REDUCING PRETRAINED REPRESENTATION DEGRADATION IN PARAMETER-EFFICIENT CONTINUAL LEARNING

Anonymous authors
Paper under double-blind review

ABSTRACT

Continual learning (CL) aims to enable models to acquire new knowledge while retaining previously learned information, yet suffers from catastrophic forgetting. With the rise of pretrained models (PTMs), recent continual learning approaches increasingly rely on parameter-efficient fine-tuning methods such as Low-Rank Adaptation (LoRA). However, existing approaches primarily focus on learning new tasks and mitigating inter-task interference, while paying limited attention to preserving the pretrained representations that encode previously acquired knowledge. In this work, we show that sequential low-rank updates can progressively alter these pretrained representations, resulting in degraded performance on previously learned tasks and unstable adaptation. Motivated by this observation, we propose Anchored Orthogonal Low-Rank Adaptation (AO-LoRA), which anchors task-specific orthogonal transformations to the pretrained model and accumulates them in a common reference space. This design preserves a consistent reference for adaptation across tasks while retaining the parameter efficiency of low-rank fine-tuning. Extensive experiments on four vision benchmarks demonstrate that AO-LoRA consistently achieves superior or competitive performance compared to existing LoRA-based CL methods, while requiring fewer trainable parameters. Furthermore, AO-LoRA exhibits improved training stability, particularly under large learning rates, suggesting that maintaining representation structure helps mitigate instability in continual learning. Code is provided in the supplement and will be publicly released.

1 INTRODUCTION

Continual Learning (CL) is widely regarded as a key capability for future AI systems, allowing them to acquire new knowledge while retaining previously learned information (Parisi et al., 2019; De Lange et al., 2021). Ideally, a CL model should maintain stable performance across all learned tasks. However, a significant drop in performance on earlier tasks after learning new ones, known as catastrophic forgetting (McCloskey & Cohen, 1989; Ratcliff, 1990), typically arises from the instability of learned representations under continual adaptation.

Before the advent of large-scale pretrained models (PTMs), continual learning approaches based on convolutional neural networks (CNNs) mitigated forgetting through mechanisms such as structural expansion (Rusu et al., 2016), data replay (Rebuffi et al., 2017), and weight regularization (Kirkpatrick et al., 2017). With the emergence of PTMs, research on continual learning has shifted toward adapting pretrained models rather than training from scratch (Yang et al., 2025). Three representative paradigms have emerged for PTM-based continual learning: prompt-based methods (Wang et al., 2022b; Smith et al., 2023), adapter-based methods (Wang et al., 2025), and LoRA-based methods (Liang & Li, 2024; Wu et al., 2025), with LoRA gaining increasing attention for its parameter efficiency (He et al., 2025; Zheng et al., 2026). However, existing works predominantly inherit design principles developed for CNN-based models, focusing on parameter-efficient adaptation to new tasks, while largely treating pretrained models as fixed backbones augmented with low-rank updates, rather than as structured knowledge to be actively preserved and exploited. Prior work on hyperspherical learning has highlighted the importance of preserving pretrained representations (Liu et al., 2018), motivating orthogonal adaptation in OFT (Qiu et al., 2023). Similar findings in Ling

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

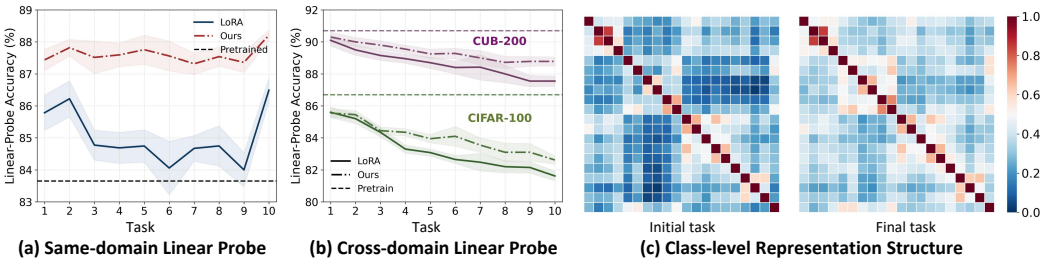


Figure 1: Effects of continual learning on the accessibility and structure of pretrained representations. (a) and (b) show held-out linear-probe accuracy under the same-domain and cross-domain settings, respectively. (c) shows the class-level representation structure among the same held-out ImageNet-R classes after the initial and final adaptation tasks, where higher off-diagonal similarity indicates reduced inter-class separation. These results show that continual learning progressively reduces the accessibility of transferable pretrained knowledge and compresses inter-class separation in the representation space. Additional visualizations are provided in Appendix D.5.

et al. (2025) show that low-rank updates can perturb pretrained parameters even under regularization, highlighting the need to better preserve pretrained representations during adaptation.

Motivation. Pretrained models encode rich and transferable knowledge acquired from large-scale pretraining, but such knowledge is difficult to quantify directly. We therefore study pretrained knowledge through two observable properties of learned representations: transferability and relational structure. We use these properties to examine whether the useful representations inherited from pretraining are preserved during LoRA-based continual learning.

Experiment 1: Held-out Linear Probe of Pretrained Knowledge. To investigate whether transferable knowledge encoded in the pretrained representation remains accessible during continual learning, we conduct linear probing on held-out data under two settings: *same-domain* and *cross-domain*.

In the ***same-domain setting***, we hold out 20 classes from ImageNet-R and never use them during continual learning. We first train a linear classifier on the frozen pretrained representations of the held-out classes to obtain a reference accuracy. We then continually learn on the remaining ImageNet-R classes, with 20 classes per task, and retrain the classifier on the same held-out classes after each task. As shown in Fig. 1 (a), LoRA-based continual learning initially improves transferability to the held-out classes, but the gain gradually diminishes with continued learning. In the ***cross-domain setting***, we evaluate the learned representations on two unseen datasets from different visual domains, CIFAR-100 and CUB-200. The model is continually trained only on ImageNet-R, while a new linear classifier is trained on each unseen dataset after every task. As shown in Fig. 1 (b), transferability generally declines as continual learning progresses, following a trend consistent with the same-domain setting. ***This leads to our first observation:*** *continual learning progressively reduces the accessibility of transferable knowledge inherited from the pretrained model.* We next investigate the changes in the inherited representation accompanying this degradation.

Experiment 2: Changes in Class-level Representation Structure. To understand how continual learning affects the representation underlying transferable knowledge, we examine how relationships among class representations evolve during learning. Let $z_i = f_\theta(x_i) \in \mathbb{R}^d$ denote the representation of sample x_i produced by a model. For each class c, we represent its class-level feature by the prototype: $\mu_c = \frac{1}{|\mathcal{D}_c|} \sum_{x_i \in \mathcal{D}_c} z_i$, where $\mathcal{D}_c$ denotes the samples belonging to class c. We characterize the class-level representation structure through pairwise relationships among class prototypes: $\mathcal{G} = \{g(\mu_i, \mu_j)\}_{i,j=1}^{C}$, where $g(\cdot, \cdot)$ measures the relationship between two prototypes. We instantiate g as cosine similarity, which captures the angular relationship between class prototypes while being invariant to their magnitudes. Specifically, we define:

$$\mathcal{G} = \left[\frac{\mu_i^\top \mu_j}{\|\mu_i\|_2 \|\mu_j\|_2} \right]_{i,j=1}^{C}. \tag{1}$$

We use the same 20 held-out classes from ImageNet-R as in Experiment 1 and track their pairwise cosine similarities throughout continual learning. Fig. 1 (c) shows the initial and final similarities,

with results for all intermediate tasks provided in Appendix D.5. Interestingly, the overall pattern of inter-class relationships remains largely preserved. However, the pairwise cosine similarities generally increase after continual learning, indicating that the angular separation between class prototypes becomes smaller. In other words, continual learning largely preserves the relative organization of classes, while progressively compressing their angular separation in the representation space. Such compression of inter-class separation suggests a degradation in the class-discriminative structure of the representation. ***This leads to our second observation:*** *the reduced transferability is accompanied by a compression of inter-class representation separation, suggesting that continual learning alters the class-discriminative structure inherited from pretraining.*

Contribution. The above observations motivate us to investigate how to preserve the discriminative structure of pretrained representations during continual learning. Since the degradation in class separability is reflected by increased inter-class cosine similarity, i.e., reduced angular separation between class representations, we seek an update mechanism that preserves this relational structure. Motivated by the angle-preserving property of orthogonal fine-tuning (Qiu et al., 2023), we propose to leverage orthogonal transformations for continual learning. However, naively composing task-specific orthogonal transformations sequentially can alter the representations relied upon by previous tasks, leading to degraded performance on earlier tasks.

Therefore, we anchor each task-specific update to the pretrained representations and accumulate them in a common space, preventing progressive representation degradation. Building on this formulation, we express the orthogonal updates in a low-rank form and impose inter-task orthogonality on the transformed update directions to reduce task interference. We call this approach Anchored Orthogonal Low-Rank Adaptation (**AO-LoRA**). Fig. 2 compares the key operations of LoRA and AO-LoRA within the attention block. Unlike the additive updates in LoRA, AO-LoRA employs Householder reflections (Householder, 2013) to construct orthogonal transformations through multiplicative updates.

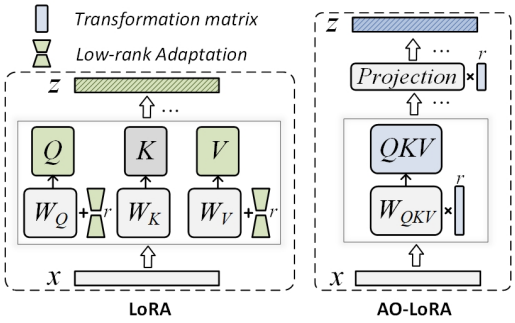


Figure 2: Update operations in LoRA and AO-LoRA. AO-LoRA applies orthogonal transformations to the QKV and output projection layers in Transformer attention blocks.

In this paper, the proposed method AO-LoRA achieves comparable or superior performance to previous LoRA-based methods on four vision CL benchmarks and one language CL benchmark, while achieving higher parameter efficiency. The main contributions of this work are as follows:

- We investigate how continual learning affects pretrained representations and show that it progressively reduces the accessibility of transferable knowledge and compresses inter-class representation separation.
- We propose Anchored Orthogonal Low-Rank Adaptation (AO-LoRA), which anchors task-specific orthogonal updates to the pretrained representations and accumulates them in a common reference space, with inter-task orthogonality imposed to reduce task interference.
- Extensive evaluations on widely adopted benchmarks show that AO-LoRA outperforms prior LoRA-based methods with higher parameter efficiency, while maintaining stable adaptation even at large learning rates, indicating improved robustness.

