Adaptive Budget Allocation for Orthogonal Subspace Adapter Tuning in LLMs Continual Learning

Anonymous authors
Paper under double-blind review

Abstract

Large language models (LLMs) often suffer from catastrophic forgetting in continual learning (CL) scenarios, where performance on previously learned tasks degrades severely while training on sequentially arriving tasks. Although pioneering CL approaches using orthogonal subspaces can mitigate task interference, they typically employ fixed budget allocation, neglecting the varying complexity across tasks and layers. Besides, recent budget-adaptive tuning methods for LLMs often adopt multi-stage paradigms that decouple optimization and budget allocation. Such decoupling results in potential misalignment, which hinders those approaches' practical application in CL scenarios. To address these limitations, we propose OA-Adapter, a novel parameter-efficient approach for continual learning in LLMs that unifies dynamic budget adaptation with orthogonal subspace learning in an end-to-end training stage. Specifically, OA-Adapter introduces a dynamic bottleneck dimension adaptation mechanism that simultaneously allocates an efficient parameter budget and optimizes task objectives without misalignment. To effectively preserve previously acquired knowledge while coordinating with the dynamic budget allocation, orthogonal constraints are applied specifically between the parameter subspace of the current task and the dynamically allocated parameter subspaces of historical tasks. Experimental results on continual learning benchmarks demonstrate that OA-Adapter outperforms state-of-the-art methods in both accuracy and parameter efficiency. OA-Adapter achieves higher average accuracy while using 58.5% fewer parameters on the Standard CL Benchmark, and maintains its advantages on two larger benchmarks comprising 15 tasks.

1 Introduction

Recent advances in large language models (LLMs) have transformed artificial intelligence by demonstrating remarkable capabilities across diverse domains (Zhou et al., 2024a; Touvron et al., 2023). However, real-world deployment demands that LLMs continually adapt to evolving user needs and emerging tasks while retaining previously acquired knowledge—a prerequisite for sustainable lifelong learning (Xi et al., 2025; Wang et al., 2023b).

Parameter-efficient fine-tuning (PEFT) methods, such as adapter modules (Houlsby et al., 2019) and low-rank adaptation (LoRA) (Hu et al., 2022), enable task-specific adaptation by updating only $0.01\% - 4\%$ of model weights (Balne et al., 2024)). While originally designed to reduce computational costs for single-task tuning (Zhou et al., 2024b; Chen et al., 2024), these methods struggle in continual learning with sequentially arriving tasks. Sequentially tuning to arriving tasks induces catastrophic forgetting-severe degradation of performance on prior tasks (McCloskey & Cohen, 1989). One intuitive solution is to store task-specific adapters for each new task. However, this approach consumes substantial storage resources and leads to considerable inflexibility in multi-task deployments. Alternatively, retraining models with archived historical and additional new data necessitates frequent model updates and large data repositories (Shi et al., 2024). Both strategies are prohibitively costly and impractical in resource-constrained environments.

To efficiently adapt LLMs to downstream tasks while preserving previously acquired knowledge, researchers proposed continual learning (CL) methods. However, most CL approaches operate within shared parameter spaces across tasks, which inherently induces cross-task interference (Wang et al., 2024; Shi et al., 2024; Chaudhry et al., 2019; Shi & Wang, 2023; Rebuffi et al., 2017; Aljundi et al., 2018; Schwarz et al., 2018; Rongali et al., 2020; Lin et al., 2022). Furthermore, unlike conventional CL (e.g., incremental image classification) that typically handle tasks with limited distribution shifts , CL of LLMs often deals with substantially divergent task distributions, thus significantly amplifying interference when tuning in a shared parameter space. Some architecture-based methods further construct task-specific parameters to resolve this problem, but they heavily rely on explicit task ID during inference (Razdaibiedina et al., 2023; Jang et al., 2022; Jin et al., 2022; Li & Lee, 2024; Yan et al., 2023). Recent researches in orthogonal subspace learning (Farajtabar et al., 2020; Guo et al., 2022; Wang et al., 2023a) offer a promising alternative by restricting task-specific updates to mutually orthogonal parameter subspaces to eliminate interference, without task ID dependency during inference. However, existing methods typically rely on a fixed budget allocation, assigning the same subspace dimensionality to different task and layer. This rigid strategy overlooks the heterogeneity of task complexity and layer-specific adaptation needs, leading to inefficient parameter utilization, allocating excessive resource to simple tasks while under-allocating resources to more complex ones. Such inflexible allocation hinder LLMs' continuous adaptation capabilities in practice.

To achieve dynamic budget allocation, emerging budget-adaptive PEFT methods like AdaLoRA (Zhang et al., 2023), ElaLoRA (Chang et al., 2025), and DiffoRA (Jiang et al., 2025) proposed multi-stage paradigms with sequential optimization and budget adjustment phases. Such decoupled optimization may create misalignment between fine-tuning objectives and budget allocation. Moreover, the inherent complexity of multi-stage designs introduces substantial computational overhead and engineering challenges, limiting their practicality for continual learning systems.

We propose OA-Adapter to address these issues, a novel parameter-efficient approach for continual learning in LLMs. Instead of manually assigning a fixed budget, OA-Adapter automatically adjusts the parameter budget for each task and layer based on the task difficulty and model capacity.

Our key contributions are as follows:

- We propose OA-Adapter, a novel parameter-efficient approach for continual learning in LLMs that unifies dynamic budget adaptation with orthogonal subspace learning in an end-to-end training stage. To the best of our knowledge, this is the first work to integrate budget adaptation into parameter-efficient fine-tuning for continual learning in LLMs.
- We design a dynamic bottleneck dimension adaptation mechanism that simultaneously allocates an efficient parameter budget and optimizes task objectives without misalignment.
- We demonstrate that the parameter budget requirements in continual learning vary with task difficulty and position in the training sequence, and that the budget requirements of parameters in different layers vary, which emphasizes the necessity of our approach.
- Experimental results demonstrate that OA-Adapter outperforms state-of-the-art methods in both accuracy and parameter efficiency. OA-Adapter achieves higher average accuracy with 58.5% fewer parameters on the Standard CL Benchmark and maintains its advantages on two larger benchmarks comprising 15 tasks.

2 Related Work.

Continual Learning for LLMs. Existing continual learning (CL) techniques for LLMs are mainly based on replay, regularization, and architecture modification, which have been most extensively applied in this context (Wang et al., 2024; Shi et al., 2024). Replay-based methods (Chaudhry et al., 2019; Shi & Wang, 2023; Rebuffi et al., 2017) store a subset of past data to rehearse on previous tasks during new training. However, this approach raises significant privacy issues and imposes high storage and computational costs, especially for LLMs. Regularization-based methods (Aljundi et al., 2018; Schwarz et al., 2018; Rongali et al.,

2020; Lin et al., 2022) introduce a regularization term that penalizes significant weight deviations across tasks. However, their computational complexity grows rapidly with the number of tasks, and performance on previous tasks deteriorates significantly. These above approaches primarily focus on learning all incremental tasks within a shared parameter space, which is a major contributor to task interference. In contrast, many architecture-based methods (Razdaibiedina et al., 2023; Jang et al., 2022; Jin et al., 2022; Li & Lee, 2024; Yan et al., 2023) address this challenge by incorporating task-specific components, isolated parameters, or dedicated pathways within the model. They essentially learn task-specific expert modules for different tasks, so rely heavily on explicit task ID during inference. Recent advancements have explored a promising direction that can retain generalization capacity while reducing task interference without requiring explicit task ID. These methods (Farajtabar et al., 2020; Guo et al., 2022; Wang et al., 2023a) constrain weight updates for each task to lie within mutually orthogonal subspaces of the high-dimensional parameter space. By doing so, they effectively decouple the optimization directions of different tasks, thereby mitigating task interference. However, a common limitation of existing orthogonal subspace methods is the fixed parameter budget for all tasks and layers.

Adaptive Budget Tuning. Existing budget-adaptive tuning methods for LLMs commonly adopt a multi-stage design where budget adaptation is a separate stage from optimization. For example, DiffoRA (Jiang et al., 2025) trains a LoRA module for each layer in the first stage, and then selects a subset of these modules based on the "Difference-aware Adaptive Matrix" in a subsequent stage. ElaLoRA (Chang et al., 2025) involves a "Warm-up" fine-tuning stage, followed by a "Dynamic Rank Adjustment" stage to adapt LoRA module ranks, and finally a "Stabilization" fine-tuning stage. AdaLoRA (Zhang et al., 2023) injects rank pruning operations after iterations by applying SVD to selectively retain low-rank components. While these methods achieve strong performance on single-task datasets, their multi-stage design introduces substantial computational and engineering overhead that compromises practicality and scalability, while creating misalignment between optimization objectives and budget adaptation criteria that ultimately hinders the achievement of truly optimal solutions. And their design is restricted to unidirectional rank reduction, which does not allow for bidirectional budget adaptation. These limitations pose significant challenges when extending such methods to continual learning, which demands efficient, unified, and adaptive training processes for sequential tasks. Moreover, although recent methods (Liu et al., 2024; Chang et al., 2025; Ding et al., 2023) provide strong interpretability, they rely on task-specific heuristic configurations rather than adapting to the inherent characteristics of sequentially arriving tasks.

3 Methodology

In this section, we introduce OA-Adapter, a novel framework for continual learning in LLMs that simultaneously improves parameter efficiency and mitigates catastrophic forgetting in an end-to-end training stage, as illustrated in Figure 1. We first describe its architectural design, including core components and computation flow. Then, we analyze the mathematical foundations of the dynamic bottleneck dimension adaptation, demonstrating how trainable thresholds enable bidirectional activation and deactivation of dimensions during an end-to-end training phase. Finally, we formalize the orthogonal parameter subspace constraints mechanism and explain how it works in concert with the dynamic bottleneck dimension adaptation to achieve parameter-efficient continual learning.

