\sqrt{2k}}{16\sigma_1\sqrt{m+n+r}}\}$$

Define

$$T_1 = \lfloor \frac{\log(\frac{12(m+n+r)\sqrt{\frac{1-\gamma}{24}}\sqrt{\sigma_k}}{c_\rho\rho\sqrt{\sigma_1}})}{\log(1 + \frac{1+\gamma}{2}\eta\sigma_k)} \rfloor + 1, \qquad T_2 = \lfloor \frac{\log(\sqrt{\frac{24}{1-\gamma}})}{\log(1 + 0.1\eta\sigma_k)} \rfloor + 1,$$
$$T_3 = \lfloor \frac{\log(\rho^{\frac{1-\gamma}{2(1+\gamma)}}/3)}{\log(1 - \frac{3}{2}\eta\sigma_r)} \rfloor + 1, \qquad T = \lfloor \frac{\log(\rho^{\frac{1-\gamma}{2(1+\gamma)}}/\rho)}{\log(1 + \gamma\eta\sigma_k)} \rfloor \tag{A.4}$$

Define $T_0 := T_1 + T_2 + T_3$, then we have

$$\frac{T_0}{T} \leq 1 - \frac{(3-2\gamma)(1-\gamma)}{6(3\gamma+1)}. \tag{A.5}$$

Furthermore, there exists a universal constant C such that with probability at leat $1 - (Cc_\rho)^{r-k+1} - C\exp(-r/C)$, for all $T_0 \leq t \leq T$, we have

$$\|\mathbf{A}_t\mathbf{B}_t - \Delta\mathbf{W}_{(k)}^*\| \leq 8\rho^{\frac{\delta}{2(2-\sigma)}}\sigma_1 + 4\rho^{\frac{\delta}{2(2-\delta)}}\sqrt{2k}\sigma_1 \tag{A.6}$$

Proof. We first start by using the singular value decomposition (SVD) on $\Delta\mathbf{W}^*$, i.e., $\Delta\mathbf{W}^* = \mathbf{\Phi}\mathbf{\Sigma}_{\Delta\mathbf{W}^*}\mathbf{\Psi}^T$, where $\mathbf{\Phi} \in \mathbb{R}^{m\times m}$, $\mathbf{\Sigma}_{\Delta\mathbf{W}^*} \in \mathbb{R}^{m\times n}$ and $\mathbf{\Psi} \in \mathbb{R}^{m\times n}$. By replacing $\mathbf{A}, \mathbf{B}$ with $\mathbf{\Phi}^T\mathbf{A}$, $\mathbf{\Psi}^T\mathbf{B}$, we can assume that without loss of generality that $\Delta\mathbf{W}^*$ is diagonal. The distribution of the initial iterate $(\mathbf{A}_0, \mathbf{B}_0)$ remains the same due to the rotational invariance of Gaussian. Therefore, the gradient descent update becomes,

$$\mathbf{A}_+ = \mathbf{A} + \eta(\mathbf{\Sigma}_{\Delta\mathbf{W}^*} - \mathbf{AB})\mathbf{B}^T, \qquad \mathbf{B}_+ = \mathbf{B} + \eta(\mathbf{\Sigma}_{\Delta\mathbf{W}^*} - \mathbf{AB})^T\mathbf{A}, \tag{A.7}$$

where the subscript $+$ indicates the next iteration. For simplicity, let $\mathbf{U}$ be the upper $k \times r$ sub-matrix of $\mathbf{A}$ and $\mathbf{J}$ be the lower $(m-k) \times r$ sub-matrix of $\mathbf{A}$. Similarly, $\mathbf{V}$ and $\mathbf{K}$ are the upper $k \times r$ sub-matrix of $\mathbf{B}$ and the lower $(n-k) \times r$ sub-matrix of $\mathbf{B}$. Let $\Sigma = diag(\sigma_1, ..., \sigma_k)$ be the upper left $k \times k$ sub-matrix and $\tilde{\Sigma} = diag(\sigma_{k+1}, ..., \sigma_{\min\{m,n\}})$. Then the gradient descent of Eqn. (A.7) can be expressed as,