2 Related Work

Continual Learning (CL). Large-scale pretrained models (PTMs) (Devlin et al., 2019) have motivated parameter-efficient fine-tuning (PEFT) (Wei et al., 2024; Zhang et al., 2026) for CL. Existing PTM-based CL methods mainly adopt prompt-based (Wang et al., 2022b; Smith et al., 2023), adapter-based (Wang et al., 2025), or LoRA-based approaches (Liang & Li, 2024; Wu et al., 2025). Recent LoRA-based CL methods primarily focus on adapting to new tasks while mitigating inter-task interference (He et al., 2025; Zheng et al., 2026). However, they largely treat the pretrained

model as a backbone to be adapted, while paying limited attention to preserving the representations inherited from pretraining. *In this work, we instead consider the pretrained representation structure as an object to be explicitly preserved during continual learning.*

Parameter-efficient Adaptation. Recent parameter-efficient adaptation methods have explored different parameterizations, with LoRA (Hu et al., 2021) adopting additive low-rank updates for efficient adaptation, while Orthogonal Fine-Tuning (OFT) (Qiu et al., 2023) uses multiplicative orthogonal transformations to preserve pairwise relationships between neuron vectors by construction. Building on OFT, subsequent works have explored more expressive and efficient forms of orthogonal adaptation. ETHER (Bini et al., 2024) employs Householder reflections to enable lightweight orthogonal transformations, while HRA (Yuan et al., 2024) combines a sequence of reflections with low-rank adaptation to bridge the advantages of LoRA and OFT. However, these methods are primarily designed for single-task adaptation and do not consider the cumulative effects of applying task-specific transformations in continual learning. Naively applying these transformations sequentially can lead to representation degradation and reduced continual learning performance. *In contrast, we extend orthogonal adaptation to continual learning by anchoring each task-specific transformation to the pretrained weights, thereby avoiding cumulative changes to the reference representation.*

Orthogonality-based Continual Learning. Orthogonality has also been explored as a mechanism for reducing interference in continual learning. Orthogonal Gradient Descent (OGD) (Farajtabar et al., 2020) projects gradients onto directions orthogonal to those associated with previous tasks, thereby reducing interference. Similarly, Continual Learning in Low-Rank Orthogonal Subspaces (Chaudhry et al., 2020) learns task-specific representations in orthogonal low-rank subspaces and further encourages their gradient directions to remain orthogonal. O-LoRA (Wang et al., 2023) extends this idea to LoRA-based continual learning by constraining task-specific LoRA subspaces to be mutually orthogonal. These methods primarily leverage orthogonality to separate task-specific updates and mitigate inter-task interference. *In contrast, we focus on preserving the transferable representations inherited from the pretrained model by maintaining a common reference throughout continual adaptation.*

3 METHODOLOGY

3.1 PRELIMINARIES

Task Formulation. We consider a class-incremental setting where a pretrained model with parameters $\boldsymbol{W}_0$ is sequentially adapted to a series of tasks $\{\mathcal{T}_t\}_{t=1}^{T}$ with datasets $\{\mathcal{D}_t\}_{t=1}^{T}$. Each dataset $\mathcal{D}_t = (x_k^{(t)}, y_k^{(t)})_{k=1}^{|\mathcal{D}_t|}$ consists of input-label pairs over C classes. After training on task $\mathcal{T}_t$, the model is evaluated on all previously seen tasks $\mathcal{T}_{1:t}$ without access to past data. At inference, the model operates in a task-agnostic manner, making predictions based on the accumulated adaptations from all learned tasks.

Householder Reflection. A Householder reflection is a parametric orthogonal transformation defined by a vector $\boldsymbol{u} \in \mathbb{R}^d$:

$$\boldsymbol{H}(\boldsymbol{u}) = \boldsymbol{I} - 2\frac{\boldsymbol{u}\boldsymbol{u}^\top}{\boldsymbol{u}^\top\boldsymbol{u}}, \tag{2}$$

where $\boldsymbol{I}$ denotes the identity matrix. By construction, $\boldsymbol{H}(\boldsymbol{u})$ is orthogonal, i.e., $\boldsymbol{H}(\boldsymbol{u})^\top\boldsymbol{H}(\boldsymbol{u}) = \boldsymbol{I}$. Therefore, it preserves both Euclidean norms and pairwise cosine similarities:

$$\|\boldsymbol{H}(\boldsymbol{u})\boldsymbol{x}\|_2 = \|\boldsymbol{x}\|_2, \qquad \cos\big(\boldsymbol{H}(\boldsymbol{u})\boldsymbol{x}, \boldsymbol{H}(\boldsymbol{u})\boldsymbol{y}\big) = \cos(\boldsymbol{x}, \boldsymbol{y}). \tag{3}$$

This structure-preserving property (Qiu et al., 2023) makes orthogonal adaptation particularly attractive for our setting: it enables adaptation while preserving pairwise angular relationships, which our analysis (***Experiment 2*** in Section 1) identifies as progressively degraded during continual learning.

Geometrically, $\boldsymbol{H}(\boldsymbol{u})$ reflects an input vector $\boldsymbol{x} \in \mathbb{R}^d$ across the hyperplane orthogonal to $\boldsymbol{u}$. In parameter-efficient adaptation, a pretrained weight matrix $\boldsymbol{W}$ can be updated via an orthogonal transformation:

$$\boldsymbol{z} = \boldsymbol{W}\boldsymbol{H}(\boldsymbol{u})\boldsymbol{x}. \tag{4}$$

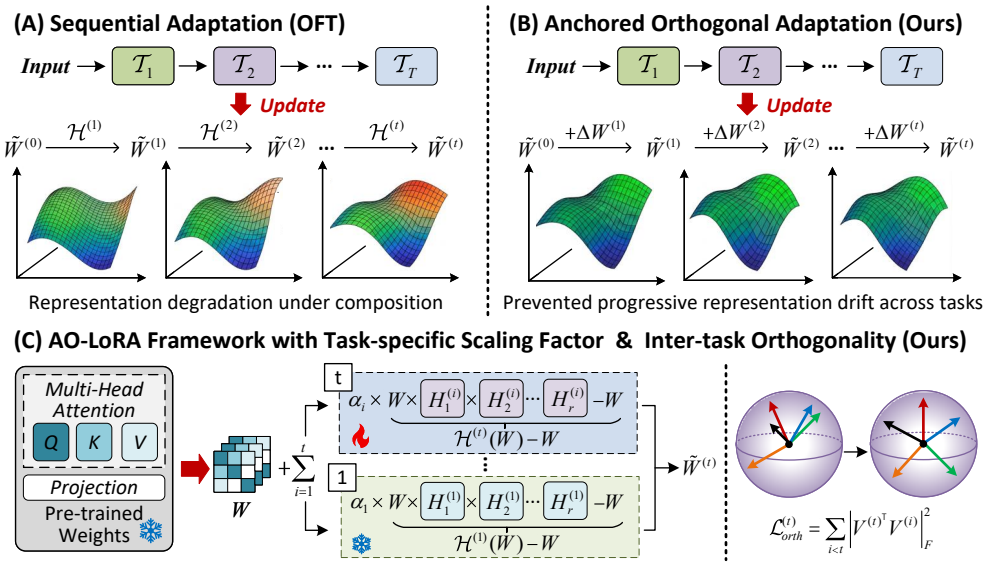


Figure 3: An overview of our proposed method. (A) Directly applying OFT can lead to representation degradation under sequential composition. (B) Our anchored residual adaptation prevents progressive drift of the reference representation across tasks. (C) AO-LoRA framework with task-specific scaling factor α_i and inter-task orthogonality $\mathcal{L}_{orth}$.

The orthogonality constraint restricts the update to a structured transformation, avoiding arbitrary changes induced by unconstrained additive updates (Dong et al., 2024). This constrained parameterization can lead to more stable optimization, as empirically demonstrated in Fig. 6, where the proposed method exhibits substantially improved robustness to large learning rates.

To further increase expressivity, multiple Householder reflections can be composed:

$$\boldsymbol{H}_r = \prod_{i=1}^{r} \boldsymbol{H}_i, \quad \boldsymbol{H}_i = \boldsymbol{I} - 2\boldsymbol{u}_i\boldsymbol{u}_i^\top, \tag{5}$$

where $\boldsymbol{u}_i \in \mathbb{S}^{d-1}$ denotes that each reflection vector is normalized to unit length. The resulting transformation $\boldsymbol{H}_r$ remains orthogonal while enabling more flexible transformations. Following Yuan et al. (2024), we adopt such a chain structure in our method.

3.2 Anchored Orthogonal Adaptation

In this section, we first examine why naive sequential orthogonal adaptation is insufficient for continual learning, and then introduce our anchored formulation to prevent representation degradation. Given a pretrained weight matrix $\boldsymbol{W}$, Yuan et al. (2024) employ Householder reflections for orthogonal adaptation by right-multiplying $\boldsymbol{W}$ with one or multiple Householder matrices:

$$\widetilde{\boldsymbol{W}} = \mathcal{H}_r(\boldsymbol{W}) = \boldsymbol{W}\boldsymbol{H}_1 \cdots \boldsymbol{H}_r, \tag{6}$$

where $\mathcal{H}_r$ denotes the composition of r Householder reflection matrices [1]. When extending orthogonal adaptation to continual learning, a natural approach is to associate each task with a task-specific orthogonal transformation and apply these transformations sequentially to the adapted weights. Formally, at task t, the updated weights are given by:

$$\widetilde{\boldsymbol{W}}^{(t)} = \mathcal{H}^{(t)}\big(\widetilde{\boldsymbol{W}}^{(t-1)}\big), \quad \widetilde{\boldsymbol{W}}^{(0)} = \boldsymbol{W}. \tag{7}$$

As illustrated in Fig. 3 (A), for an input $\boldsymbol{x}$ from an earlier task $k < t$, its representation is further transformed by subsequent task-specific updates, causing it to deviate from the representation established when task k was learned. This cumulative transformation leads to representation degradation and can degrade performance on previously learned tasks.

[1] For notational simplicity, we omit the subscript r when no ambiguity arises.

To mitigate the representation degradation caused by sequential composition, we propose an anchored formulation of orthogonal adaptation. Instead of directly replacing $\boldsymbol{W}$ with $\mathcal{H}(\boldsymbol{W})$, we construct each task-specific transformation with respect to the pretrained weights $\boldsymbol{W}$. Specifically, for task t, we construct an orthogonal transformation $\mathcal{H}^{(t)}$ parameterized by a chain of Householder reflections, and define its task-specific update as

$$\Delta \boldsymbol{W}^{(t)} = \mathcal{H}^{(t)}(\boldsymbol{W}) - \boldsymbol{W}. \tag{8}$$

As shown in Fig. 3 (B), each task-specific update is anchored to the same pretrained weights, rather than being applied recursively to the weights adapted for previous tasks. This provides a common reference for all task updates and prevents the reference representation from being progressively altered by subsequent tasks. The task-specific updates are then accumulated to incorporate knowledge from multiple tasks without compounding transformations across tasks.