3.1 Module Structure

Standard Adapter. To enable efficient adaptation of pre-trained language models (PLMs) to downstream tasks, adapter modules inject lightweight trainable parameters into PLMs while keeping the original weights frozen. These modules employ a bottleneck architecture to minimize trainable parameters, consisting of a down-projection layer that reduces the original d-dimensional representation x to a lower dimension r, a nonlinear activation function $f(\cdot)$, and an up-projection layer that restores the features to dimension d. The architecture ensures near-zero initialization of the projection layers while maintaining a skip connection to preserve the original features during initial training stages. For an input representation $x \in \mathbb{R}^d$, the output $y \in \mathbb{R}^d$ can be formalized as:

$$y = x + \mathcal{W}_2 \cdot f(\mathcal{W}_1 \cdot x + b_1) + b_2, \tag{1}$$

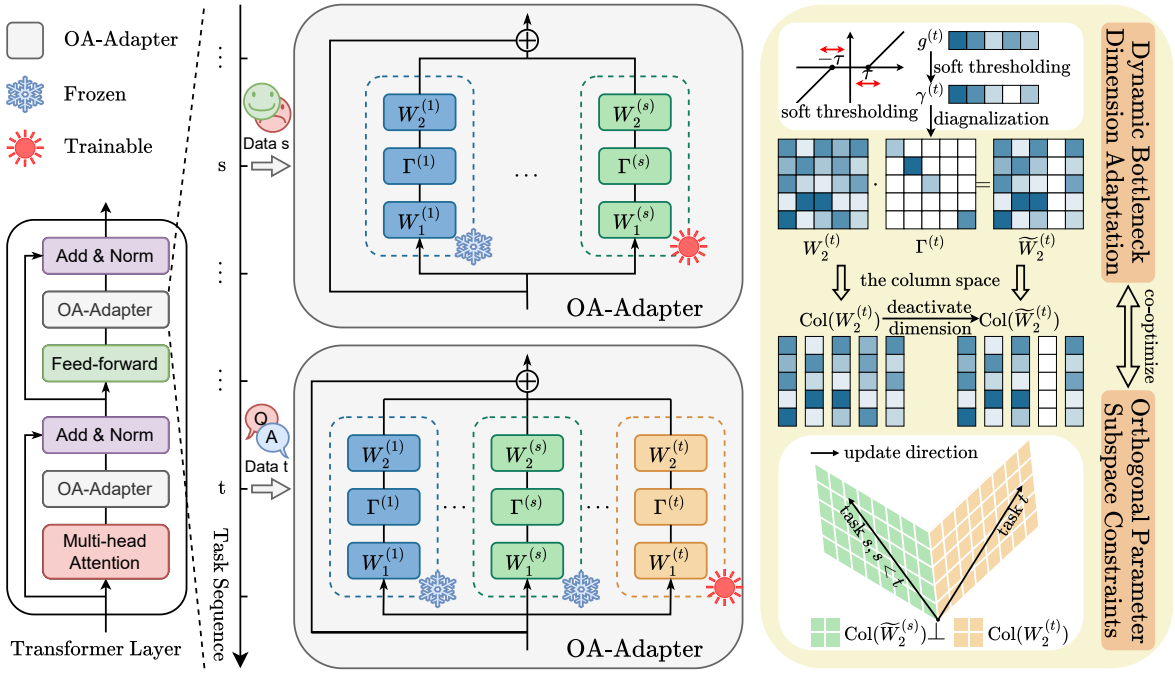

Figure 1: The OA-Adapter framework for LLM continual learning. Each task-specific OA-Adapter module (task t) comprises three core components: (1) a down-projection layer $\mathcal{W}_1^{(t)}$, (2) a diagonal mask $\Gamma^{(t)}$ with trainable threshold $\tau^{(t)}$, (3) and an up-projection layer $\mathcal{W}_2^{(t)}$. The dynamic masking mechanism enables bidirectional dimension adaptation through activation/deactivation of latent dimensions. Orthogonal subspace constraints are enforced between the column space of the t-th task parameters $\mathrm{Col}(\mathcal{W}_2^{(t)})$ and the dynamically allocated parameter subspaces of historical tasks $\mathrm{Col}(\widetilde{\mathcal{W}}_2^{(s)})$ (for $s < t$). Here, $\widetilde{\mathcal{W}}_2^{(s)}$ incorporates only the activated dimensions from the s-th task.

where $\mathcal{W}_1 \in \mathbb{R}^{r\times d}, \mathcal{W}_2 \in \mathbb{R}^{d\times r}$ denote the down-projection and up-projection matrices. $\mathbf{b}_1 \in \mathbb{R}^r$ and $\mathbf{b}_2 \in \mathbb{R}^d$ denote the bias terms, respectively, with bottleneck dimension $r \ll d$.

OA-Adapter. Building upon the standard Adapter's bottleneck architecture, OA-Adapter introduces two structural modifications: 1) the removal of bias terms in projection layers to create a bias-free parameter space containing only linear transformations, and 2) the replacement of static non-linear activations with a trainable diagonal masking matrix Γ that dynamically adjusts the bottleneck dimension, as detailed in Section 3.2. These modifications enable the enforcement of orthogonal parameter subspace constraints, as detailed in Section 3.3, to mitigate cross-task interference and co-optimization of budget adaptation with continual learning in an end-to-end training phase. Specifically, the forward computation of OA-Adapter operates as follows:

$$y = x + \mathcal{W}_2 \cdot \Gamma \cdot \mathcal{W}_1 \cdot x, \tag{2}$$

where $\mathcal{W}_1 \in \mathbb{R}^{r_{\max}\times d}$ and $\mathcal{W}_2 \in \mathbb{R}^{d\times r_{\max}}$ denote the down-projection and up-projection matrices, respectively, and $\Gamma \in \mathbb{R}^{r_{\max}\times r_{\max}}$ is a trainable diagonal masking matrix. Here, $r_{\max} \ll d$ represents the pre-defined maximum bottleneck dimension.

3.2 Dynamic Bottleneck Dimension Adaptation

Adaptation Mechanism. To dynamically allocate parameter budget, we adjust the effective bottleneck dimensions of OA-Adapter using a trainable diagonal masking matrix $\Gamma \in \mathbb{R}^{r_{\max}\times r_{\max}} : \Gamma = \mathrm{diag}(\gamma)$. The sparsity of the vector $\gamma \in \mathbb{R}^{r_{\max}}$ is controlled via a soft thresholding mechanism applied to a trainable vector

$g \in \mathbb{R}^{r_{\max}}$. Specifically, each diagonal entry γ_i is computed as:

$$\gamma_i = \text{soft}(g_i; \tau) = \text{sign}(g_i) \cdot \max(|g_i| - \tau, 0), \tag{3}$$

where $\tau > 0$ is a trainable threshold that dynamically modulates the sparsity level of Γ throughout the training process. The projection path of OA-Adapter can then be equivalently reformulated as:

$$\mathcal{W}_2 \cdot \Gamma \cdot \mathcal{W}_1 = \sum_{i=1}^{r_{\max}} \gamma_i \cdot \mathcal{W}_2[:, i] \otimes \mathcal{W}_1[i, :], \tag{4}$$

where $\otimes$ denotes the outer product. This decomposition clearly demonstrates how each γ_i dynamically adjusts the contribution of the ith latent dimension pair: when $|g_i| \leq \tau$, we have $\gamma_i = 0$, causing both the ith column of $\mathcal{W}_2$ and i^{th} row of $\mathcal{W}_1$ to be disabled, effectively deactivating the corresponding dimension. This mechanism adaptively controls the bottleneck dimension through $r_{\text{eff}} = \|\gamma\|_0$, where $\|\gamma\|_0$ represents the count of non-zero entries in γ.

Gradient Analysis. Our method's bidirectional dimension adaptation capability, enabled by the trainable threshold τ, offers critical advantages. When $|g_i| \leq \tau$, the corresponding dimension pair is deactivated by setting $\gamma_i = 0$. This zeros the i-th diagonal entry of the masking matrix Γ, effectively removing that dimension's contribution in forward propagation. However, this operation also blocks gradient flow from the downstream loss $\mathcal{L}$ to g_i when $\gamma_i = 0$. To clarify why g_i becomes non-trainable in such cases, consider the gradient calculation via chain rule:

$$\frac{\partial \mathcal{L}}{\partial g_i} = \frac{\partial \mathcal{L}}{\partial \gamma_i} \frac{\partial \gamma_i}{\partial g_i}. \tag{5}$$

The derivative $\frac{\partial \gamma_i}{\partial g_i}$ of the soft thresholding function equals $\text{sign}(g_i)$ when $|g_i| > \tau$, but critically becomes 0 when $|g_i| \leq \tau$. This implies that deactivated dimensions (where $|g_i| \leq \tau$) produce zero gradients in Equation (5) due to $\frac{\partial \gamma_i}{\partial g_i} = 0$, blocking gradient updates through g_i. With a fixed threshold τ, such dimensions would remain permanently disabled throughout training. Crucially, our method implements τ as a learnable parameter shared across all dimensions within each OA-Adapter module. Thus, the gradient of τ with respect to the total loss $\mathcal{L}$ is:

$$\frac{\partial \mathcal{L}}{\partial \tau} = \sum_{i=1}^{r_{\max}} \frac{\partial \mathcal{L}}{\partial \gamma_i} \frac{\partial \gamma_i}{\partial \tau}, \tag{6}$$

where $\frac{\partial \gamma_i}{\partial \tau} = -\text{sign}(g_i)$ when $|g_i| > \tau$ and 0 otherwise. This derivative relationship ensures threshold updates are primarily governed by dimensions exceeding the current τ. As τ evolves during training, dimensions previously deactivated with $|g_i| \leq \tau$ may become reactivated when they satisfy $|g_i| > \tau$ under the updated threshold, thereby reactivating their corresponding projection paths. This bidirectional adaptation mechanism automatically suppresses dimensions while maintaining their potential for reactivation in later training iterations. The bidirectional nature of this dynamic parameter budget adaptation approach ensures optimal parameter allocation that continuously adapts to the evolving requirements of sequential tasks in continual learning.

3.3 Orthogonal Parameter Subspace Constraints for Continual Learning

Continual Learning Setup. Continual learning focuses on incrementally acquiring knowledge from evolving data distributions of sequential tasks while mitigating catastrophic forgetting of previously acquired knowledge. Formally, models are trained on a sequential stream of tasks denoted as $\{D_1, D_2, \cdots, D_t\}$. Each task $D_t = (x_t^i, y_t^i)_{i=1}^{n_t}$ consists of input instances $x_t^i \in \mathcal{X}_t$ paired with corresponding labels $y_t^i \in \mathcal{Y}_t$, where $\mathcal{X}_t$ and $\mathcal{Y}_t$ represent the task-specific input and label spaces. During the training phase for task D_t, model parameters Φ are updated exclusively using data from D_t. The objective of continual learning can be formalized as optimizing:

$$\max_{\Phi} \sum_{t=1}^{T} \sum_{\{x_i^t, y_i^t\} \in \mathcal{D}_t} \log P_{\Phi}(y_t^i \mid x_t^i). \tag{7}$$

Orthogonal Parameter Subspace Constraints. Catastrophic forgetting arises when subsequent adaptations overwrite parameters critical for previous tasks. To mitigate this, we introduce orthogonality constraints that enforce parameter updates across tasks to occupy mutually independent subspaces. Let $\Delta\Phi_k$ represent the OA-Adapter's parameter for the k-th task. We have:

$$\Delta\Phi_k = \mathcal{W}_2^{(k)} \cdot \Gamma^{(k)} \cdot \mathcal{W}_1^{(k)} = \widetilde{\mathcal{W}}_2^{(k)} \cdot \mathcal{W}_1^{(k)}. \tag{8}$$