$$\begin{cases} \mathbf{U}^{+} = \mathbf{U} + \eta\mathbf{\Sigma}\mathbf{V} - \eta\mathbf{U}(\mathbf{V}^T\mathbf{V} + \mathbf{K}^T\mathbf{K}) \\ \mathbf{V}^{+} = \mathbf{V} + \eta\mathbf{\Sigma}\mathbf{U} - \eta\mathbf{V}(\mathbf{U}^T\mathbf{U} + \mathbf{J}^T\mathbf{J}) \\ \mathbf{J}^{+} = \mathbf{J} + \eta\tilde{\mathbf{\Sigma}}\mathbf{K} - \eta\mathbf{J}(\mathbf{V}^T\mathbf{V} + \mathbf{K}^T\mathbf{K}) \\ \mathbf{K}^{+} = \mathbf{K} + \eta\tilde{\mathbf{\Sigma}}^T\mathbf{J} - \eta\mathbf{K}(\mathbf{U}^T\mathbf{U} + \mathbf{J}^T\mathbf{J}) \end{cases} \tag{A.8}$$

To account for the potential imbalance of $\mathbf{U}$ and $\mathbf{V}$, we introduce the following quantities,

$$\mathbf{F} = \frac{\mathbf{U}+\mathbf{V}}{2}, \quad \mathbf{G} = \frac{\mathbf{U}-\mathbf{V}}{2}, \quad \mathbf{P} = \mathbf{\Sigma} - \mathbf{F}\mathbf{F}^T + \mathbf{G}\mathbf{G}^T, \quad \mathbf{Q} = \mathbf{F}\mathbf{G}^T - \mathbf{G}\mathbf{F}^T.$$

Here $\mathbf{F}$ is the symmetrized part of the matrix $\mathbf{U}, \mathbf{V}$ while $\mathbf{G}$ denote the imbalance part between them. The part $\mathbf{P} + \mathbf{Q} = \mathbf{\Sigma} - \mathbf{U}\mathbf{V}^T$ captures the approximation error. Based on these definitions and the Eqn. (A.8), we can further derive that,

$$\begin{aligned} \mathbf{F}_{+} &= \mathbf{F} + \eta\mathbf{P}\mathbf{F} - \eta(\mathbf{F}\mathbf{G}^T - \mathbf{G}\mathbf{F}^T)\mathbf{G} - \eta\mathbf{F}\frac{\mathbf{K}^T\mathbf{K} + \mathbf{J}^T\mathbf{J}}{2} - \eta\mathbf{G}\frac{\mathbf{K}^T\mathbf{K} - \mathbf{J}^T\mathbf{J}}{2} \\ \mathbf{G}_{+} &= \mathbf{G} - \eta\mathbf{P}\mathbf{G} + \eta(\mathbf{F}\mathbf{G}^T - \mathbf{G}\mathbf{F}^T)\mathbf{F} - \eta\mathbf{F}\frac{\mathbf{K}^T\mathbf{K} - \mathbf{J}^T\mathbf{J}}{2} - \eta\mathbf{G}\frac{\mathbf{K}^T\mathbf{K} + \mathbf{J}^T\mathbf{J}}{2} \end{aligned} \tag{A.9}$$

and

$$\begin{aligned} \mathbf{P}_{+} = \mathbf{P} &- \eta\mathbf{P}(\mathbf{\Sigma} - \mathbf{P}) - \eta(\mathbf{\Sigma} - \mathbf{P})\mathbf{P} + \eta^2(\mathbf{P}\mathbf{P}\mathbf{P} - \mathbf{P}\mathbf{\Sigma}\mathbf{P}) - 2\eta\mathbf{G}\mathbf{G}^T\mathbf{P} - 2\eta\mathbf{P}\mathbf{B}\mathbf{B}^T \\ &- \eta(\mathbf{F} + \eta\mathbf{P}\mathbf{F})C^T - \eta C(\mathbf{F} + \eta\mathbf{P}\mathbf{A})^T - \eta^2 CC^T \\ &+ \eta(\mathbf{G} + \eta\mathbf{P}\mathbf{G})D^T + \eta\mathbf{D}(\mathbf{B} + \eta\mathbf{P}\mathbf{B})^T + \eta^2\mathbf{D}\mathbf{D}^T \end{aligned} \tag{A.10}$$

where

$$\begin{aligned} C &= -\mathbf{F}\mathbf{G}^T\mathbf{G} + \mathbf{G}\mathbf{F}^T\mathbf{G} - \mathbf{F}\frac{\mathbf{K}^T\mathbf{K} + \mathbf{J}^T\mathbf{J}}{2} - \mathbf{G}\frac{\mathbf{K}^T\mathbf{K} - \mathbf{J}^T\mathbf{J}}{2}, \\ D &= \mathbf{F}\mathbf{G}^T\mathbf{F} - \mathbf{G}\mathbf{F}^T\mathbf{F} - \mathbf{F}\frac{\mathbf{K}^T\mathbf{K} - \mathbf{J}^T\mathbf{J}}{2} - \mathbf{G}\frac{\mathbf{K}^T\mathbf{K} + \mathbf{J}^T\mathbf{J}}{2}. \end{aligned}$$