However, directly accumulating the task-specific updates gives equal weight to all tasks, which may lead to imbalanced adaptation as some task updates become more dominant than others. To address this, we introduce a learnable task-specific scaling factor α_i to modulate each task's contribution (Fig. 3 (C)). The final update is defined as:

$$\widetilde{\boldsymbol{W}}^{(t)} = \boldsymbol{W} + \sum_{i=1}^{t} \alpha_i \big(\mathcal{H}^{(i)}(\boldsymbol{W}) - \boldsymbol{W}\big). \tag{9}$$

3.3 Low-Rank Formulation and Inter-task Orthogonality

The composition of r Householder reflections applied to $\boldsymbol{W}$ can be written as:

$$\mathcal{H}_r(\boldsymbol{W}) = \boldsymbol{W} \prod_{i=1}^{r} (\boldsymbol{I} - 2\boldsymbol{u}_i \boldsymbol{u}_i^\top) = \boldsymbol{W} + \boldsymbol{W}\boldsymbol{U}_r \boldsymbol{\Gamma}_r \boldsymbol{U}_r^\top, \tag{10}$$

where $\boldsymbol{U}_r = [\boldsymbol{u}_1, \boldsymbol{u}_2, \ldots, \boldsymbol{u}_r] \in \mathbb{R}^{d\times r}$, and $\boldsymbol{\Gamma}_r \in \mathbb{R}^{r\times r}$ is an upper triangular matrix defined recursively. Specifically, let $\boldsymbol{\Gamma}_1 = [-2]$. For $r \geq 2$, $\boldsymbol{\Gamma}_r$ is constructed as:

$$\boldsymbol{\Gamma}_r = \begin{bmatrix} \boldsymbol{\Gamma}_{r-1} & -2\,\boldsymbol{\Gamma}_{r-1}\boldsymbol{U}_{r-1}^\top \boldsymbol{u}_r \\ \boldsymbol{0}^\top & -2 \end{bmatrix}, \tag{11}$$

Substituting Eq. 10 into the update rule Eq. 9 yields:

$$\widetilde{\boldsymbol{W}}^{(t)} = \boldsymbol{W} + \sum_{i=1}^{t} \alpha_i \boldsymbol{W}\boldsymbol{U}_r^{(i)} \boldsymbol{\Gamma}_r^{(i)} {\boldsymbol{U}_r^{(i)}}^\top. \tag{12}$$

By defining $\boldsymbol{B} = \boldsymbol{U}_r^\top$ and $\boldsymbol{A} = \alpha \boldsymbol{W}\boldsymbol{U}_r\boldsymbol{\Gamma}_r$, we obtain $\widetilde{\boldsymbol{W}} = \boldsymbol{W} + \boldsymbol{A}\boldsymbol{B}$. Here, $\boldsymbol{B}$ is directly parameterized by the matrix of Householder vectors $\boldsymbol{U}_r$, while $\boldsymbol{A}$ is jointly determined by $\boldsymbol{W}$ and $\boldsymbol{U}_r$. Thus, the update takes the same low-rank form as LoRA, while its low-rank factors are constrained by the orthogonal transformation, providing a structured alternative to unconstrained low-rank updates.

To reduce inter-task interference, we encourage the update directions of different tasks to be mutually orthogonal after transformation by the pretrained weights. For task t, given the pretrained weight $\boldsymbol{W}$ and $\boldsymbol{U}_r^{(t)} = [\boldsymbol{u}_1^{(t)}, \ldots, \boldsymbol{u}_r^{(t)}]$, we define the corresponding task-specific directions as

$$\boldsymbol{V}_r^{(t)} = \boldsymbol{W}\boldsymbol{U}_r^{(t)} \in \mathbb{R}^{d_{\text{out}}\times r}. \tag{13}$$

We minimize the overlap between the directions of the current task and those of all previous tasks:

$$\mathcal{L}_{\text{orth}}^{(t)} = \sum_{i<t} \left\| {\boldsymbol{V}^{(t)}}^\top \boldsymbol{V}^{(i)} \right\|_F^2 = \sum_{i<t} \left\| {\boldsymbol{U}^{(t)}}^\top \boldsymbol{W}^\top \boldsymbol{W} \boldsymbol{U}^{(i)} \right\|_F^2. \tag{14}$$

This encourages the update directions of the current task to remain orthogonal to those learned for previous tasks, thereby reducing their overlap and mitigating inter-task interference. The overall objective at task t is defined as:

$$\mathcal{L}^{(t)} = \mathcal{L}_{ce}^{(t)} + \lambda \mathcal{L}_{orth}^{(t)}, \tag{15}$$

where $\mathcal{L}_{ce}^{(t)}$ denotes the cross-entropy loss for the current task, and λ is a regularization coefficient. In practice, we apply AO-LoRA to the QKV and output projections in Transformer attention blocks. Each linear layer is replaced with an AO-LoRA-enhanced module. During training, the pretrained weights remain frozen, while only the Householder vectors and scaling factors are optimized. See Algorithm 1 in the Appendix for the detailed learning procedure.

Table 1: Comparison on ImageNet-R with 5, 10, and 20 tasks. Params (%) denotes the proportion of trainable parameters relative to the pretrained backbone. '†' indicates reproduced results.

Tasks	**Params**	**ImageNet-R (N=5)**		**ImageNet-R (N=10)**		**ImageNet-R (N=20)**	
Methods	%	$\overline{A}_5$ (↑)	Avg. (↑)	$\overline{A}_{10}$ (↑)	Avg. (↑)	$\overline{A}_{20}$ (↑)	Avg. (↑)
Joint Train	-	$81.69_{(0.30)}$	$85.57_{(0.13)}$	$81.69_{(0.28)}$	$86.25_{(0.09)}$	$81.66_{(0.22)}$	$86.41_{(0.10)}$
Finetune	-	$69.26_{(0.74)}$	$78.88_{(0.31)}$	$60.42_{(1.64)}$	$73.18_{(0.32)}$	$47.06_{(2.05)}$	$63.01_{(0.56)}$
L2P (Wang et al., 2022b)	0.2	$73.04_{(0.71)}$	$76.94_{(0.41)}$	$71.26_{(0.44)}$	$76.13_{(0.46)}$	$68.97_{(0.51)}$	$74.16_{(0.32)}$
DualPrompt (Wang et al., 2022a)	0.5	$69.99_{(0.57)}$	$72.24_{(0.41)}$	$68.22_{(0.20)}$	$73.81_{(0.39)}$	$65.23_{(0.45)}$	$71.30_{(0.16)}$
CODA-Prompt (Smith et al., 2023)	4.6	$76.63_{(0.27)}$	$80.30_{(0.28)}$	$74.05_{(0.41)}$	$78.14_{(0.39)}$	$69.38_{(0.33)}$	$73.95_{(0.63)}$
InfLoRA (Liang & Li, 2024)	0.3	$77.52_{(0.37)}$	$82.01_{(0.12)}$	$75.65_{(0.14)}$	$80.82_{(0.24)}$	$71.01_{(0.45)}$	$77.28_{(0.45)}$
CL-LoRA† (He et al., 2025)	0.3	$\underline{81.41}_{(0.13)}$	$\underline{84.24}_{(0.23)}$	$\underline{79.55}_{(0.31)}$	$\underline{83.25}_{(0.12)}$	$\underline{76.84}_{(0.34)}$	$\underline{80.99}_{(0.61)}$
SD-LoRA (Wu et al., 2025)	0.4	$79.15_{(0.20)}$	$83.01_{(0.42)}$	$77.34_{(0.35)}$	$82.04_{(0.24)}$	$75.26_{(0.37)}$	$80.22_{(0.72)}$
EWC-LoRA (Zheng et al., 2026)	0.4	$76.36_{(0.21)}$	$81.43_{(0.13)}$	$72.86_{(0.79)}$	$78.95_{(0.86)}$	$70.18_{(1.06)}$	$77.06_{(0.54)}$
AO-LoRA	0.2	$\mathbf{82.20}_{(0.21)}$	$\mathbf{84.88}_{(0.19)}$	$\mathbf{80.94}_{(0.24)}$	$\mathbf{84.55}_{(0.13)}$	$\mathbf{78.85}_{(0.23)}$	$\mathbf{83.37}_{(0.18)}$

4 EXPERIMENTS

4.1 BENCHMARKS

Datasets. To evaluate AO-LoRA, we conduct experiments on four CIL benchmarks: CIFAR-100 (Krizhevsky et al., 2009), ImageNet-R (Hendrycks et al., 2021a; Wang et al., 2022a), ImageNet-A (Hendrycks et al., 2021b), and CUB-200 (Wah et al., 2011). We consider two task partition settings. (1) *Standard setting* (Liang & Li, 2024): CIFAR-100, CUB-200 and ImageNet-A are split into 10 tasks, while ImageNet-R is evaluated with 5, 10, and 20 tasks, corresponding to 40, 20, and 10 classes per task. (2) *Longer sequence setting* (He et al., 2025): CIFAR-100 and ImageNet-R are split into 20 and 40 tasks, respectively, with 5 classes per task.

Evaluation Metrics. We use classification accuracy as the primary evaluation metric. Let $A_{t,i}$ denote the accuracy on task i after the model has been trained up to task t. The average accuracy at step t is computed as $\overline{A}_t = \frac{1}{t}\sum_{i=1}^{t} A_{t,i}$. To summarize performance over the entire training process, we report the mean of these intermediate accuracies, defined as Avg. $= \frac{1}{T}\sum_{t=1}^{T} \overline{A}_t$, where T denotes the total number of tasks. Details of the evaluation protocol are provided in Appendix C.3.

Implementation Details. We validate the method on both supervised and self-supervised pretrained models. Specifically, we adopt a *ViT-B/16* (Dosovitskiy et al., 2020) backbone pretrained on ImageNet-21K, and further evaluate self-supervised variants, including *DINO* (Caron et al., 2021) and *DINOv2* (Oquab et al., 2023). Additional details on the experimental setup are provided in Appendix C.1. For training details and hyperparameters used in this work, please refer to Appendix C.4.

4.2 MAIN RESULTS

Comparison with Various CL Baselines. We benchmark AO-LoRA against state-of-the-art PTM-based continual learning methods, including L2P (Wang et al., 2022b), DualPrompt (Wang et al., 2022a), and CODA-Prompt (Smith et al., 2023) for prompt-based approaches, InfLoRA (Liang & Li, 2024), CL-LoRA (He et al., 2025), SD-LoRA (Wu et al., 2025), and EWC-LoRA (Zheng et al., 2026) for LoRA-based approaches on vision benchmarks. Table 1 reports the results on ImageNet-R across varying numbers of tasks. While achieving superior performance, AO-LoRA uses trainable parameters amounting to only 0.2% of the backbone, demonstrating remarkable parameter efficiency. Table 2 reports the results on CIFAR-100, CUB-200, and ImageNet-A.

Analysis on Stability and Plasticity. We analyze the stability-plasticity trade-off of AO-LoRA and other LoRA-based methods on ImageNet-R. See Appendix C.3 for the formal definitions of the stability and plasticity metrics. As shown in Fig. 4, AO-LoRA clearly outperforms all baselines. Existing methods such as CL-LoRA achieve a relatively balanced trade-off, yet still lag behind AO-LoRA. In contrast, InfLoRA and SD-LoRA cluster in the lower-left region, indicating weaker performance in both aspects, while EWC-LoRA tends to favor stability at the expense of plasticity. These results demonstrate that AO-LoRA effectively balances the two objectives, enabling effective adaptation to new tasks while mitigating forgetting of previously learned knowledge.