Here, the columns of $\widetilde{\mathcal{W}}_2^{(k)}$ serve as orthogonal basis vectors spanning the parameter update subspace for the k-th task, while $\mathcal{W}_1^{(k)}$ determines how these basis vectors are combined. We therefore formally define the task-specific parameter subspace as the column space of $\widetilde{\mathcal{W}}_2^{(k)}$, which intrinsically aligns with the activated dimensions for the k-th task through the dimension-selective masking operation of $\Gamma^{(k)}$. Thus, we enforce strict orthogonality to new OA-Adapter parameters across sequential tasks, ensuring new task adaptations occupy parameter subspaces orthogonal to previous tasks' frozen parameter subspaces. Formally, the constraints for the t-th task are defined as:

$$\langle \mathcal{W}_2^{(t)}[:,i], \widetilde{\mathcal{W}}_2^{(s)}[:,j] \rangle = 0, \ \forall i, \ j, \ s < t. \tag{9}$$

The columns of $\widetilde{\mathcal{W}}_2^{(t)}$ inherit directional properties from $\mathcal{W}_2^{(t)}$, ensuring orthogonal relationships persist regardless of dynamic dimension activation patterns. These asymmetric orthogonality constraints enable simultaneous optimization of dynamic bottleneck dimension adaptation and historical knowledge preservation. To formalize this approach, we incorporate an orthogonality regularization term into the optimization objective. Specifically, the pairwise orthogonality loss between current task t and each historical task $s < t$ is quantified as:

$$\mathcal{L}_{\text{orth}}^{(s,t)} = \sum_{i,j} \left\langle \mathcal{W}_2^{(t)}[:,i], \widetilde{\mathcal{W}}_2^{(s)}[:,j] \right\rangle^2. \tag{10}$$

Minimizing the loss term $\mathcal{L}_{\text{orth}}^{(s,t)}$ drives the inner product $\langle \mathcal{W}_2^{(t)}[:,i], \widetilde{\mathcal{W}}_2^{(s)}[:,j] \rangle$ toward zero, enforcing parameter subspace orthogonality. The complete training objective, integrating both task-specific performance term $\mathcal{L}_{\text{task}}^{(t)}$ and orthogonality constraints term $\mathcal{L}_{\text{orth}}^{(s,t)}$, is formulated as:

$$\mathcal{L}_{\text{total}} = \mathcal{L}_{\text{task}}^{(t)} + \lambda_{\text{orth}} \cdot \sum_{s<t} \mathcal{L}_{\text{orth}}^{(s,t)}, \tag{11}$$

where λ_{orth} is a hyperparameter controlling the strength of orthogonal regularization.

The synergistic benefits of adaptive rank allocation and orthogonal subspace constraints. In Equations equation 8 and equation 9, OA-Adapter deliberately enforces orthogonality on the effective up-projection $\mathcal{W}_2^{(k)}\Gamma^{(k)}$ rather than on $\Gamma^{(k)}\mathcal{W}_1^{(k)}$. This choice is not arbitrary: it establishes task-specific adaptations in mutually orthogonal subspaces, which directly enables efficient adaptive rank allocation while maximizing parameter utilization in continual learning. To understand the intuition, first consider a standard low-rank adapter without gating. The output is $y = \mathcal{W}_2(\mathcal{W}_1 x)$, which can be expressed as a linear combination $y = \sum_{i=1}^{r} \alpha_i w_{2,i}$, where the columns of $\mathcal{W}_2$ define the basis directions injected into the hidden state, and $\mathcal{W}_1$ supplies input-dependent coefficients. Thus, the up-projection $\mathcal{W}_2$ determines the actual directions added to the representation. With the task-specific diagonal gating $\Gamma^{(k)}$, OA-Adapter selectively activates a subset of these directions. By constraining orthogonality on $\mathcal{W}_2^{(k)}\Gamma^{(k)}$—the precise directions that are injected—we ensure that different tasks occupy orthogonal subspaces in the high-dimensional hidden state space. This prevents interference: subsequent tasks cannot overwrite or align with previously learned directions. This combination of adaptive rank allocation and orthogonal subspace constraints yields critical synergistic benefits. The adaptive gating mechanism significantly simplifies the computation of the orthogonality constraints: since only the active (non-zero) dimensions in $\Gamma^{(k)}$ contribute to the effective up-projection, the constraint is enforced over a reduced effective rank, lowering the computational cost of the orthogonal regularization during training. More importantly, in long task sequences, earlier tasks typically converge to lower effective ranks for simpler objectives, leaving a larger portion of the total parameter space unexplored and available

as orthogonal directions. This preserves greater capacity for subsequent tasks, allowing harder or later tasks to explore and claim higher ranks without encroaching on prior task subspaces—a capability that would be severely limited under fixed-rank or non-orthogonal adaptations. Enforcing orthogonality on the alternative $\Gamma^{(k)}\mathcal{W}_1^{(k)}$ would fail to preserve these advantages: the subsequent multiplication by $\mathcal{W}_2^{(k)}$ would generally violate orthogonality in the injected directions, reintroducing interference and diminishing the benefits of adaptive allocation.

4 Experiments and Analysis

4.1 Experimental Settings.

Datasets. We evaluate OA-Adapter using three CL benchmarks for LLMs: 1) **Standard CL Benchmark** (Zhang et al., 2015): a continual learning benchmark comprising five text classification datasets: AG News, Amazon Reviews, Yelp Reviews, DBpedia, and Yahoo Answers. 2) **Long Sequence Benchmark** (Razdaibiedina et al., 2023): a continual learning benchmark of 15 classification datasets, including five tasks from the Standard CL Benchmark, four from the GLUE benchmark (MNLI, QQP, RTE, SST2) (Wang et al., 2019b), five from the SuperGLUE benchmark (WiC, CB, COPA, MultiRC, BoolQA) (Wang et al., 2019a), and the IMDB movie reviews dataset (Maas et al., 2011). 3) **SuperNI Benchmark** (Wang et al., 2022a): a benchmark of diverse NLP tasks with expert-written instructions, enabling rigorous benchmarking of practical continual learning settings for LLMs. Specifically, we focus on five task categories: dialogue generation, information extraction, question answering, summarization, and sentiment analysis. From each category, three tasks are selected, resulting in a sequence of 15 tasks. Following Razdaibiedina et al. (2023); Zhao et al. (2024), for all benchmarks we randomly select 1,000 samples per class for training. The task details and training sequences of tasks used in our experiments are provided in Appendix A.2.

Metrics. Let $a_{i,j}$ denote the test accuracy on the i-th task after training on the j-th task, the metrics for evaluating are: 1) **Average Accuracy(AA)** (Chaudhry et al., 2018) over all tasks after completing training on the final task, i.e., $\text{AA}_T = \frac{1}{T}\sum_{i=1}^{T} a_{i,T}$. 2) **Forgetting Rate (F.Ra)** (Chaudhry et al., 2018) measures how much knowledge has been forgotten across the first $T-1$ tasks, i.e., $FR_T = \frac{1}{T-1}\sum_{t=1}^{T-1}\left(\max_{k\in\{1,\ldots,T-1\}} a_{k,t} - a_{T,t}\right)$. 3) **Backward Transfer (BWT)** (Ke & Liu, 2022) which measures how much the learning of subsequent tasks influences the performance of a learned task, i.e.,$\text{BWT}_T = \frac{1}{T-1}\sum_{t=1}^{T-1}\left(a_{T,t} - a_{t,t}\right)$

Baselines. We compare our method against various CL baseline approaches, including: **Replay** finetunes the whole model with a memory buffer, and replay samples from old tasks when learning new tasks to avoid forgetting. **L2P (Wang et al., 2022b)** uses the input to dynamically select and update prompts from the prompt pool in an instance-wise fashion. **LFPT5 (Qin & Joty, 2021)** continuously train a soft prompt that simultaneously learns to solve the tasks and generate training samples, which are subsequently used in experience replay. **O-LoRA (Wang et al., 2023a)** train new LoRA parameters on a sequential series of tasks in orthogonal subspace while fixing the LoRA matrices of previous tasks. **ProgPrompt (Razdaibiedina et al., 2023)** adopts a task-specific soft prompt for each distinct task, sequentially appending it to prior learned prompts. In essence, it trains individual models per task, leveraging the task ID to select the appropriate model during inference. In our continual learning baseline selection, we specifically focused on methods that could be reliably reproduced to ensure fair comparison. Furthermore, to guarantee the authenticity and consistency of our experimental results, we reproduced baselines in our infrastructure.

Implementation Details. All our experiments involving T5 models were performed on a server outfitted with four NVIDIA GeForce RTX 3090 GPUs, utilizing the DeepSpeed repository for implementation. Following previous studies (de Masson d'Autume et al., 2019; Rao et al., 2019), for each dataset, we use the available validation set as the test set(since test data is not available) and hold out 500 samples from the training set to construct the validation set. For every sequence of tasks across different orders, we trained the models for one epoch using a batch size of 32 (8 per GPU), a dropout rate of 0.1, and no weight decay. Across all experiments, we primarily used Adapter modules with a bottleneck dimension of

Table 1: Testing performance on three benchmarks with T5-large, ↑ indicates higher values are better and ↓ indicates lower values are better.

	Standard CL Benchmark			Long Sequence Benchmark			SuperNI Benchmark		
	AA↑	F.Ra↓	BWT↑	AA↑	F.Ra↓	BWT↑	AA↑	F.Ra↓	BWT↑
Replay	57.8	12.3	-13.4	54.2	13.5	-12.2	20.5	14.7	-15.8
L2P	60.7	15.3	-11.2	56.1	19.4	-16.6	12.7	11.9	-8.0
LFPT5	72.6	9.1	-8.3	68.6	12.7	-12.8	26.7	15.0	-14.5
O-LoRA	75.3	12.8	-9.1	68.7	7.0	-4.1	25.9	26.4	-24.6
ProgPrompt	75.1	11.3	-8.1	63.2	9.4	-6.7	23.3	13.7	-13.2
OA-Adapter	**76.0**	**8.7**	**-7.5**	**69.2**	**4.4**	**-3.2**	**29.3**	**8.2**	**-6.0**

16, and applied a sparsification threshold chosen from $\{1e\text{-}3, 1e\text{-}4, 1e\text{-}5\}$. The learning rate was selected from $\{5e\text{-}3, 3e\text{-}3, 1e\text{-}3, 5e\text{-}4\}$ depending on task characteristics. We applied an orthogonality regularization on the Adapter's upsampling matrix with a coefficient $\lambda_{orth} \in \{0.5, 1, 5\}$, and used an additional coefficient $\lambda_2 \in \{0, 0.1, 0.5\}$ to scale the associated L2 loss term. To ensure experimental comparability, we maintain consistency with O-LoRA (Wang et al., 2023a) by adopting instruction tuning as the training paradigm. The specific instructions used for each task are detailed in Appendix A.3.