Note that

$$\mathbf{A}_t\mathbf{B}_t - \Delta\mathbf{W}^*_{(r)} = \begin{pmatrix} \mathbf{U}_t \\ \mathbf{J}_t \end{pmatrix} \begin{pmatrix} \mathbf{V}_t^T & \mathbf{K}_t^T \end{pmatrix} - \begin{pmatrix} \mathbf{\Sigma} & 0 \\ 0 & 0 \end{pmatrix} = \begin{pmatrix} \mathbf{U}_t\mathbf{V}_t^T - \mathbf{\Sigma} & \mathbf{U}_t\mathbf{K}_t^T \\ \mathbf{J}_t\mathbf{V}_t^T & \mathbf{J}_t\mathbf{K}_t^T \end{pmatrix}$$

According to the Proposition B.2 and B.5 of Jiang et al. (2023), it holds with high probability that for any $T_1 + T_2 + T_3 \le t \le T$,

$$\|\mathbf{U}_t\mathbf{K}_t^T\| \le 3\rho^{\frac{1-\gamma}{2(1+\gamma)}}\sigma_1, \quad \|\mathbf{J}_t\mathbf{V}_t^T\| \le 3\rho^{\frac{1-\gamma}{2(1+\gamma)}}\sigma_1, \quad \|\mathbf{J}_t\mathbf{K}_t^T\| \le \rho^{\frac{1-\gamma}{(1+\gamma)}}\sigma_1 \tag{A.11}$$

and

$$\|\mathbf{U}_t\mathbf{V}_t^T - \mathbf{\Sigma}\| = \|\mathbf{P}_t + \mathbf{Q}_t\| \le \|\mathbf{P}_t\| + \|\mathbf{Q}_t\| \le \rho^{\frac{1-\gamma}{2(1+\gamma)}}\sigma_1 + 4\rho^{\frac{1-\gamma}{2(1+\gamma)}}\sqrt{2r}\sigma_1. \tag{A.12}$$

□

By combining these parts, we can have

$$\|\mathbf{A}_t\mathbf{B}_t - \Delta\mathbf{W}^*_{(k)}\| \le 8\rho^{\frac{1-\gamma}{2(1+\gamma)}}\sigma_1 + 4\rho^{\frac{1-\gamma}{2(1+\gamma)}}\sqrt{2k}\sigma_1 = (8\rho^{\frac{1-\gamma}{2(1+\gamma)}} + 4\rho^{\frac{1-\gamma}{2(1+\gamma)}}\sqrt{2k})\sigma_1 \tag{A.13}$$

A.2 Additional Results of Sec. 4.2

To demonstrate the general trend of descent in the learned $\boldsymbol{\alpha}$ values, we plot these values for a much longer task length setting on ImageNet-R ($N = 20$). Additionally, we present the $\boldsymbol{\alpha}$ values on another dataset, DomainNet ($N = 5$), for further validation, as shown in Fig. 6. The results demonstrate that even during long task sequences or across various datasets, S-LoRA consistently exhibits a decreasing trend in the learned $\boldsymbol{\alpha}$ values. This aligns with the theoretical insights discussed in Sec. 5.