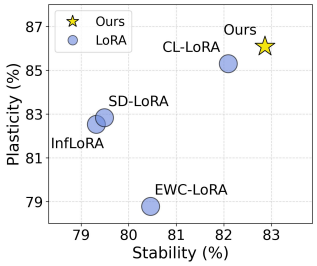


Figure 4: Stability-Plasticity Trade-off across methods.

Table 2: Comparison results on CIFAR-100, CUB-200, and ImageNet-A (in %). '†' indicates reproduced results. The full results are provided in Appendix D.1.

Tasks	CIFAR-100 (N=10)		CUB-200 (N=10)		ImageNet-A (N=10)	
Methods	$\overline{A}_{10}$ (↑)	Avg. (↑)	$\overline{A}_{10}$ (↑)	Avg. (↑)	$\overline{A}_{10}$ (↑)	Avg. (↑)
InfLoRA†	86.51	91.70	72.57	82.76	50.75	64.36
CL-LoRA†	87.65	92.25	<u>78.34</u>	85.79	57.62	<u>70.76</u>
SD-LoRA†	86.77	90.96	77.29	<u>85.91</u>	55.23	66.10
EWC-LoRA	<u>87.91</u>	<u>92.27</u>	74.34	80.49	<u>59.89</u>	68.33
AO-LoRA	**90.09**	**93.65**	**79.27**	**86.54**	**64.44**	**74.97**

Efficiency Comparison with LoRA. For a linear layer with input and output dimension d and adaptation rank r, LoRA introduces two low-rank matrices $A \in \mathbb{R}^{d\times r}$ and $B \in \mathbb{R}^{r\times d}$, resulting in a parameter overhead of $\mathcal{O}(2dr)$. AO-LoRA introduces $\mathcal{O}(dr)$ trainable parameters per adapted linear layer, compared with $\mathcal{O}(2dr)$ for a standard LoRA adapter of the same rank. We compare methods in terms of the trade-off between performance and parameter efficiency on ImageNet-R, as shown in Fig. 5. AO-LoRA achieves the best performance while using substantially fewer parameters. Notably, under a similar parameter budget to InfLoRA, AO-LoRA yields a significant accuracy gain, highlighting the advantage of leveraging structured transformations for continual learning.

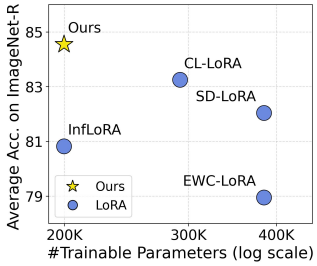


Figure 5: Trainable parameter efficiency on ImageNet-R with 10 tasks.

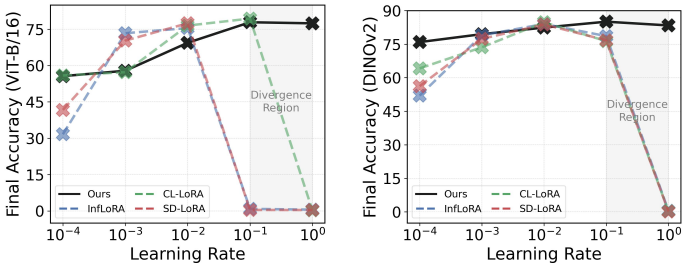


Figure 6: Learning rate sensitivity of AO-LoRA and LoRA-based methods. Performance across a range of learning rates is shown for ViT-B/16 (left) and DINOv2 (right) backbones.

Sensitivity to Learning Rates. LoRA-based methods perform unconstrained additive updates in Euclidean space, where the update magnitude scales with the learning rate, often leading to instability or collapse. AO-LoRA instead constructs task-specific updates using Householder transformations and exhibits greater stability under large learning rates, demonstrating its robustness to learning rate variations. As shown in Fig. 6, on ImageNet-R across a wide range of learning rates, AO-LoRA remains stable, whereas LoRA-based methods are more sensitive and collapse at higher learning rates. This improved robustness can be attributed to Householder reflections, which preserve feature norms and inner products, preventing magnitude amplification and enabling stable optimization. Consequently, AO-LoRA alleviates the need for careful learning rate tuning.

4.3 Ablation Studies

Effect of Different Components. We conduct a detailed ablation study on each component, including anchored orthogonal adaptation (*A-Orth.*), task-specific scaling factor (*Scal.*), and inter-task orthogonality (*I-Orth.*). As shown in Table 3, replacing LoRA with OFT leads to a significant performance degradation (from 74.13% to 62.08%), which verifies that orthogonality alone is insufficient under sequential adaptation. By introducing *A-Orth.*, the performance is substantially recovered, boosting $\overline{A}_{20}$ from 62.08% to 74.53% and even surpassing the LoRA baseline. Furthermore, adding the scaling factor (*Scal.*) and inter-task orthogonality (*I-Orth.*) further improves the final accuracy $\overline{A}_{20}$ to 77.52% and 78.85%, respectively.

Position of AO-LoRA. AO-LoRA is applied by replacing the linear layers within the attention module. We investigate different strategies, specifically replacing the query-key-value (*QKV*) projection layers and the output projection (*Proj.*) layer. As reported in Table 4, the performance is relatively limited when AO-LoRA is applied only to the *QKV* or *Proj.* projection layers. However, combining *QKV* and *Proj.* achieves the best results (90.09% and 80.94% on CIFAR-100 and ImageNet-R, respectively), demonstrating that these components play complementary roles. Although this setting nearly doubles the number of parameters from 1.0×10^5 to 1.9×10^5, the resulting gains in accuracy offer a worthwhile trade-off in parameter efficiency.

Table 3: Ablation study on ImageNet-R with ViT-B/16. The first row shows the LoRA baseline. *OFT* denotes the results obtained by applying orthogonal fine-tuning alone.

OFT	*A-Orth.*	*Scal.*	*I-Orth.*	$\overline{A}_{20}$	Avg.
-	-	-	-	74.13	81.11
✓				62.08	68.59
✓	✓			74.53	82.82
✓	✓	✓		77.52	**83.93**
✓	✓	✓	✓	**78.85**	83.37

Table 4: Effect of applying AO-LoRA to different linear components, reporting $\overline{A}_{10}$ and #Parameters.

Layer	CIFAR-100	ImageNet-R
QKV	89.53	78.08
	#Parameters: 1.0×10^5	
Proj.	<u>89.56</u>	<u>78.93</u>
	#Parameters: 1.0×10^5	
QKV, *Proj.*	**90.09**	**80.94**
	#Parameters: 1.9×10^5	

Evaluation on Longer Task Sequences. We extend the task sequence to $N = 20$ for CIFAR-100 and $N = 40$ for ImageNet-R. The results are reported in Table 5. On CIFAR-100, AO-LoRA attains the best performance in both $\overline{A}_{20}$ and Avg., outperforming all baselines by a large margin. On the more challenging ImageNet-R, AO-LoRA achieves the best average accuracy (79.70%) and remains highly competitive in $\overline{A}_{40}$. These results demonstrate that AO-LoRA performs well in standard settings and remains robust to catastrophic forgetting in longer continual learning scenarios.

Varying the Pretrained Model. We evaluate AO-LoRA initialized with self-supervised models DINO and DINOv2 on ImageNet-R, as shown in Table 6. AO-LoRA achieves the highest $\overline{A}_{10}$ under both backbones. With DINO, it achieves the best $\overline{A}_{10}$ of 70.03%, exceeding CL-LoRA by 1.06%. With the stronger DINOv2 backbone, it further attains 85.05% in $\overline{A}_{10}$ and 89.55% in average accuracy. These results demonstrate the robustness of AO-LoRA across different pretrained models and its effectiveness in continual learning.

Table 5: Results on longer task sequences on CIFAR-100 (N=20) and ImageNet-R (N=40).

Tasks	**CIFAR-100 (N=20)**		**ImageNet-R (N=40)**	
Methods	$\overline{A}_{20}$ (↑)	Avg. (↑)	$\overline{A}_{40}$ (↑)	Avg. (↑)
InfLoRA	80.97	88.84	64.51	73.22
CL-LoRA	85.32	<u>91.02</u>	**71.65**	<u>79.02</u>
SD-LoRA†	<u>85.71</u>	90.84	70.38	76.70
AO-LoRA	**88.55**	**92.35**	<u>70.58</u>	**79.70**

Table 6: Results on self-supervised pretrained models.

	Method	$\overline{A}_{10}$	Avg.
DINO	InfLoRA	68.31	**76.15**
	CL-LoRA	<u>68.97</u>	75.85
	AO-LoRA	**70.03**	<u>76.01</u>
DINO v2	InfLoRA	84.03	87.17
	CL-LoRA	<u>84.10</u>	<u>88.24</u>
	AO-LoRA	**85.05**	**89.55**

5 Conclusion

In this paper, we study continual learning from the perspective of preserving transferable pretrained representations. Our analysis shows that continual learning reduces their accessibility and compresses inter-class representation separation. Motivated by these observations, we propose Anchored Orthogonal Low-Rank Adaptation (AO-LoRA), which maintains a common pretrained reference for task-specific adaptation. Extensive experiments show that AO-LoRA consistently outperforms existing LoRA-based methods across diverse task sequences and pretrained models, while exhibiting improved robustness to large learning rates.

AI USE STATEMENT

In this work, we used generative AI tools to assist with literature search and to improve the clarity, conciseness, and readability of the manuscript. We have not used generative AI tools to generate experimental results, implement the proposed method, interpret experimental results, or formulate mathematical claims. All AI-assisted content was reviewed and verified by the authors, and all experiments and results were independently conducted and validated by the authors. We take full responsibility for the final content of this work, including all text, claims, results, and artifacts produced with the aid of generative AI.

ETHICS STATEMENT

This work does not involve human subjects, personal or sensitive data, or potentially harmful applications. We use publicly available benchmark datasets for research purposes and are not aware of any additional ethical concerns.

REPRODUCIBILITY STATEMENT

The main paper and supplementary materials provide the details necessary to reproduce our method and experiments. The method and training objective are described in Sections 3.2 and 3.3, with the complete learning procedure in Algorithm 1. Experimental settings, including datasets, task partitions, baselines, and implementation details, are provided in Section 4 and the supplementary materials. Additional analyses and results are reported in the appendix.

REFERENCES

Massimo Bini, Karsten Roth, Zeynep Akata, and Anna Khoreva. ETHER: Efficient finetuning of large-scale models with hyperplane reflections. In Proceedings of the 41st International Conference on Machine Learning, volume 235, pp. 4007–4026. PMLR, 21–27 Jul 2024.

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.

Arslan Chaudhry, Naeemullah Khan, Puneet Dokania, and Philip Torr. Continual learning in low-rank orthogonal subspaces. Advances in Neural Information Processing Systems, 33:9900–9911, 2020.

Matthias De Lange, Rahaf Aljundi, Marc Masana, Sarah Parisot, Xu Jia, Aleš Leonardis, Gregory Slabaugh, and Tinne Tuytelaars. A continual learning survey: Defying forgetting in classification tasks. IEEE transactions on pattern analysis and machine intelligence, 44(7):3366–3385, 2021.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. In Proceedings of the 2019 conference of the North American chapter of the association for computational linguistics: human language technologies, volume 1 (long and short papers), pp. 4171–4186, 2019.