4.2 Main Results

Our experiments employ both the encoder-decoder T5 model (Raffel et al., 2020) and the decoder-only LLaMA-7B model (Touvron et al., 2023), consistent with baselines in CL for NLP. In Table 1, we present the main results using T5-large across three CL benchmarks. Following previous works (Qin & Joty, 2021; Wang et al., 2023a), we report the results of three independent runs with different task orders on each benchmark. All experimental results are reported as the average of three runs.

Results on Continual Learning Benchmarks. As illustrated in Table 1, across all task orders of the Standard CL Benchmark, OA-Adapter consistently surpasses previous methods by a significant margin. On the other two benchmarks comprising 15 tasks, OA-Adapter consistently outperforms previous methods. Notably, on the SuperNI dataset, which is primarily designed for generative tasks, CL methods still perform poorly, underscoring that continual learning for complex tasks remains a significant challenge. To disentangle the roles of orthogonal constraints and adaptive budget, we compare the full OA-Adapter with two variants: a no-orthogonality variant that removes the orthogonal parameter subspace constraint, and a fixed budget variant that disables dynamic bottleneck dimension adaptation. As shown in Table 2, removing orthogonality leads to significant degradation, underscoring its importance in mitigating catastrophic forgetting; disabling adaptation also reduces performance, indicating that adaptive capacity allocation provides additional gains.

Table 2: Ablation study to evaluate the effects of orthogonality and dynamic budget adaptation using T5-large.

Method	Dataset		
	Standard	Long	SuperNI
OA-Adapter	76.0	69.2	29.3
- Orthogonality	57.1	52.8	13.7
- Budget Adaptation	73.3	65.6	24.2

Table 3: Comparisons of parameter efficiency between OA-Adapter and O-LoRA using T5-large.

Method	Budget Allocation	Params	AA
O-LoRA	16→16	4.72M	75.3
OA-Adapter	16→9.95	1.96M-58.5%	76.0+0.7
O-LoRA	8→8	2.36M	74.5
OA-Adapter	8→6.05	1.18M-50.0%	74.7+0.2
O-LoRA	4→4	1.18M	73.8
OA-Adapter	4→3.18	0.63M-46.6%	74.1+0.3

Parameter Efficiency Analysis. As established in Section 3.2, OA-Adapter leverages dynamic parameter budget allocation to achieve enhanced parameter efficiency. We compare the parameter utilization between OA-Adapter and O-LoRA across various initial budget conditions, as illustrated in Table 3. Pa-

rameter budget represents bottleneck dimension distribution for OA-Adapter, while determines the module's intrinsic rank for O-LoRA. Remarkably, OA-Adapter achieves superior performance while using 46.6% to 58.5% fewer parameters compared to O-LoRA's fixed budget approach. Moreover, OA-Adapter maintains consistent performance excellence across all initial budget settings, demonstrating its robust dynamic allocation capabilities. These results highlight OA-Adapter's ability to effectively adapt parameter budget allocation, providing substantial efficiency benefits over static parameter budget allocation approaches.

Heterogeneous Budget Requirements Across Tasks and Layers. Intuitively, adapting to individual downstream datasets requires varying parameter budgets across different tasks and layers. To validate this, we analyze budget allocation patterns in CL scenarios, as shown in Figure 2. Our results reveal heterogeneous budget requirements across tasks and layer positions, confirming that optimal parameter allocation cannot follow uniform rules but demands task-specific consideration. Notably, in CL scenarios, the parameter matrices for the initial task exhibit significantly higher sparsity compared to subsequent tasks. This pattern supports our hypothesis that initial tasks primarily leverage capabilities inherent in the pretrained model, while later tasks must additionally preserve knowledge from preceding tasks, necessitating more complex parameter spaces. Comprehensive analysis is provided in Appendix A.4. These findings validate the necessity of adaptive budget allocation for CL based on the characteristics of layer, task and training sequence. Furthermore, this adaptive allocation aligns with established findings in the intrinsic dimensionality literature, which demonstrate that task complexity is positively correlated with the minimal intrinsic dimension required for successful optimization (Li et al., 2018; Gressmann et al., 2020; Aghajanyan et al., 2021). For instance, more challenging objectives typically exhibit higher intrinsic dimensions, necessitating greater parameter capacity (Li et al., 2018). In our experiments, we observe consistent empirical patterns that support this relationship: simpler topic classification tasks (e.g., AG News and DBpedia) systematically converge to lower average final ranks than more nuanced sentiment analysis tasks (e.g., Amazon Reviews and Yelp Reviews), as evidenced across multiple task orders in Figure 2 and Appendix A.4. This alignment provides additional quantitative grounding for the intuition that harder tasks receive more parameters, reinforcing the effectiveness of OA-Adapter's dynamic budget adaptation mechanism.

4.3 Discussions and Analysis

Occurrence and Mitigation of Catastrophic Forgetting. As shown in Figure 3, we demonstrate the occurrence and mitigation of catastrophic forgetting on the Standard CL Benchmark. Solid and dashed lines denote models with and without orthogonal parameter-subspace constraints, and each color corresponds to a distinct task. Shaded background bands mark the training phase of each task, and the X-axis is scaled proportionally to training steps. We observe a significant decline of each task after their training phase without orthogonal constraints. Performance on Task 2 drops to nearly zero by the end of the subsequent tasks, while Task 1 and Task 3 also decrease substantially. In contrast, the performance with orthogonal constraints remains largely maintained through Task 4, with the most severe degradation limited to only 14% performance loss on Task 2. These results demonstrate that severe forgetting phenomena occur during multi-task training and orthogonal parameter subspace constraints can effectively mitigate it. Similar trends are consistently observed under other task orders, as detailed in Appendix A.5. Additionally, as expected, each task achieves higher performance more rapidly during its corresponding training phase without orthogonal constraints, though final performance is not substantially higher than with orthogonal constraints. This suggests that the model has greater flexibility in parameter update directions when not constrained to preserve previous knowledge, and orthogonal subspace still maintain sufficient capacity for subsequent tasks. Interestingly, tasks exhibit brief performance recovery at the beginning of subsequent training phases before further decline. Performance on Task 1 recovers to nearly 100% at the start of Task 3 and to approximately 60% during Task 4. Although Task 2 drops to near-zero by the end of Task 3, it improves rapidly early in Task 4 training. This behavior resembles human memory, where disused knowledge can be reactivated with minimal effort. This suggests that catastrophic forgetting may only superficially mask a latent capacity, and knowledge representations remain partially preserved despite significant performance degradation.

Mitigating Capacity Saturation in Late-Stage Tasks. A critical challenge in long-sequence continual learning is the saturation of parameter space. Fixed-budget methods like O-LoRA impose rigid constraints

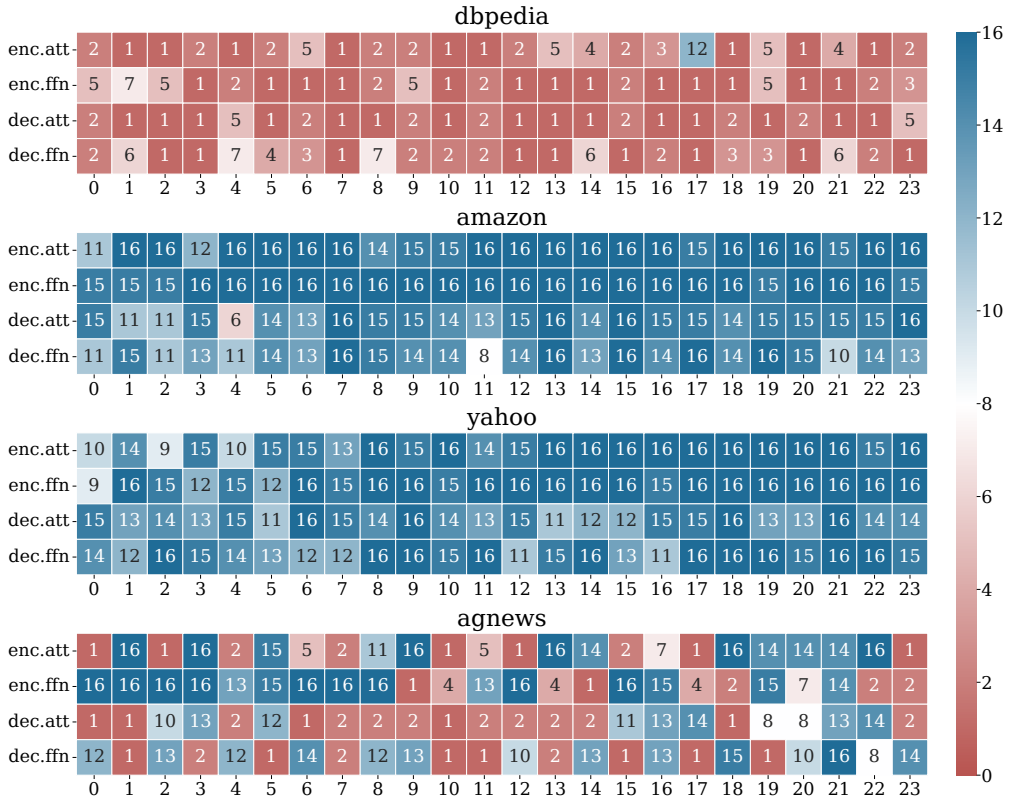

Figure 2: Final dimensions after sequential training following Order-1 using T5-large with OA-Adapter across four datasets (i.e., DBpedia, Amazon, Yahoo, AG News). The X-axis is the index of T5-large layers, and the Y-axis indicates different layers OA-Adapter applies to.

(e.g., fixed rank or strict orthogonality within a limited subspace), which can exhaust the model's learnable capacity as the task sequence lengthens. This often results in suboptimal performance for tasks encountered later in the sequence. To verify the effectiveness of our adaptive budget mechanism in mitigating this issue, we analyze the performance of the final five tasks in the Long Sequence Benchmark (Tasks 11-15) using the T5-Large model following Order 4. We compare OA-Adapter against O-LoRA and Single-Task Learning (STL), which serves as the theoretical upper bound. As illustrated in Table 4, O-LoRA exhibits a noticeable performance gap compared to STL on these late-stage tasks, suggesting that the remaining parameter space is insufficient to capture the specific features of these new tasks effectively. In contrast, OA-Adapter dynamically allocates bottleneck dimensions based on task difficulty. Consequently, it achieves performance nearly identical to the STL upper bound. This result confirms that our adaptive approach prevents capacity saturation, ensuring that the model retains sufficient plasticity to learn new, complex tasks even at the end of a long training sequence.

Table 4: Performance comparison on the final 5 tasks of the Long Sequence Benchmark Order 4. Small red values indicate the performance gap relative to Single-Task Learning (STL).