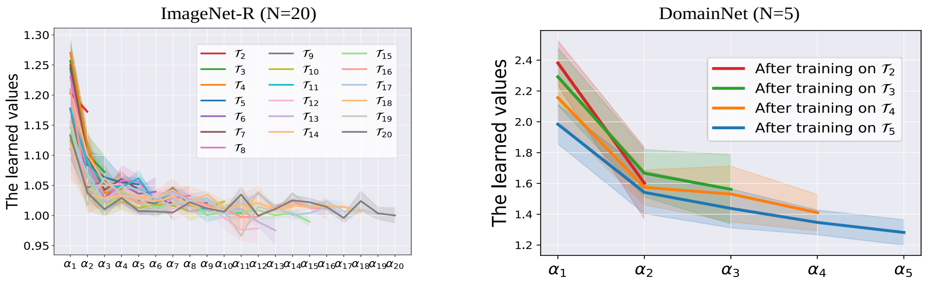
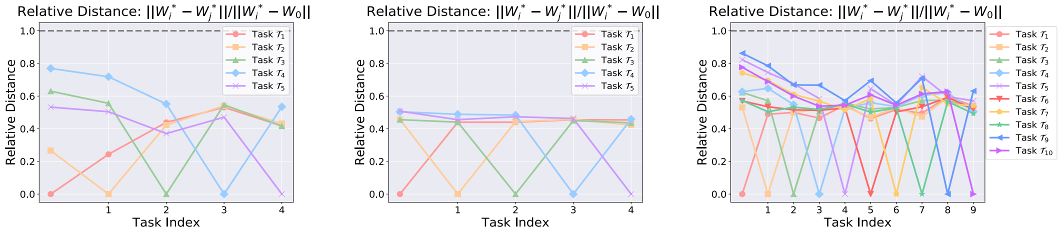


Figure 6: (a) illustrates the gradually learned $\boldsymbol{\alpha}$ values on ImageNet-R with 20 tasks, and (b) exhibits the $\boldsymbol{\alpha}$ values on DomainNet.



(a) IN-R(N=5) with DINO (b) DomainNet (N=5) (c) IN-R(N=10)

Figure 7: Other relative distance in Finding 1. (a)(c) shows the results on DomainNet and ImageNet(N=10), respectively; (b) exhibits the results on ImageNet(N=10) with DINO.

A.3 Results on other CL Benchmarks

In addition to ImageNet-R, ImageNet-A, and DomainNet, we conduct experiments on the CIFAR-100 (Krizhevsky et al., 2009) and CUB-200 (Wah et al., 2011b) datasets. CIFAR-100, a widely-used benchmark for image classification, contains 60,000 images distributed across 100 classes, with each class having 600 images. Specifically, we split CIFAR-100 into 10 tasks, each consisting of 10 classes. CUB-200, a fine-grained dataset focused on bird species, includes 11,788 images across 200 classes. We divide it into 10 tasks, with each task containing 20 species. As shown in Table 6, our proposed S-LoRA and ES-LoRA consistently achieve outstanding performance on both datasets.

In addition to Fig. 2(a), we also present additional results on relative distances in Fig. 7, including results on different benchmarks in (a), different backbones in (b), and varying task lengths in (c). Consistent with Finding 1, the same trend is observed across different benchmarks, backbones, and task lengths.

Table 6: Performance comparison on benchmark datasets.

Method	CIFAR100		CUB200	
	Acc↑	AAA ↑	Acc↑	AAA ↑
Fine-Tuning	$69.49_{(0.50)}$	$80.35_{(0.87)}$	$51.43_{(1.41)}$	$69.74_{(0.93)}$
L2P (Wang et al., 2022b)	$83.18_{(1.20)}$	$87.69_{(1.05)}$	$65.18_{(2.49)}$	$76.12_{(1.27)}$
Dual-Prompt (Wang et al., 2022a)	$81.48_{(0.86)}$	$86.41_{(0.66)}$	$68.00_{(1.06)}$	$79.40_{(0.88)}$
Coda-Prompt (Smith et al., 2023a)	$86.31_{(0.12)}$	$90.67_{(0.22)}$	$71.92_{(0.33)}$	$78.76_{(0.65)}$
InfLoRA (Liang & Li, 2024)	$86.75_{(0.35)}$	$91.72_{(0.15)}$	$70.82_{(0.23)}$	$81.39_{(0.14)}$
S-LoRA	$\mathbf{88.01}_{(0.31)}$	$\mathbf{92.54}_{(0.18)}$	$\mathbf{77.48}_{(0.20)}$	$\mathbf{85.59}_{(0.44)}$
ES-LoRA1	$87.26_{(0.22)}$	$92.05_{(0.31)}$	$76.35_{(0.28)}$	$83.89_{(0.35)}$
ES-LoRA2	$87.09_{(0.45)}$	$92.01_{(0.33)}$	$75.95_{(0.55)}$	$83.21_{(0.31)}$