Wei Dong, Yuan Sun, Yiting Yang, Xing Zhang, Zhijun Lin, Qingsen Yan, Haokui Zhang, Peng Wang, Yang Yang, and Hengtao Shen. Efficient adaptation of pre-trained vision transformer via householder transformation. Advances in Neural Information Processing Systems, 37:102056–102077, 2024.

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.

Mehrdad Farajtabar, Navid Azizan, Alex Mott, and Ang Li. Orthogonal gradient descent for continual learning. In International conference on artificial intelligence and statistics, pp. 3762–3773. PMLR, 2020.

Jiangpeng He, Zhihao Duan, and Fengqing Zhu. Cl-lora: Continual low-rank adaptation for rehearsal-free class-incremental learning. In Proceedings of the Computer Vision and Pattern Recognition Conference, pp. 30534–30544, 2025.

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, pp. 8340–8349, 2021a.

Dan Hendrycks, Kevin Zhao, Steven Basart, Jacob Steinhardt, and Dawn Song. Natural adversarial examples. CVPR, 2021b.

Alston S Householder. The theory of matrices in numerical analysis. Courier Corporation, 2013.

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.

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.

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.

Shimou Ling, Liang Zhang, Jiangwei Zhao, Lili Pan, and Hongliang Li. Lora-based continual learning with constraints on critical parameter changes. Pattern Recognition, pp. 112086, 2025.

Weiyang Liu, Rongmei Lin, Zhen Liu, Lixin Liu, Zhiding Yu, Bo Dai, and Le Song. Learning towards minimum hyperspherical energy. Advances in neural information processing systems, 31, 2018.

David Lopez-Paz and Marc'Aurelio Ranzato. Gradient episodic memory for continual learning. Advances in neural information processing systems, 30, 2017.

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.

Maxime Oquab, Timothée Darcet, Théo Moutakanni, Huy Vo, Marc Szafraniec, Vasil Khalidov, Pierre Fernandez, Daniel Haziza, Francisco Massa, Alaaeldin El-Nouby, et al. Dinov2: Learning robust visual features without supervision. arXiv preprint arXiv:2304.07193, 2023.

German I Parisi, Ronald Kemker, Jose L Part, Christopher Kanan, and Stefan Wermter. Continual lifelong learning with neural networks: A review. Neural networks, 113:54–71, 2019.

Yu-Yang Qian, Yuan-Ze Xu, Zhen-Yu Zhang, Peng Zhao, and Zhi-Hua Zhou. Treelora: Efficient continual learning via layer-wise loras guided by a hierarchical gradient-similarity tree. arXiv preprint arXiv:2506.10355, 2025.

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.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu. Exploring the limits of transfer learning with a unified text-to-text transformer. Journal of machine learning research, 21(140):1–67, 2020.

Roger Ratcliff. Connectionist models of recognition memory: constraints imposed by learning and forgetting functions. Psychological review, 97(2):285, 1990.

Sylvestre-Alvise Rebuffi, Alexander Kolesnikov, Georg Sperl, and Christoph H Lampert. icarl: Incremental classifier and representation learning. In Proceedings of the IEEE conference on Computer Vision and Pattern Recognition, pp. 2001–2010, 2017.

Andrei A Rusu, Neil C Rabinowitz, Guillaume Desjardins, Hubert Soyer, James Kirkpatrick, Koray Kavukcuoglu, Razvan Pascanu, and Raia Hadsell. Progressive neural networks. arXiv preprint arXiv:1606.04671, 2016.

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, 2023.

Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, et al. Llama: Open and efficient foundation language models. arXiv preprint arXiv:2302.13971, 2023.

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

Huiyi Wang, Haodong Lu, Lina Yao, and Dong Gong. Self-expansion of pre-trained models with mixture of adapters for continual learning. In 2025 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), pp. 10087–10098. IEEE, 2025.

Xiao Wang, Tianze Chen, Qiming Ge, Han Xia, Rong Bao, Rui Zheng, Qi Zhang, Tao Gui, and Xuan-Jing Huang. Orthogonal subspace learning for language model continual learning. In Findings of the Association for Computational Linguistics: EMNLP 2023, pp. 10658–10671, 2023.

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.

Zhixiang Wei, Lin Chen, Yi Jin, Xiaoxiao Ma, Tianle Liu, Pengyang Ling, Ben Wang, Huaian Chen, and Jinjin Zheng. Stronger fewer & superior: Harnessing vision foundation models for domain generalized semantic segmentation. In Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, pp. 28619–28630, 2024.

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.

Yutao Yang, Jie Zhou, Xuanwen Ding, Tianyu Huai, Shunyu Liu, Qin Chen, Yuan Xie, and Liang He. Recent advances of foundation language models-based continual learning: A survey. ACM Computing Surveys, 57(5):1–38, 2025.

Shen Yuan, Haotian Liu, and Hongteng Xu. Bridging the gap between low-rank and orthogonal adaptation via householder reflection adaptation. Advances in Neural Information Processing Systems, 37:113484–113518, 2024.

Yin Zhang, Yongqiang Zhang, Yaoyue Zheng, Bogdan Raducanu, and Dan Liu. Causal-tune: Mining causal factors from vision foundation models for domain generalized semantic segmentation. In Proceedings of the AAAI Conference on Artificial Intelligence, volume 40, pp. 12916–12924, 2026.

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.

APPENDIX

A SCOPE AND SUMMARY OF NOTATION

Scope. We consider a class-incremental continual learning setting with pretrained Transformer models, where tasks arrive sequentially without access to past data. The model is adapted in a parameter-efficient manner with frozen pretrained weights, and evaluated in a task-agnostic setting over all observed tasks.

Notation. We summarize the key notations used throughout the paper.

Symbol	Description
$\boldsymbol{W}_0$	Pretrained weight matrix
$\boldsymbol{W}, \widetilde{\boldsymbol{W}}^{(t)}$	Base and adapted weights at task t
$\mathcal{T}_t, \mathcal{D}_t$	Task t and its dataset
x_k^t, y_k^t	Input and label of the k-th sample in task t
t, i	Task indices
r	Number of Householder transformations
$\boldsymbol{u}_i$	Householder vector
$\boldsymbol{U}_r$	Matrix of Householder vectors $[\boldsymbol{u}_1, \ldots, \boldsymbol{u}_r]$
$\boldsymbol{\Gamma}_r$	Upper triangular matrix for composed transformations
$\mathcal{H}^{(t)}$	Task-specific orthogonal transformation
$\Delta\boldsymbol{W}^{(t)}$	Residual update for task t
α_t	Task-specific scaling factor
$\boldsymbol{V}^{(t)}$	Feature-space directions $\boldsymbol{W}\boldsymbol{U}^{(t)}$
$\mathcal{L}_{ce}^{(t)}$	Cross-entropy loss for task t
$\mathcal{L}_{orth}^{(t)}$	Feature-space orthogonality loss
λ	Regularization coefficient

All vectors are column vectors by default. $\|\cdot\|_F$ denotes the Frobenius norm. Superscripts (t) indicate task-specific quantities, while subscripts denote indices within a task or transformation.

B ANALYSIS OF ANCHORED ORTHOGONAL ADAPTATION

The anchored formulation admits a useful algebraic interpretation. Let $\boldsymbol{W} \in \mathbb{R}^{d_{\text{out}} \times d}$ denote a pretrained weight matrix, and let $\{\mathcal{H}^{(i)}\}_{i=1}^t$ be a sequence of task-specific orthogonal transformations, where $\mathcal{H}^{(i)}(\boldsymbol{W}) = \boldsymbol{W}\boldsymbol{H}^{(i)}$ and ${\boldsymbol{H}^{(i)}}^\top \boldsymbol{H}^{(i)} = \boldsymbol{I}$. AO-LoRA accumulates updates as:

$$\widetilde{\boldsymbol{W}}^{(t)} = \boldsymbol{W} + \sum_{i=1}^{t} \alpha_i \left(\mathcal{H}^{(i)}(\boldsymbol{W}) - \boldsymbol{W} \right). \tag{16}$$

Proposition 1. For any input $\boldsymbol{x} \in \mathbb{R}^{1 \times d}$, the resulting transformation can be written as

$$\boldsymbol{x}\widetilde{\boldsymbol{W}}^{(t)\top} = \sum_{i=0}^{t} \beta_i \boldsymbol{x} {\boldsymbol{H}^{(i)}}^\top \boldsymbol{W}^\top, \tag{17}$$

where $\boldsymbol{H}^{(0)} = \boldsymbol{I}$, $\beta_0 = 1 - \sum_{i=1}^t \alpha_i$, and $\beta_i = \alpha_i$ for $i \geq 1$.

Proof. Substituting $\mathcal{H}^{(i)}(\boldsymbol{W}) = \boldsymbol{W}\boldsymbol{H}^{(i)}$ into the anchored formulation gives

$$\widetilde{\boldsymbol{W}}^{(t)} = \boldsymbol{W}\left(\boldsymbol{I} + \sum_{i=1}^{t} \alpha_i \left(\boldsymbol{H}^{(i)} - \boldsymbol{I}\right)\right). \tag{18}$$

Therefore,

$$\boldsymbol{x}\widetilde{\boldsymbol{W}}^{(t)\top} = \boldsymbol{x}\left(\boldsymbol{I} + \sum_{i=1}^{t} \alpha_i \left(\boldsymbol{H}^{(i)} - \boldsymbol{I}\right)\right)^{\top} \boldsymbol{W}^{\top} \tag{19}$$

$$= \left(\left(1 - \sum_{i=1}^{t} \alpha_i\right)\boldsymbol{x} + \sum_{i=1}^{t} \alpha_i \boldsymbol{x}\boldsymbol{H}^{(i)^{\top}}\right)\boldsymbol{W}^{\top}, \tag{20}$$

which gives the stated result. □

This result shows that the adapted transformation combines multiple task-specific orthogonal transformations, all defined relative to the same pretrained weight $\boldsymbol{W}$. In contrast, sequential composition applies each new transformation to the previously adapted weights, such that the reference for subsequent transformations changes recursively. AO-LoRA instead defines each task-specific update with respect to the fixed pretrained weights, providing a common reference throughout continual adaptation. Importantly, anchoring does not reset the model to the pretrained weights at each task. Rather, the pretrained weights provide a fixed reference for constructing new task-specific updates, while updates learned from previous tasks remain accumulated in $\widetilde{\boldsymbol{W}}^{(t)}$. Consequently, continual learning is preserved through the accumulation of task-specific updates, while their reference remains fixed to the pretrained model.

C Implementation Details

In this section, we present the implementation details of AO-LoRA. We begin with the experimental setup and the optimization procedure in the class-incremental setting, followed by the training configurations and key hyperparameters. We then provide ablation studies to analyze the effects of the scaling factor α, orthogonality coefficient λ, and structural design choices. Finally, we describe the task order settings used in the language continual learning benchmark.