Method	**Task 11**	**Task 12**	**Task 13**	**Task 14**	**Task 15**
STL	92.4	94.8	86.1	74.3	62.2
OA-Adapter	92.1 -0.3	94.5 -0.3	85.8 -0.3	73.8 -0.5	61.5 -0.7
O-LoRA	86.5 -5.9	91.0 -3.8	79.2 -6.9	65.4 -8.9	54.1 -8.1

Budget Adaptation Mechanism Analysis. To assess the impact of our threshold strategy, we compare two policies: (a) a fixed, non-learnable threshold and (b) our proposed learnable threshold that adapts across

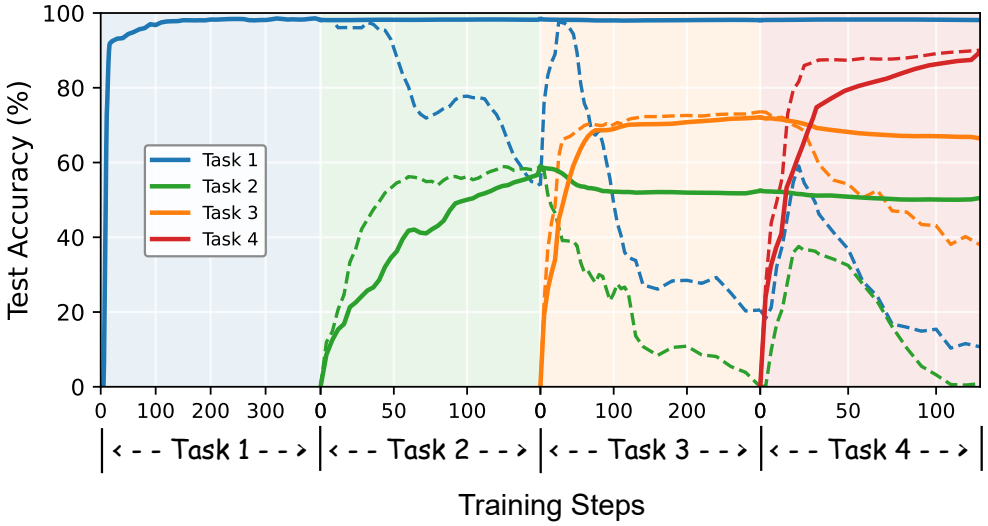

Figure 3: Occurrence and mitigation of catastrophic forgetting during sequential training following Order-1 using T5-large across multiple tasks.

different layers and tasks during training, as described in Section 3.2. These strategies are assessed using the T5-large model across three task orders on the Standard CL Benchmark. The results, depicted in Table 5, reveal that the dynamic threshold consistently demonstrates superior performance compared to the fixed threshold. This confirms that the dynamic threshold mechanism enables bidirectional adjustment of budget allocation, thereby enhancing more flexibility in the optimization process.

Table 5: Comparisons of threshold strategies.

Threshold Initial Value	Threshold Strategy	Order 1	Order 2	Order 3	avg
1e-3	fixed	71.5	71.1	71.4	71.3
	dynamic	**73.0**	**76.4**	**74.6**	**74.7**
1e-4	fixed	73.7	73.1	71.4	72.7
	dynamic	**75.7**	**75.6**	**75.1**	**75.5**
1e-5	fixed	72.8	73.5	72.3	72.9
	dynamic	**74.8**	**74.9**	**73.9**	**74.5**

Table 6: Comparisons of pre-trained models.

Model	Method	Order 1	Order 2	Order 3	avg
T5-base	O-LoRA	73.9	**74.8**	74.1	74.3
	OA-Adapter	**74.7**	**74.8**	**74.5**	**74.7**
T5-large	O-LoRA	75.0	75.4	75.6	75.3
	OA-Adapter	**75.7**	**76.2**	**76.1**	**76.0**
T5-XL	O-LoRA	77.9	**78.5**	77.4	77.9
	OA-Adapter	**78.0**	78.2	**77.9**	**78.0**
Llama-7B	O-LoRA	76.2	76.3	75.7	76.1
	OA-Adapter	**76.4**	**76.8**	**76.7**	**76.6**

Pre-trained Model Analysis. We investigate the performance of OA-Adapter and O-LoRA across varying models (T5-base, T5-large, T5-XL, Llama-7B) using the Standard CL Benchmark. The results, depicted in Table 6, reveal that OA-Adapter's average accuracy consistently improves, which suggests that OA-Adapter effectively leverages the increased representational capacity of larger models. The consistent superiority across different models indicates that OA-Adapter's mechanism provides effective protection against catastrophic forgetting while enabling precise task-specific optimization, regardless of the underlying model architecture's complexity and scales.

Sensitivity to Orthogonality Coefficient. To evaluate the impact of the orthogonality regularization strength on continual learning performance, we conducted a sensitivity analysis on the coefficient λ_{orth} using the T5-Large model across three different task orders on the Standard CL Benchmark. As shown in Table 7, the optimal regularization strength varies by task order (e.g., $\lambda_{\text{orth}} = 5$ for Order 1 and $\lambda_{\text{orth}} = 1$ for Order

Table 7: Effect of the orthogonality coefficient λ_{orth}.

λ_{orth}	Order 1	Order 2	Order 3	Average
0.5	74.3	75.3	74.5	74.7
1	73.4	74.8	**75.1**	74.4
2	73.5	74.9	73.2	73.9
3	74.2	73.9	74.1	74.1
4	74.7	**76.4**	73.6	74.9
5	**75.8**	75.6	74.6	**75.3**

Table 8: Compatibility with replay-based strategies.

Method	AA↑	F.Ra↓	BWT↑
Replay	20.5	14.7	-15.8
SAPT − Replay	11.1	42.8	-40.4
SAPT + Replay	44.2	7.7	-6.79
SAPT + ARM	50.8	1.2	-1.04
OA-Adapter	29.3	8.2	-6.0
OA-Adapter + Replay	48.3	1.2	-0.87

3), reflecting a context-dependent plasticity-stability trade-off. Therefore, unless otherwise specified, the reported main results use the λ_{orth} selected by the best validation AA averaged across orders. This protocol ensures that the regularization strength is adaptively tailored to the specific requirements of the learning trajectory, yielding stable performance across long sequences with distribution shifts.

Compatibility with Replay and Comparison with Replay-based Method SAPT. To further demonstrate the flexibility of OA-Adapter, we investigate its compatibility with replay-based continual learning strategies. Replay-based methods remain a prominent approach for mitigating catastrophic forgetting despite their practical drawbacks. Specifically, we combine OA-Adapter with a simple experience replay buffer and compare the resulting hybrid against SAPT (Zhao et al., 2024), a strong replay-based baseline, on the challenging SuperNI benchmark using the T5-large model. As shown in Table 8, the OA-Adapter+Replay hybrid achieves an average accuracy of 48.3%, substantially outperforming SAPT+Replay (44.2%) while maintaining low forgetting. This consistent gain highlights that OA-Adapter integrates effectively with replay mechanisms, leveraging stored exemplars to further stabilize learning without compromising its adaptive parameter allocation. While the OA-Adapter+Replay hybrid achieves slightly lower performance than the reported SAPT+ARM (50.8%), this gap is expected given that ARM is a specialized replay mechanism tailored specifically to SAPT's attention-based architecture (Zhao et al., 2024). ARM introduces additional complexity and computational overhead through its adversarial memory management and targeted replay strategy, which are tightly coupled with SAPT's design. In contrast, our hybrid employs a simple, generic experience replay buffer that is agnostic to the underlying parameter allocation mechanism. Despite this simplicity, OA-Adapter+Replay delivers highly competitive results, demonstrating near-state-of-the-art performance with significantly lower implementation complexity. Nevertheless, replay-based methods—whether standard experience replay or specialized variants like ARM—incur substantial practical costs, including large-scale storage of task-specific data (raising privacy and memory concerns) or computationally intensive synthetic data generation. These limitations restrict their scalability in real-world, resource-constrained, or privacy-sensitive applications. Given that replay-based mitigation is not the primary focus of this work, and considering the associated overheads, we elected to center our core contributions on a fully replay-free, memory-efficient paradigm that achieves strong performance through adaptive parameter allocation alone. Exploring tighter integration with advanced replay techniques remains a promising direction for future work, where additional resources are available.

5 Conclusion

In this paper, we introduce OA-Adapter, a novel parameter-efficient approach for continual learning (CL) in LLMs that considers both dynamic budget adaptation and orthogonal subspace learning in an end-to-end training stage. Comprehensive experiments demonstrate OA-Adapter's consistent superiority over existing methods across multiple benchmarks while using significantly fewer parameters. The observed heterogeneity in optimal parameter allocation across tasks and layers validates the necessity of our budget-adaptive approach. As the first work to integrate budget adaptation into parameter-efficient fine-tuning for CL in LLMs, OA-Adapter establishes a new paradigm that jointly optimizes parameter budget allocation and knowledge preservation. This advancement paves the way for efficient and effective adaptation of LLMs to evolving real-world.

References

Armen Aghajanyan, Sonal Gupta, and Luke Zettlemoyer. Intrinsic dimensionality explains the effectiveness of language model fine-tuning. In *Proceedings of the 59th annual meeting of the association for computational linguistics and the 11th international joint conference on natural language processing (volume 1: long papers)*, pp. 7319–7328, 2021.

Rahaf Aljundi, Francesca Babiloni, Mohamed Elhoseiny, Marcus Rohrbach, and Tinne Tuytelaars. Memory aware synapses: Learning what (not) to forget. In Vittorio Ferrari, Martial Hebert, Cristian Sminchisescu, and Yair Weiss (eds.), *Computer Vision - ECCV 2018 - 15th European Conference, Munich, Germany, September 8-14, 2018, Proceedings, Part III*, volume 11207 of *Lecture Notes in Computer Science*, pp. 144–161. Springer, 2018. doi: 10.1007/978-3-030-01219-9_9. URL https://doi.org/10.1007/978-3-030-01219-9_9.

Charith Chandra Sai Balne, Sreyoshi Bhaduri, Tamoghna Roy, Vinija Jain, and Aman Chadha. Parameter efficient fine tuning: A comprehensive analysis across applications. *CoRR*, abs/2404.13506, 2024. doi: 10.48550/ARXIV.2404.13506. URL https://doi.org/10.48550/arXiv.2404.13506.

Huandong Chang, Zicheng Ma, Mingyuan Ma, Zhenting Qi, Andrew Sabot, Hong Jiang, and HT Kung. Elalora: Elastic & learnable low-rank adaptation for efficient model fine-tuning. *arXiv preprint arXiv:2504.00254*, 2025.

Arslan Chaudhry, Puneet Kumar Dokania, Thalaiyasingam Ajanthan, and Philip H. S. Torr. Riemannian walk for incremental learning: Understanding forgetting and intransigence. In Vittorio Ferrari, Martial Hebert, Cristian Sminchisescu, and Yair Weiss (eds.), *Computer Vision - ECCV 2018 - 15th European Conference, Munich, Germany, September 8-14, 2018, Proceedings, Part XI*, volume 11215 of *Lecture Notes in Computer Science*, pp. 556–572. Springer, 2018. doi: 10.1007/978-3-030-01252-6_33. URL https://doi.org/10.1007/978-3-030-01252-6_33.

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.

Zezhou Chen, Zhaoxiang Liu, Kai Wang, and Shiguo Lian. Reparameterization-based parameter-efficient fine-tuning methods for large language models: A systematic survey. In Derek F. Wong, Zhongyu Wei, and Muyun Yang (eds.), *Natural Language Processing and Chinese Computing - 13th National CCF Conference, NLPCC 2024, Hangzhou, China, November 1-3, 2024, Proceedings, Part III*, volume 15361 of *Lecture Notes in Computer Science*, pp. 107–118. Springer, 2024. doi: 10.1007/978-981-97-9437-9_9. URL https://doi.org/10.1007/978-981-97-9437-9_9.

Cyprien de Masson d'Autume, Sebastian Ruder, Lingpeng Kong, and Dani Yogatama. Episodic memory in lifelong language learning. In Hanna M. Wallach, Hugo Larochelle, Alina Beygelzimer, Florence d'Alché-Buc, Emily B. Fox, and Roman Garnett (eds.), *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada*, pp. 13122–13131, 2019. URL https://proceedings.neurips.cc/paper/2019/hash/f8d2e80c1458ea2501f98a2cafadb397-Abstract.html.

Ning Ding, Xingtai Lv, Qiaosen Wang, Yulin Chen, Bowen Zhou, Zhiyuan Liu, and Maosong Sun. Sparse low-rank adaptation of pre-trained language models. In Houda Bouamor, Juan Pino, and Kalika Bali (eds.), *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pp. 4133–4145, Singapore, December 2023. Association for Computational Linguistics. doi: 10.18653/v1/2023.emnlp-main.252. URL https://aclanthology.org/2023.emnlp-main.252/.

Mehrdad Farajtabar, Navid Azizan, Alex Mott, and Ang Li. Orthogonal gradient descent for continual learning. In Silvia Chiappa and Roberto Calandra (eds.), *The 23rd International Conference on Artificial Intelligence and Statistics, AISTATS 2020, 26-28 August 2020, Online [Palermo, Sicily, Italy]*, volume 108 of *Proceedings of Machine Learning Research*, pp. 3762–3773. PMLR, 2020. URL http://proceedings.mlr.press/v108/farajtabar20a.html.

Frithjof Gressmann, Zach Eaton-Rosen, and Carlo Luschi. Improving neural network training in low dimensional random bases. *Advances in Neural Information Processing Systems*, 33:12140–12150, 2020.

Yiduo Guo, Wenpeng Hu, Dongyan Zhao, and Bing Liu. Adaptive orthogonal projection for batch and online continual learning. In *Thirty-Sixth AAAI Conference on Artificial Intelligence, AAAI 2022, Thirty-Fourth Conference on Innovative Applications of Artificial Intelligence, IAAI 2022, The Twelveth Symposium on Educational Advances in Artificial Intelligence, EAAI 2022 Virtual Event, February 22 - March 1, 2022*, pp. 6783–6791. AAAI Press, 2022. doi: 10.1609/AAAI.V36I6.20634. URL https://doi.org/10.1609/aaai.v36i6.20634.

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 Kamalika Chaudhuri and Ruslan Salakhutdinov (eds.), *Proceedings of the 36th International Conference on Machine Learning, ICML 2019, 9-15 June 2019, Long Beach, California, USA*, volume 97 of *Proceedings of Machine Learning Research*, pp. 2790–2799. PMLR, 2019. URL http://proceedings.mlr.press/v97/houlsby19a.html.

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. In *The Tenth International Conference on Learning Representations, ICLR 2022, Virtual Event, April 25-29, 2022*. OpenReview.net, 2022. URL https://openreview.net/forum?id=nZeVKeeFYf9.

Joel Jang, Seonghyeon Ye, Sohee Yang, Joongbo Shin, Janghoon Han, Gyeonghun Kim, Stanley Jungkyu Choi, and Minjoon Seo. Towards continual knowledge learning of language models. In *The Tenth International Conference on Learning Representations, ICLR 2022, Virtual Event, April 25-29, 2022*. OpenReview.net, 2022. URL https://openreview.net/forum?id=vfsRB5MImo9.

Tangyu Jiang, Haodi Wang, and Chun Yuan. Diffora: Enabling parameter-efficient LLM fine-tuning via differential low-rank matrix adaptation. *CoRR*, abs/2502.08905, 2025. doi: 10.48550/ARXIV.2502.08905. URL https://doi.org/10.48550/arXiv.2502.08905.

Xisen Jin, Dejiao Zhang, Henghui Zhu, Wei Xiao, Shang-Wen Li, Xiaokai Wei, Andrew O. Arnold, and Xiang Ren. Lifelong pretraining: Continually adapting language models to emerging corpora. In Marine Carpuat, Marie-Catherine de Marneffe, and Iván Vladimir Meza Ruíz (eds.), *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL 2022, Seattle, WA, United States, July 10-15, 2022*, pp. 4764–4780. Association for Computational Linguistics, 2022. doi: 10.18653/V1/2022.NAACL-MAIN.351. URL https://doi.org/10.18653/v1/2022.naacl-main.351.

Zixuan Ke and Bing Liu. Continual learning of natural language processing tasks: A survey. *CoRR*, abs/2211.12701, 2022. doi: 10.48550/ARXIV.2211.12701. URL https://doi.org/10.48550/arXiv.2211.12701.

Chen-An Li and Hung-Yi Lee. Examining forgetting in continual pre-training of aligned large language models. *CoRR*, abs/2401.03129, 2024. doi: 10.48550/ARXIV.2401.03129. URL https://doi.org/10.48550/arXiv.2401.03129.

Chunyuan Li, Heerad Farkhoor, Rosanne Liu, and Jason Yosinski. Measuring the intrinsic dimension of objective landscapes. *arXiv preprint arXiv:1804.08838*, 2018.

Guoliang Lin, Hanlu Chu, and Hanjiang Lai. Towards better plasticity-stability trade-off in incremental learning: A simple linear connector. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2022, New Orleans, LA, USA, June 18-24, 2022*, pp. 89–98. IEEE, 2022. doi: 10.1109/CVPR52688.2022.00019. URL https://doi.org/10.1109/CVPR52688.2022.00019.

Zequan Liu, Jiawen Lyn, Wei Zhu, Xing Tian, and Yvette Graham. ALoRA: Allocating low-rank adaptation for fine-tuning large language models. In Kevin Duh, Helena Gomez, and Steven Bethard (eds.), *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational*

Linguistics: Human Language Technologies (Volume 1: Long Papers), pp. 622–641, Mexico City, Mexico, June 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.naacl-long.35. URL https://aclanthology.org/2024.naacl-long.35/.

Andrew L. Maas, Raymond E. Daly, Peter T. Pham, Dan Huang, Andrew Y. Ng, and Christopher Potts. Learning word vectors for sentiment analysis. In Dekang Lin, Yuji Matsumoto, and Rada Mihalcea (eds.), *The 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies, Proceedings of the Conference, 19-24 June, 2011, Portland, Oregon, USA*, pp. 142–150. The Association for Computer Linguistics, 2011. URL https://aclanthology.org/P11-1015/.

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.

Chengwei Qin and Shafiq Joty. Lfpt5: A unified framework for lifelong few-shot language learning based on prompt tuning of t5. *arXiv preprint arXiv:2110.07298*, 2021.

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. *J. Mach. Learn. Res.*, 21:140:1–140:67, 2020. URL https://jmlr.org/papers/v21/20-074.html.

Dushyant Rao, Francesco Visin, Andrei A. Rusu, Razvan Pascanu, Yee Whye Teh, and Raia Hadsell. Continual unsupervised representation learning. In Hanna M. Wallach, Hugo Larochelle, Alina Beygelzimer, Florence d'Alché-Buc, Emily B. Fox, and Roman Garnett (eds.), *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada*, pp. 7645–7655, 2019. URL https://proceedings.neurips.cc/paper/2019/hash/861578d797aeb0634f77aff3f488cca2-Abstract.html.

Anastasia Razdaibiedina, Yuning Mao, Rui Hou, Madian Khabsa, Mike Lewis, and Amjad Almahairi. Progressive prompts: Continual learning for language models. In *The Eleventh International Conference on Learning Representations, ICLR 2023, Kigali, Rwanda, May 1-5, 2023*. OpenReview.net, 2023. URL https://openreview.net/forum?id=UJTgQBc91_.

Sylvestre-Alvise Rebuffi, Alexander Kolesnikov, Georg Sperl, and Christoph H. Lampert. icarl: Incremental classifier and representation learning. In *2017 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017, Honolulu, HI, USA, July 21-26, 2017*, pp. 5533–5542. IEEE Computer Society, 2017. doi: 10.1109/CVPR.2017.587. URL https://doi.org/10.1109/CVPR.2017.587.

Subendhu Rongali, Abhyuday Jagannatha, Bhanu Pratap Singh Rawat, and Hong Yu. Continual domain-tuning for pretrained language models. *arXiv preprint arXiv:2004.02288*, 2020.

Jonathan Schwarz, Wojciech Czarnecki, Jelena Luketina, Agnieszka Grabska-Barwinska, Yee Whye Teh, Razvan Pascanu, and Raia Hadsell. Progress & compress: A scalable framework for continual learning. In Jennifer G. Dy and Andreas Krause (eds.), *Proceedings of the 35th International Conference on Machine Learning, ICML 2018, Stockholmsmässan, Stockholm, Sweden, July 10-15, 2018*, volume 80 of *Proceedings of Machine Learning Research*, pp. 4535–4544. PMLR, 2018. URL http://proceedings.mlr.press/v80/schwarz18a.html.

Haizhou Shi and Hao Wang. A unified approach to domain incremental learning with memory: Theory and algorithm. In Alice Oh, Tristan Naumann, Amir Globerson, Kate Saenko, Moritz Hardt, and Sergey Levine (eds.), *Advances in Neural Information Processing Systems 36: Annual Conference on Neural Information Processing Systems 2023, NeurIPS 2023, New Orleans, LA, USA, December 10 - 16, 2023*, 2023. URL http://papers.nips.cc/paper_files/paper/2023/hash/30d046e94d7b8037d6ef27c4357a8dd4-Abstract-Conference.html.