C.1 Experimental Setup

Our experiments are conducted on a single NVIDIA A40 GPU with 48GB of memory. We compare the GPU memory consumption of AO-LoRA with LoRA in Table 7. For certain comparison methods, such as SD-LoRA with 20 and 40 tasks, we utilize two GPUs due to their higher computational demands. Unless otherwise specified, all experiments are performed on a single GPU. We follow commonly adopted preprocessing pipelines for each dataset, including standard normalization and resizing where applicable.

Table 7: GPU memory usage during training.

Method	**Trainable Params**	**GPU Memory (GB)**
LoRA	$\sim$ 0.3-0.4%	12.0
AO-LoRA (Ours)	$\sim$ 0.2%	11.2

We adopt CL-LoRA (He et al., 2025) as our baseline, fine-tune all models and report results from the original papers when available, otherwise reproducing them and marking with †. To contextualize performance, we report both an upper bound (*Joint Train*) and a lower bound (*Finetune*). The upper bound corresponds to joint training on all tasks, while the lower bound sequentially trains on tasks without any mechanism to mitigate forgetting. All results are averaged over five runs with random seeds, and we report the mean and standard deviation.

C.2 Optimization Algorithms

Algorithm 1 summarizes the optimization procedure of AO-LoRA in the class-incremental setting. For each incoming task, AO-LoRA instantiates a new set of task-specific Householder vectors and scaling factors for the adapted linear modules, while keeping the pretrained weights and previously learned adapters fixed. During training, the forward pass applies the accumulated orthogonal updates anchored to the frozen pretrained weights. The current task is optimized using the cross-entropy loss on its data, together with the inter-task orthogonality regularizer defined in Section 3.3. After training, the task-specific adapters are stored and frozen, the classifier is expanded to accommodate newly observed classes, and the task data are discarded.

Algorithm 1: Learning procedure of AO-LoRA.

Input: A sequence of tasks $\{\mathcal{T}_t\}_{t=1}^{T}$ with datasets $\{\mathcal{D}_t\}_{t=1}^{T}$; frozen pretrained weights $\{\boldsymbol{W}_{0,\ell}\}_{\ell\in\mathcal{A}}$ for the adapted linear modules $\mathcal{A}$; Householder rank r; orthogonality coefficient λ.

Output: Learned AO-LoRA adapters $\{\boldsymbol{U}_\ell^{(t)}, \alpha_\ell^{(t)}\}_{t=1,\ell\in\mathcal{A}}^{T}$ and classifier parameters.

Initialize adapter memory $\mathcal{M} \leftarrow \emptyset$;

for $t = 1$ **to** T **do**

  Prepare the current-task classifier for the new classes in $\mathcal{T}_t$;

  foreach adapted module $\ell \in \mathcal{A}$ **do**

    Initialize task-specific Householder vectors

$$\boldsymbol{U}_\ell^{(t)} = [\boldsymbol{u}_{\ell,1}^{(t)}, \ldots, \boldsymbol{u}_{\ell,r}^{(t)}]$$

    and a task-specific scaling factor $\alpha_\ell^{(t)}$;

  while training on $\mathcal{D}_t$ **do**

    foreach adapted module $\ell \in \mathcal{A}$ **do**

      Construct the task-specific Householder transformation

$$\boldsymbol{H}_\ell^{(t)} = \prod_{j=1}^{r} \left(\boldsymbol{I} - 2\boldsymbol{u}_{\ell,j}^{(t)} \boldsymbol{u}_{\ell,j}^{(t)\top} \right), \qquad \|\boldsymbol{u}_{\ell,j}^{(t)}\|_2 = 1.$$

      Form the update anchored at the pretrained weight

$$\Delta\boldsymbol{W}_\ell^{(t)} = \boldsymbol{W}_{0,\ell}\boldsymbol{H}_\ell^{(t)} - \boldsymbol{W}_{0,\ell}.$$

      Use the effective adapted weight in the forward pass:

$$\widetilde{\boldsymbol{W}}_\ell^{(t)} = \boldsymbol{W}_{0,\ell} + \sum_{i=1}^{t} \alpha_\ell^{(i)} \Delta\boldsymbol{W}_\ell^{(i)}.$$

    Compute the inter-task orthogonality loss

$$\mathcal{L}_{orth}^{(t)} = \sum_{\ell\in\mathcal{A}} \sum_{i<t} \left\| \boldsymbol{V}_\ell^{(t)\top} \boldsymbol{V}_\ell^{(i)} \right\|_F^2, \qquad \boldsymbol{V}_\ell^{(i)} = \boldsymbol{W}_{0,\ell}\boldsymbol{U}_\ell^{(i)}.$$

    Compute logits using the model parameterized by $\{\widetilde{\boldsymbol{W}}_\ell^{(t)}\}_{\ell\in\mathcal{A}}$ and optimize

$$\mathcal{L}^{(t)} = \mathcal{L}_{ce}^{(t)} + \lambda\mathcal{L}_{orth}^{(t)},$$

    updating only current-task adapters, scaling factors, and classifier;

  Update the prototype classifier for the newly observed classes using $\mathcal{D}_t$;

  Store the learned task adapters $\{\boldsymbol{U}_\ell^{(t)}, \alpha_\ell^{(t)}\}_{\ell\in\mathcal{A}}$ in $\mathcal{M}$ and freeze them for subsequent tasks;

  Discard $\mathcal{D}_t$;

C.3 Evaluation Metrics

Accuracy. Following prior work (Lopez-Paz & Ranzato, 2017; Chaudhry et al., 2019), we use classification accuracy as the primary evaluation metric. Let $A_{t,i}$ denote the accuracy on task i after the model has been trained up to task t. The average accuracy at step t is computed as:

$$\overline{A}_t = \frac{1}{t}\sum_{i=1}^{t} A_{t,i}. \tag{21}$$

To summarize performance over the entire training process, we report the mean of these intermediate accuracies:

$$\text{Avg.} = \frac{1}{T}\sum_{t=1}^{T} \overline{A}_t, \tag{22}$$

where T denotes the total number of tasks.

Stability and Plasticity. To further analyze the trade-off between retaining previous knowledge and learning new tasks, we adopt stability and plasticity metrics. Stability measures the ability to preserve performance on previously learned tasks, defined as:

$$\text{Stability} = \frac{1}{T-1}\sum_{t=2}^{T} \frac{1}{t-1}\sum_{i=1}^{t-1} A_{t,i}. \tag{23}$$

Plasticity evaluates the capability of adapting to new tasks, defined as:

$$\text{Plasticity} = \frac{1}{T}\sum_{t=1}^{T} A_{t,t}. \tag{24}$$

Stability captures how well the model mitigates forgetting across tasks, while plasticity reflects how effectively it learns newly introduced tasks.

C.4 Hyperparameters

Following prior works (Wu et al., 2025), we adopt the SGD optimizer, with the number of training epochs determined by each dataset. Specifically, we use 20 epochs for CIFAR-100, ImageNet-R, and ImageNet-A, and 10 for CUB-200. The batch size is set to 32 for ImageNet-R, ImageNet-A, and CUB-200, and 64 for CIFAR-100, without data shuffling.

For AO-LoRA, the task-specific scaling factor in Eq. 9 is parameterized as $\alpha = 0.5 \cdot \sigma(\alpha_{\text{raw}})$, where α_{raw} is a learnable parameter. We initialize $\alpha_{\text{raw}} = -3$, corresponding to an effective scaling factor of $\alpha \approx 0.024$. The effect of the scaling factor is analyzed in Fig. 7. The orthogonality coefficient in Eq. 15 is fixed to $\lambda = 10^{-2}$, with its impact on CIFAR-100 and ImageNet-R shown in Fig. 8. For fair comparison with LoRA, we use Householder reflections with rank $r = 10$ in the main experiments, and further analyze the choices of rank r and the number of blocks n in Fig. 9.

Analysis on Scaling Factor α. We study the effect of the task scaling factor α on the plasticity-stability trade-off by varying its initialization from 0 to 0.5, corresponding to increasing model adaptability to new tasks. The results on CIFAR-100 and ImageNet-R are shown in Fig. 7. We observe that the choice of the scaling factor varies slightly across datasets with different levels of difficulty. Specifically, for simpler datasets such as CIFAR-100, a very small value of α close to zero is sufficient. In contrast, for more challenging datasets like ImageNet-R, a relatively larger initial value of α leads to better performance. A common observation is that a large α is not effective. This is similar to the initialization strategy in LoRA, where the up-projection matrix is initialized to zero to stabilize the training process. Based on this observation, we initialize α to 0.024 in all experiments.

Analysis on Orthogonality Coefficient λ. We study the effect of the orthogonality coefficient λ on model performance, with results shown in Fig. 8. Overall, AO-LoRA demonstrates stable performance across a wide range of λ values, indicating low sensitivity to this hyperparameter. As λ

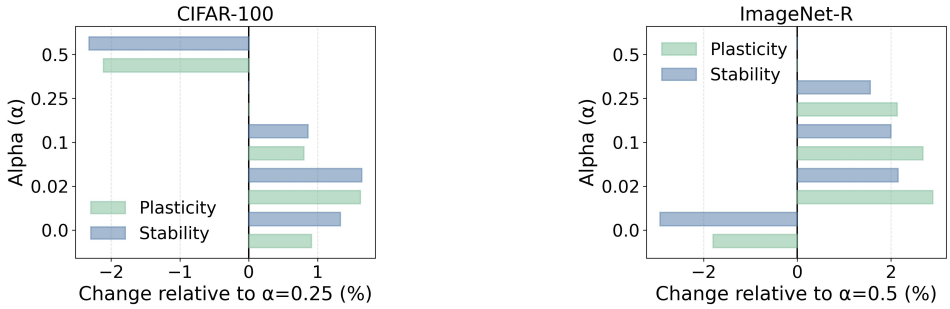


Figure 7: Effect of different α initializations.

increases from 0 to 10^{-2}, performance on both CIFAR-100 and ImageNet-R consistently improves, suggesting that introducing orthogonality regularization effectively enhances representation stability. The best performance is achieved at $\lambda = 10^{-2}$, where the model reaches 90.09% on CIFAR-100 and 80.94% on ImageNet-R. When λ is further increased to 10^{-1}, a slight performance drop is observed, indicating that overly strong orthogonality constraints may limit the model's flexibility to adapt to new tasks. Nevertheless, the degradation is marginal, further confirming the robustness of AO-LoRA to the choice of λ. These results highlight that a moderate orthogonality constraint strikes a favorable balance between stability and plasticity, leading to optimal performance.

Analysis on rank r and blocks n. To investigate the impact of the hyperparameters rank r and number of blocks n, we conduct an ablation study by varying r from 4 to 10 and n from 3 to 12 (i.e., the last n blocks) on ImageNet-R. As shown in Fig. 9, model performance improves significantly as n increases from 3 to 6, and continues to rise more gradually when extending to $n = 9$ and $n = 12$. In contrast, the model is relatively insensitive to the choice of rank r, with the best performance achieved at $r = 8$ and $n = 12$. In the main experiments, we adopt $r = 10$ to maintain consistency with LoRA-based methods; however, these results indicate that smaller values of r are already sufficient to achieve strong performance, highlighting the efficiency of the proposed method.