Haizhou Shi, Zihao Xu, Hengyi Wang, Weiyi Qin, Wenyuan Wang, Yibin Wang, and Hao Wang. Continual learning of large language models: A comprehensive survey. *CoRR*, abs/2404.16789, 2024. doi: 10.48550/ARXIV.2404.16789. URL https://doi.org/10.48550/arXiv.2404.16789.

Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, Aurélien Rodriguez, Armand Joulin, Edouard Grave, and Guillaume Lample. Llama: Open and efficient foundation language models. *CoRR*, abs/2302.13971, 2023. doi: 10.48550/ARXIV.2302.13971. URL https://doi.org/10.48550/arXiv.2302.13971.

Alex Wang, Yada Pruksachatkun, Nikita Nangia, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel Bowman. Superglue: A stickier benchmark for general-purpose language understanding systems. *Advances in neural information processing systems*, 32, 2019a.

Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R. Bowman. GLUE: A multi-task benchmark and analysis platform for natural language understanding. In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net, 2019b. URL https://openreview.net/forum?id=rJ4km2R5t7.

Liyuan Wang, Xingxing Zhang, Hang Su, and Jun Zhu. A comprehensive survey of continual learning: Theory, method and application. *IEEE Trans. Pattern Anal. Mach. Intell.*, 46(8):5362–5383, 2024. doi: 10.1109/TPAMI.2024.3367329. URL https://doi.org/10.1109/TPAMI.2024.3367329.

Xiao Wang, Tianze Chen, Qiming Ge, Han Xia, Rong Bao, Rui Zheng, Qi Zhang, Tao Gui, and Xuanjing Huang. Orthogonal subspace learning for language model continual learning. In Houda Bouamor, Juan Pino, and Kalika Bali (eds.), *Findings of the Association for Computational Linguistics: EMNLP 2023, Singapore, December 6-10, 2023*, pp. 10658–10671. Association for Computational Linguistics, 2023a. doi: 10.18653/V1/2023.FINDINGS-EMNLP.715. URL https://doi.org/10.18653/v1/2023.findings-emnlp.715.

Xiao Wang, Yuansen Zhang, Tianze Chen, Songyang Gao, Senjie Jin, Xianjun Yang, Zhiheng Xi, Rui Zheng, Yicheng Zou, Tao Gui, Qi Zhang, and Xuanjing Huang. TRACE: A comprehensive benchmark for continual learning in large language models. *CoRR*, abs/2310.06762, 2023b. doi: 10.48550/ARXIV.2310.06762. URL https://doi.org/10.48550/arXiv.2310.06762.

Yizhong Wang, Swaroop Mishra, Pegah Alipoormolabashi, Yeganeh Kordi, Amirreza Mirzaei, Atharva Naik, Arjun Ashok, Arut Selvan Dhanasekaran, Anjana Arunkumar, David Stap, Eshaan Pathak, Giannis Karamanolakis, Haizhi Gary Lai, Ishan Purohit, Ishani Mondal, Jacob Anderson, Kirby Kuznia, Krima Doshi, Kuntal Kumar Pal, Maitreya Patel, Mehrad Moradshahi, Mihir Parmar, Mirali Purohit, Neeraj Varshney, Phani Rohitha Kaza, Pulkit Verma, Ravsehaj Singh Puri, Rushang Karia, Savan Doshi, Shailaja Keyur Sampat, Siddhartha Mishra, Sujan Reddy A, Sumanta Patro, Tanay Dixit, and Xudong Shen. Super-naturalinstructions: Generalization via declarative instructions on 1600+ NLP tasks. In Yoav Goldberg, Zornitsa Kozareva, and Yue Zhang (eds.), *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing, EMNLP 2022, Abu Dhabi, United Arab Emirates, December 7-11, 2022*, pp. 5085–5109. Association for Computational Linguistics, 2022a. doi: 10.18653/V1/2022.EMNLP-MAIN.340. URL https://doi.org/10.18653/v1/2022.emnlp-main.340.

Zifeng Wang, Zizhao Zhang, Chen-Yu Lee, Han Zhang, Ruoxi Sun, Xiaoqi Ren, Guolong Su, Vincent Perot, Jennifer G. Dy, and Tomas Pfister. Learning to prompt for continual learning. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2022, New Orleans, LA, USA, June 18-24, 2022*, pp. 139–149. IEEE, 2022b. doi: 10.1109/CVPR52688.2022.00024. URL https://doi.org/10.1109/CVPR52688.2022.00024.

Zhiheng Xi, Wenxiang Chen, Xin Guo, Wei He, Yiwen Ding, Boyang Hong, Ming Zhang, Junzhe Wang, Senjie Jin, Enyu Zhou, Rui Zheng, Xiaoran Fan, Xiao Wang, Limao Xiong, Yuhao Zhou, Weiran Wang, Changhao Jiang, Yicheng Zou, Xiangyang Liu, Zhangyue Yin, Shihan Dou, Rongxiang Weng, Wenjuan Qin, Yongyan Zheng, Xipeng Qiu, Xuanjing Huang, Qi Zhang, and Tao Gui. The rise and potential of large language model based agents: a survey. *Sci. China Inf. Sci.*, 68(2), 2025. doi: 10.1007/S11432-024-4222-0. URL https://doi.org/10.1007/s11432-024-4222-0.

Yongyu Yan, Kui Xue, Xiaoming Shi, Qi Ye, Jingping Liu, and Tong Ruan. AF adapter: Continual pretraining for building chinese biomedical language model. In Xingpeng Jiang, Haiying Wang, Reda Alhajj, Xiaohua Hu, Felix Engel, Mufti Mahmud, Nadia Pisanti, Xuefeng Cui, and Hong Song (eds.), *IEEE International Conference on Bioinformatics and Biomedicine, BIBM 2023, Istanbul, Turkiye, December 5-8, 2023*, pp. 953–957. IEEE, 2023. doi: 10.1109/BIBM58861.2023.10385733. URL https://doi.org/10.1109/BIBM58861.2023.10385733.

Qingru Zhang, Minshuo Chen, Alexander Bukharin, Nikos Karampatziakis, Pengcheng He, Yu Cheng, Weizhu Chen, and Tuo Zhao. Adalora: Adaptive budget allocation for parameter-efficient fine-tuning, 2023. URL https://arxiv.org/abs/2303.10512.

Xiang Zhang, Junbo Jake Zhao, and Yann LeCun. Character-level convolutional networks for text classification. In Corinna Cortes, Neil D. Lawrence, Daniel D. Lee, Masashi Sugiyama, and Roman Garnett (eds.), *Advances in Neural Information Processing Systems 28: Annual Conference on Neural Information Processing Systems 2015, December 7-12, 2015, Montreal, Quebec, Canada*, pp. 649–657, 2015. URL https://proceedings.neurips.cc/paper/2015/hash/250cf8b51c773f3f8dc8b4be867a9a02-Abstract.html.

Weixiang Zhao, Shilong Wang, Yulin Hu, Yanyan Zhao, Bing Qin, Xuanyu Zhang, Qing Yang, Dongliang Xu, and Wanxiang Che. SAPT: A shared attention framework for parameter-efficient continual learning of large language models. In Lun-Wei Ku, Andre Martins, and Vivek Srikumar (eds.), *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), ACL 2024, Bangkok, Thailand, August 11-16, 2024*, pp. 11641–11661. Association for Computational Linguistics, 2024. doi: 10.18653/V1/2024.ACL-LONG.625. URL https://doi.org/10.18653/v1/2024.acl-long.625.

Da-Wei Zhou, Hai-Long Sun, Jingyi Ning, Han-Jia Ye, and De-Chuan Zhan. Continual learning with pre-trained models: A survey. In *Proceedings of the Thirty-Third International Joint Conference on Artificial Intelligence, IJCAI 2024, Jeju, South Korea, August 3-9, 2024*, pp. 8363–8371. ijcai.org, 2024a. URL https://www.ijcai.org/proceedings/2024/924.

Xiongtao Zhou, Jie He, Yuhua Ke, Guangyao Zhu, Víctor Gutiérrez-Basulto, and Jeff Z. Pan. An empirical study on parameter-efficient fine-tuning for multimodal large language models. In Lun-Wei Ku, Andre Martins, and Vivek Srikumar (eds.), *Findings of the Association for Computational Linguistics, ACL 2024, Bangkok, Thailand and virtual meeting, August 11-16, 2024*, pp. 10057–10084. Association for Computational Linguistics, 2024b. doi: 10.18653/V1/2024.FINDINGS-ACL.598. URL https://doi.org/10.18653/v1/2024.findings-acl.598.

A Appendix

A.1 LLM Usage Declaration

Large language models (e.g., ChatGPT) were used solely for language editing and formatting. They did not contribute to the conception, design, implementation, analysis, data generation or labeling, or evaluation of the methods and results. All technical content and claims were authored and verified by the authors, and no personal, proprietary, or sensitive data were shared with LLM services.

A.2 Datasets and Task Orders.

Table 9 and 10 show details of the datasets we used for our experiments, along with their evaluation metrics. Overall, we used datasets from the Standard CL benchmark Zhang et al. (2015), GLUE Wang et al. (2019b), and SuperGLUE Wang et al. (2019a) benchmarks, and added the IMDB movie reviews dataset Maas et al. (2011), following Razdaibiedina et al. (2023). Additionally, we incorporated the SuperNI Benchmark Wang et al. (2022a), a benchmark of diverse NLP tasks with expert-written instructions, for more rigorous evaluation of our model. Furthermore, Table 11 shows details of task orders used in our CL experiments.

Table 9: The details of 15 classification datasets in the Long Sequence Benchmark. NLI denotes natural language inference, QA denotes questions and answers task. First five tasks correspond to the Standard CL Benchmark, all other tasks are used in long-sequence experiments.

Dataset	**Category**	**Task**	**Domain**	**Metric**
Yelp	CL Benchmark	Sentiment Analysis	Yelp Reviews	Accuracy
Amazon	CL Benchmark	Sentiment Analysis	Amazon Reviews	Accuracy
DBPedia	CL Benchmark	Topic Classification	Wikipedia	Accuracy
Yahoo	CL Benchmark	Topic Classification	Yahoo Q&A	Accuracy
AG News	CL Benchmark	Topic Classification	News	Accuracy
MNLI	GLUE	NLI	Various	Accuracy
QQP	GLUE	Paraphrase Detection	Quora	Accuracy
RTE	GLUE	NLI	News, Wikipedia	Accuracy
SST-2	GLUE	Sentiment Analysis	Movie Reviews	Accuracy
WIC	SuperGLUE	Word Sense Disambiguation	Lexical Databases	Accuracy
CB	SuperGLUE	NLI	Various	Accuracy
COPA	SuperGLUE	QA	Blogs, Encyclopedia	Accuracy
BoolQA	SuperGLUE	Boolean QA	Wikipedia	Accuracy
MultiRC	SuperGLUE	QA	Various	Accuracy
IMDB	SuperGLUE	Sentiment Analysis	Movie Reviews	Accuracy

Table 10: The details of 15 datasets in the SuperNI Benchmark.