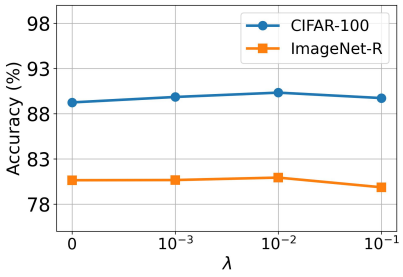


Figure 8: Effect of orthogonality coefficient λ in Eq. 15 on CIFAR-100 and ImageNet-R.

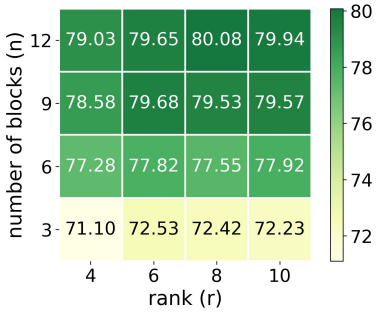


Figure 9: Analysis on Householder reflection rank r and the number of blocks n.

D Additional Experiments

In this section, we provide extended experimental results and analyses to further validate the effectiveness of AO-LoRA. We first present comparison results across multiple datasets and task granularities. We then analyze task-wise performance on language benchmarks to examine stability and plasticity across task orders. To better understand the source of performance gains, we further decompose task-agnostic inference into task-level and class-level components. Finally, we provide a class similarity analysis to visualize how representation structures evolve during continual learning and to demonstrate AO-LoRA's ability to better preserve representation.

D.1 Comparison Results on Vision Benchmarks

The results in Table 8 further validate the effectiveness of our method. Specifically, on CIFAR-100, AO-LoRA achieves the best results with 90.09% accuracy and 93.65% average accuracy, surpassing EWC-LoRA by approximately 2.2% and 1.4%, respectively. On CUB-200, AO-LoRA also achieves the best performance, reaching 79.27% and 86.54%, respectively. On the challenging ImageNet-A, AO-LoRA achieves 64.44% in $\overline{A}_{10}$ and 74.97% in Avg., outperforming the previous best results by 4.55% and 4.21%, respectively. Overall, AO-LoRA consistently achieves the best performance across all three benchmarks, demonstrating its effectiveness under different task sequences.

Table 8: Comparison results on CIFAR-100, CUB-200, and ImageNet-A (in %). **Bold** and underline denote the best and second-best results, respectively. '†' indicates reproduced results.

Tasks	CIFAR-100 (N=10)		CUB-200 (N=10)		ImageNet-A (N=10)	
Methods	$\overline{A}_{10}$ (↑)	Avg. (↑)	$\overline{A}_{10}$ (↑)	Avg. (↑)	$\overline{A}_{10}$ (↑)	Avg. (↑)
Joint Train	$92.82_{(0.20)}$	$95.41_{(0.05)}$	$85.90_{(0.47)}$	$86.75_{(0.22)}$	$65.01_{(0.74)}$	$75.10_{(0.44)}$
Finetune	$79.09_{(1.53)}$	$88.17_{(0.45)}$	$62.93_{(0.78)}$	$68.09_{(0.53)}$	$32.85_{(1.53)}$	$54.55_{(1.44)}$
L2P (Wang et al., 2022b)	$83.18_{(1.20)}$	$87.69_{(1.05)}$	$64.83_{(2.49)}$	$75.55_{(1.27)}$	$42.94_{(1.27)}$	$51.40_{(1.95)}$
DualPrompt (Wang et al., 2022a)	$81.48_{(0.86)}$	$86.41_{(0.66)}$	$68.13_{(1.06)}$	$79.52_{(0.88)}$	$45.49_{(0.96)}$	$54.68_{(1.24)}$
CODA-Prompt (Smith et al., 2023)	$86.31_{(0.12)}$	$90.67_{(0.22)}$	$72.62_{(0.33)}$	$78.45_{(0.65)}$	$45.36_{(0.78)}$	$57.03_{(0.94)}$
InfLoRA† (Liang & Li, 2024)	$86.51_{(0.73)}$	$91.70_{(0.32)}$	$72.57_{(0.23)}$	$82.76_{(0.14)}$	$50.75_{(1.33)}$	$64.36_{(1.01)}$
CL-LoRA† (He et al., 2025)	$87.65_{(0.53)}$	$92.25_{(0.41)}$	$\underline{78.34}_{(0.63)}$	$85.79_{(0.42)}$	$57.62_{(0.89)}$	$\underline{70.76}_{(0.63)}$
SD-LoRA† (Wu et al., 2025)	$86.77_{(0.30)}$	$90.96_{(0.30)}$	$77.29_{(0.20)}$	$\underline{85.91}_{(0.44)}$	$55.23_{(0.94)}$	$66.10_{(0.54)}$
EWC-LoRA (Zheng et al., 2026)	$\underline{87.91}_{(0.57)}$	$\underline{92.27}_{(0.39)}$	$74.34_{(0.64)}$	$80.49_{(0.99)}$	$\underline{59.89}_{(0.26)}$	$68.33_{(0.67)}$
AO-LoRA	$\mathbf{90.09}_{(0.10)}$	$\mathbf{93.65}_{(0.03)}$	$\mathbf{79.27}_{(0.33)}$	$\mathbf{86.54}_{(0.37)}$	$\mathbf{64.44}_{(0.19)}$	$\mathbf{74.97}_{(0.24)}$

Table 9 reports results on CIFAR-100 under different task granularities. AO-LoRA consistently achieves the best performance across all three settings. Compared with the strongest LoRA-based baselines, AO-LoRA improves $\overline{A}_5$, $\overline{A}_{10}$, and $\overline{A}_{20}$, demonstrating that the proposed anchored orthogonal adaptation remains effective from short to longer task sequences. These results complement the ImageNet-R results in Table 1, suggesting that AO-LoRA's advantage is not specific to a single dataset or task partition.

Table 9: Comparison on CIFAR-100 with 5, 10, and 20 tasks (in %). **Bold** and underline denote the best and second-best results, respectively. '†' indicates reproduced results.

Tasks	CIFAR-100 (N=5)†		CIFAR-100 (N=10)		CIFAR-100 (N=20)	
Methods	$\overline{A}_5$ (↑)	Avg. (↑)	$\overline{A}_{10}$ (↑)	Avg. (↑)	$\overline{A}_{20}$ (↑)	Avg. (↑)
InfLoRA (Liang & Li, 2024)	$89.07_{(0.34)}$	$92.98_{(0.17)}$	$86.51_{(0.73)}$	$91.70_{(0.32)}$	$80.97_{(0.74)}$	$88.84_{(0.90)}$
CL-LoRA (He et al., 2025)	$89.01_{(0.14)}$	$92.77_{(0.09)}$	$87.65_{(0.53)}$	$92.25_{(0.41)}$	$85.32_{(0.08)}$	$\underline{91.02}_{(0.12)}$
SD-LoRA (Wu et al., 2025)	$89.47_{(0.46)}$	$92.36_{(0.52)}$	$86.77_{(0.30)}$	$90.96_{(0.30)}$	$\underline{85.71}_{(0.38)}$	$90.84_{(0.39)}$
EWC-LoRA (Zheng et al., 2026)	$\underline{89.61}_{(0.23)}$	$\underline{92.98}_{(0.06)}$	$\underline{87.91}_{(0.57)}$	$\underline{92.27}_{(0.39)}$	$84.79_{(0.66)}$	$90.34_{(0.13)}$
AO-LoRA	$\mathbf{90.99}_{(0.06)}$	$\mathbf{93.80}_{(0.02)}$	$\mathbf{90.09}_{(0.10)}$	$\mathbf{93.65}_{(0.03)}$	$\mathbf{88.55}_{(0.19)}$	$\mathbf{92.35}_{(0.08)}$

We do not apply dataset shuffling in our main experiments. To further analyze its impact, we report results with `shuffle=True` in Table 10 and compare them with other methods. As shown, enabling shuffling leads to negligible performance differences among methods on CIFAR-100. However, on the more challenging ImageNet-A dataset, some methods exhibit a performance gap. This suggests that for more complex datasets, dataset shuffling can significantly affect the final results. Therefore, it should be consistently controlled when making comparisons.

D.2 Performance on Language CL Benchmarks

We further evaluate AO-LoRA on language continual learning. Following prior work (Wang et al., 2023), we adopt the standard language CL benchmark consisting of four text classification datasets: DBpedia, Amazon Reviews, Yahoo Answers, and AG News. The benchmark considers three different task orders to evaluate the robustness of continual learning methods to task ordering. The corresponding task sequences are shown in Table 11. Specifically, Order 1 follows DBpedia →

Table 10: Comparison results with dataset shuffling enabled. '†' indicates reproduced results.

Tasks	CIFAR-100 (N=10)		CUB-200 (N=10)		ImageNet-A (N=10)	
Methods	$\overline{A}_{10}$ (↑)	Avg. (↑)	$\overline{A}_{10}$ (↑)	Avg. (↑)	$\overline{A}_{10}$ (↑)	Avg. (↑)
CL-LoRA† (He et al., 2025)	$87.40_{(0.63)}$	$92.17_{(0.50)}$	$\underline{77.64}_{(0.39)}$	$85.23_{(0.56)}$	$\underline{56.33}_{(1.37)}$	$\mathbf{70.24}_{(0.94)}$
SD-LoRA (Wu et al., 2025)	$\underline{88.01}_{(0.31)}$	$\underline{92.54}_{(0.18)}$	$77.48_{(0.20)}$	$\underline{85.59}_{(0.44)}$	$55.96_{(0.73)}$	$64.95_{(1.63)}$
AO-LoRA	$\mathbf{89.65}_{(0.18)}$	$\mathbf{93.35}_{(0.26)}$	$\mathbf{78.53}_{(0.28)}$	$\mathbf{85.80}_{(0.37)}$	$\mathbf{58.66}_{(0.82)}$	$\underline{67.21}_{(1.68)}$

Amazon → Yahoo → AG, while Order 2 swaps the last two tasks as DBpedia → Amazon → AG → Yahoo. Order 3 adopts a different ordering: Yahoo → Amazon → AG → DBpedia.

Table 11: Task order of the standard CL benchmark for language tasks.

	Task 1	Task 2	Task 3	Task 4
Order 1	DBpedia	Amazon	Yahoo	AG
Order 2	DBpedia	Amazon	AG	Yahoo
Order 3	Yahoo	Amazon	AG	DBpedia

For language continual learning, we evaluate AO-LoRA with two pretrained language models, T5-large (Raffel et al., 2020) and LLaMA-3.2-1B-Instruct (Touvron et al., 2023). The comparison results are reported in Table 12. AO-LoRA achieves the best average performance on both backbones, reaching 76.68% on T5-large and 65.05% on LLaMA-3.2-1B-Instruct. On T5-large, AO-LoRA performs competitively with EWC-LoRA across the three task orders, while obtaining the highest average accuracy. On LLaMA-3.2-1B-Instruct, AO-LoRA achieves consistently higher accuracy than the compared methods across all three task orders, resulting in a 1.97% improvement over EWC-LoRA in average accuracy.

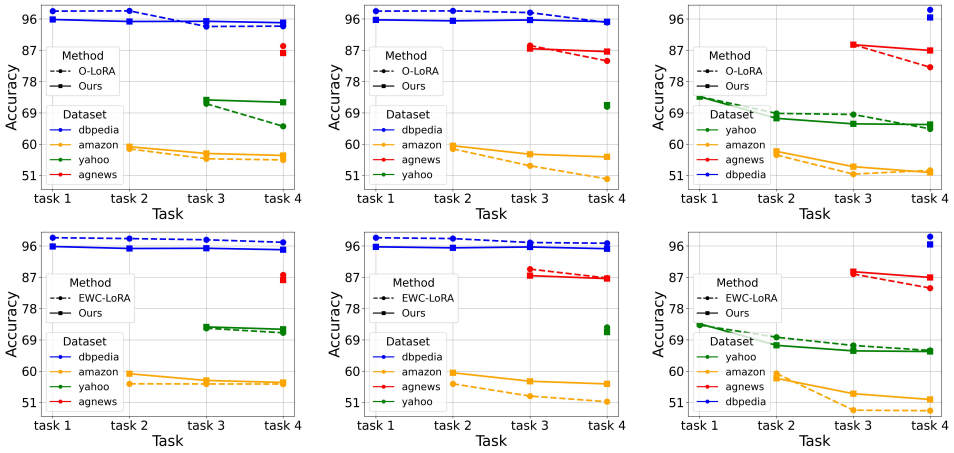
Table 12: Comparison results (in %) on the standard language CL benchmark with two pretrained models.

Backbone	Method	Order 1	Order 2	Order 3	Avg.
T5-large	O-LoRA (Wang et al., 2023)	75.69	74.92	<u>74.40</u>	75.01
	EWC-LoRA (Zheng et al., 2026)	**78.01**	<u>76.85</u>	74.30	<u>76.39</u>
	AO-LoRA	<u>77.48</u>	**77.35**	**75.21**	**76.68**
LLaMA-3.2 -1B-Instruct	O-LoRA (Wang et al., 2023)	56.96	55.74	67.32	60.01
	TreeLoRA (Qian et al., 2025)	58.54	56.96	65.42	60.30
	EWC-LoRA (Zheng et al., 2026)	<u>61.17</u>	<u>60.47</u>	<u>67.61</u>	<u>63.08</u>
	AO-LoRA	**62.28**	**62.12**	**70.75**	**65.05**

To further examine the behavior across the task sequence, Fig. 10 shows the task-wise performance of AO-LoRA, O-LoRA, and EWC-LoRA on T5-large under the three task orders. AO-LoRA generally maintains stronger performance on previously learned tasks while retaining competitive performance on later tasks. In comparison, O-LoRA shows more noticeable degradation on earlier tasks as training progresses. EWC-LoRA provides stronger stability on some earlier tasks, but its performance on later tasks can be lower. These results indicate that AO-LoRA can maintain a favorable balance between retaining previously learned knowledge and adapting to subsequent tasks.

D.3 Task-agnostic Inference Decomposition

To better understand why AO-LoRA improves task-agnostic class-incremental prediction, we decompose the final ImageNet-R (N=10) accuracy into task-level and class-level components in Table 13. Since this diagnostic requires access to model logits, all entries are computed from reproduced checkpoints using the same post-hoc decomposition procedure, test split, task partition, and set of seeds. The checkpoints are trained with the corresponding method-specific hyperparameters following the original implementations or recommended configurations. CL-LoRA achieves the



(a) Task Order 1 (b) Task Order 2 (c) Task Order 3

Figure 10: Task-wise performance across the three task orders on the T5-large model.

highest TIL accuracy, indicating strong within-task discrimination when the task identity is known. However, its larger TIL-CIL gap suggests that many errors arise from task-level misalignment. In contrast, AO-LoRA obtains the highest task-ID accuracy and the smallest TIL-CIL gap, indicating better compatibility among task-specific representations under task-agnostic inference.

Table 13: Task-agnostic inference decomposition on reproduced ImageNet-R (N=10). CIL Acc. is final task-agnostic accuracy ($\overline{A}_{10}$), Task-ID Acc. measures whether the predicted class belongs to the correct task, and TIL Acc. uses the oracle task ID.

Tasks	**ImageNet-R (N=10)**			
Methods	CIL Acc. (↑)	Task-ID Acc. (↑)	TIL Acc. (↑)	TIL-CIL Gap (↓)
InfLoRA	$75.04_{(0.46)}$	$78.86_{(0.74)}$	$87.63_{(0.22)}$	$12.59_{(0.49)}$
SD-LoRA	$71.00_{(0.52)}$	$74.99_{(0.54)}$	$85.24_{(0.33)}$	$14.24_{(0.27)}$
EWC-LoRA	$72.68_{(0.57)}$	$76.03_{(0.46)}$	$87.07_{(0.37)}$	$14.38_{(0.38)}$
CL-LoRA	$\underline{79.55}_{(0.31)}$	$\underline{82.58}_{(0.26)}$	$\mathbf{91.03}_{(0.09)}$	$\underline{11.48}_{(0.29)}$
AO-LoRA	$\mathbf{80.94}_{(0.24)}$	$\mathbf{84.90}_{(0.26)}$	$\underline{90.80}_{(0.16)}$	$\mathbf{9.86}_{(0.25)}$

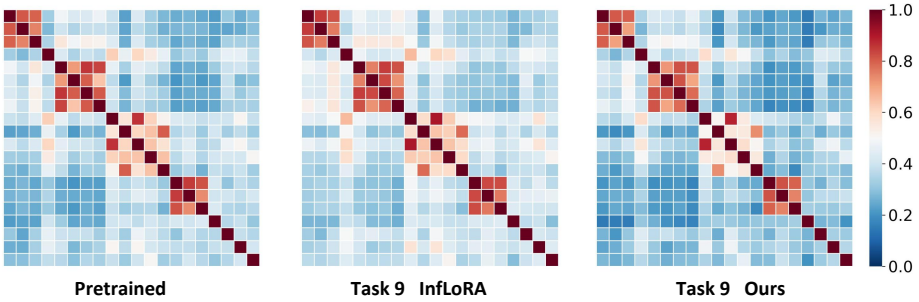


Figure 11: Class-wise cosine similarity on the CUB-200 dataset. The model is continually trained on ImageNet-R, and CUB-200 is used only for direct evaluation to compare with InfLoRA and our method after Task 9.

D.4 Representation Degradation Analysis on an Out-of-Distribution Dataset

To further analyze representation degradation after continual learning, we evaluate the models on an out-of-distribution CUB-200 dataset. Specifically, the models are trained on ImageNet-R and

directly evaluated on CUB-200. We extract class representations and compute their pairwise cosine similarities, using the original pretrained model as a reference. As shown in Fig. 11, the similarity structure produced by InfLoRA after Task 9 exhibits noticeable deviations from that of the pretrained model. In contrast, our method better mitigates representation degradation during continual learning.

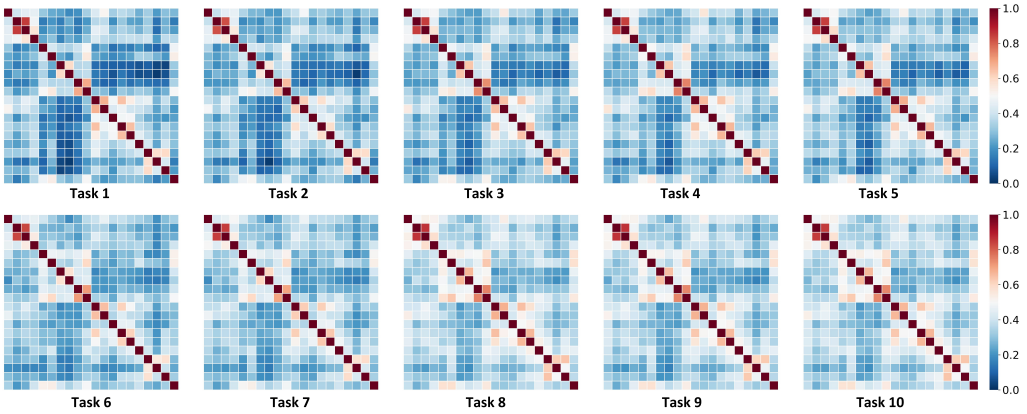


Figure 12: Evolution of class similarity from Task 1 to Task 10 under LoRA-based fine-tuning.

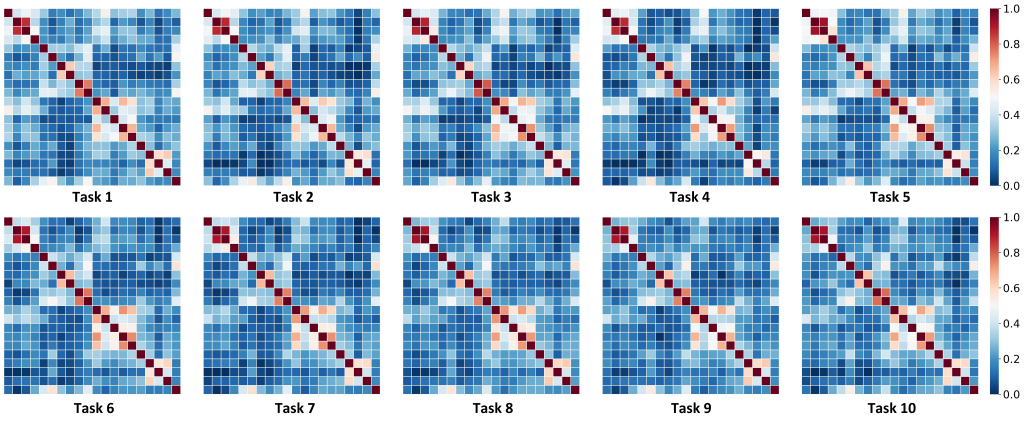


Figure 13: Evolution of class similarity from Task 1 to Task 10 under AO-LoRA-based fine-tuning.

D.5 Class Similarity Analysis

We visualize the inter-class similarity using cosine similarity to examine how representation structures evolve during continual learning. As shown in Fig. 12 and Fig. 13, clear differences emerge between LoRA-based fine-tuning and the proposed AO-LoRA.

For LoRA, the similarity matrices gradually become more diffuse as tasks progress, with off-diagonal entries increasing over time. This indicates that feature representations of different classes become less distinguishable, reflecting accumulated interference and a degradation of class separability. In particular, later tasks exhibit more pronounced cross-class similarity, suggesting that previously learned structures are increasingly distorted. In contrast, AO-LoRA maintains a much sharper and more stable pattern throughout the entire learning process. The inter-class similarities remain consistently low, while intra-class coherence is well preserved. This demonstrates that AO-LoRA effectively maintains the geometric structure of the feature space, reducing representation degradation across tasks. The results provide empirical evidence that preserving structured representations in pretrained models is crucial for mitigating interference and maintaining discriminability in continual learning.