Dataset name	**Task**	**Metric**
task639_multi_woz_user_utterance_generation	dialogue generation	Rouge-L
task1590_diplomacy_text_generation	dialogue generation	Rouge-L
task1729_personachat_generate_next	dialogue generation	Rouge-L
task181_outcome_extraction	information extraction	Rouge-L
task748_glucose_reverse_cause_event_detection	information extraction	Rouge-L
task1510_evalution_relation_extraction	information extraction	Rouge-L
task002_quoref_answer_generation	question answering	Rouge-L
task073_commonsenseqa_answer_generation	question answering	Rouge-L
task591_sciq_answer_generation	question answering	Rouge-L
task511_reddit_tifu_long_text_summarization	summarization	Rouge-L
task1290_xsum_summarization	summarization	Rouge-L
task1572_samsum_summary	summarization	Rouge-L
task363_sst2_polarity_classification	sentiment analysis	accuracy
task875_emotion_classification	sentiment analysis	accuracy
task1687_sentiment140_classification	sentiment analysis	accuracy

Table 11: Eight different task sequence orders utilized in our experiments. Orders 1-3 follow the Standard CL Benchmark as established by previous research, focusing on a more traditional task sequence. Orders 4-6 are customized for long-sequence experimentation, encompassing 15 tasks each and are structured according to the methodologies outlined in Razdaibiedina et al. (2023). Orders 7-8 correspond to the SuperNI benchmark.

Benchmarks	Order	Task Sequence
Standard CL Benchmark	1	dbpedia → amazon → yahoo → ag
	2	dbpedia → amazon → ag → yahoo
	3	yahoo → amazon → ag → dbpedia
Long Sequence Benchmark	4	mnli → cb → wic → copa → qqp → boolqa → rte → imdb → yelp → amazon → sst-2 → dbpedia → ag → multirc → yahoo
	5	multirc → boolqa → wic → mnli → cb → copa → qqp → rte → imdb → sst-2 → yelp → amazon → ag → dbpedia → yahoo
	6	yelp → amazon → mnli → cb → copa → qqp → rte → imdb → sst-2 → dbpedia → ag → yahoo → multirc → boolqa → wic
SuperNI Benchmark	7	task1572 → task363 → task1290 → task181 → task002 → task1510 → task639 → task1729 → task073 → task1590 → task748 → task511 → task591 → task1687 → task875
	8	task748 → task073 → task1590 → task639 → task1572 → task1687 → task591 → task363 → task1510 → task1729 → task181 → task511 → task002 → task1290 → task875

A.3 Task Instructions

Consistent with the experimental settings of previous works (Wang et al., 2023a; Zhao et al., 2024), we used instruction tuning in the Stand CL Benchmark and Long Sequence Benchmark experiments. Detailed instructions used for instruction tuning across different tasks are provided in Table 12. This approach offers dual advantages: it incorporates human expertise for efficient learning while enabling models to better capture underlying principles through explicit guidance, thereby enhancing generalization capabilities. The consistent instruction-based framework allows for direct performance comparisons while leveraging the benefits of natural language supervision.

Table 12: Instructions for different tasks used in our experiments.

Task	Prompts
NLI	What is the logical relationship between the "sentence 1" and the "sentence 2"? Choose one from the option.
QQP	Whether the "first sentence" and the "second sentence" have the same meaning? Choose one from the option.
SC	What is the sentiment of the following paragraph? Choose one from the option.
TC	What is the topic of the following paragraph? Choose one from the option.
BoolQA	According to the following passage, is the question true or false? Choose one from the option.
MultiRC	According to the following passage and question, is the candidate answer true or false? Choose one from the option.
WiC	Given a word and two sentences, whether the word is used with the same sense in both sentence? Choose one from the option.

A.4 Heterogeneous Budget Requirements Across Tasks and Layers.

As discussed in Section 4.2, we extend our analysis to investigate budget allocation in the context of continual learning. Here, we further present results under other task orders, as illustrated in Figure 4 and 5. These findings corroborate our analysis in Section 4.2: (a) The relationship between performance and parameter budget does not follow constant rules but rather necessitates case-specific consideration. (b) Within continual learning scenarios, the first task primarily focuses on acquiring capabilities built upon the pretrained model, whereas subsequent tasks must additionally preserve knowledge from preceding tasks, thus requiring more nuanced fine-tuning. This further substantiates our analysis that different tasks in the CL scenarios require varying budgets, and that allocating budgets according to training sequence and task characteristics is both necessary and justified.

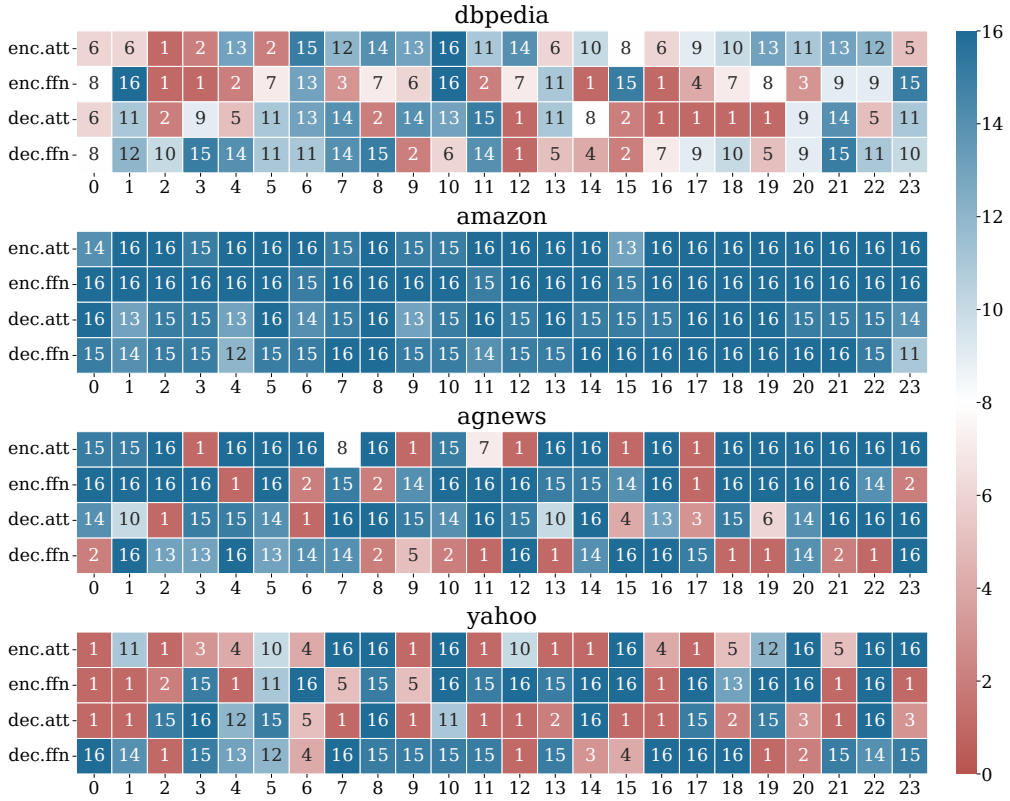

Figure 4: Final dimensions after sequential training following Order-2 with OA-Adapter.

A.5 Occurrence and Mitigation of Catastrophic forgetting.

To further validate the effectiveness and consistency of orthogonal parameter subspace constraints, we conduct sequential training following Order-2 and Order-3, with results illustrated in Figure 6 and 7, respectively. Consistent with the results reported in Section 4.3, we observe severe catastrophic forgetting in the absence of orthogonal constraints, especially for earlier tasks. In contrast, models trained with orthogonal parameter subspace constraints are able to preserve performance across all tasks to a much greater extent. Notably, although the specific tasks affected most by forgetting vary depending on the task order, the general trend holds: orthogonal constraints provide consistent mitigation of forgetting regardless of task permutation. These results reinforce our earlier findings and highlight the robustness of orthogonal subspace regularization as a general mechanism for alleviating forgetting in continual learning scenarios.

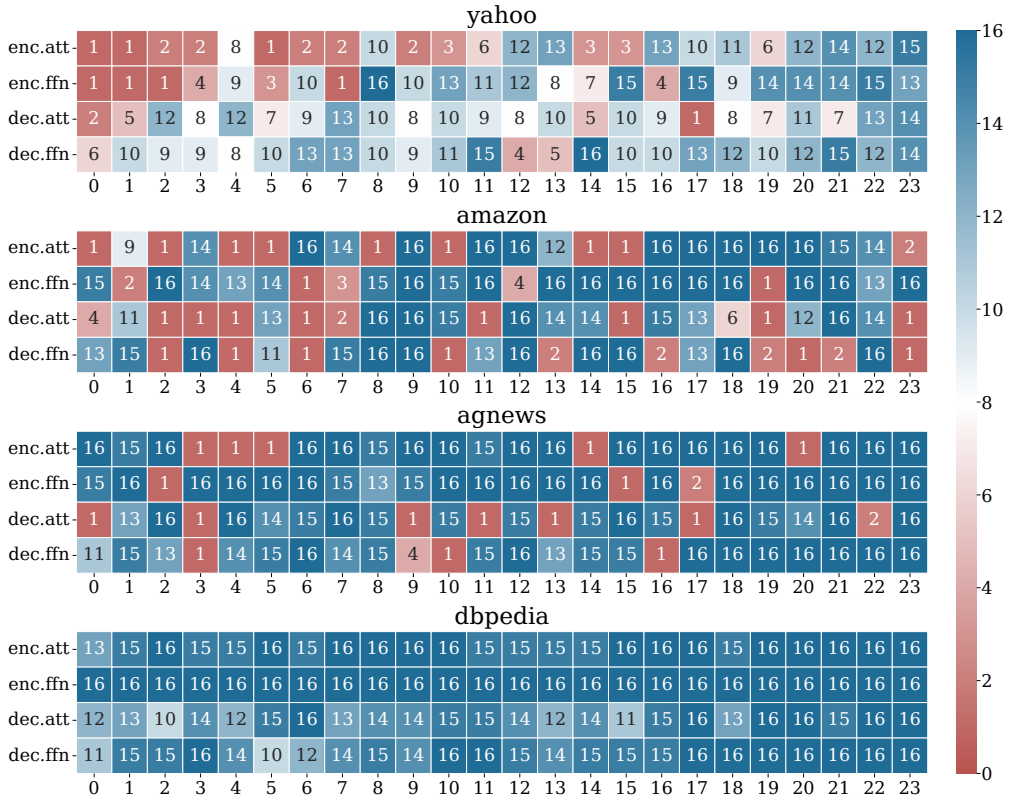

Figure 5: Final dimensions after sequential training following Order-3 with OA-Adapter.

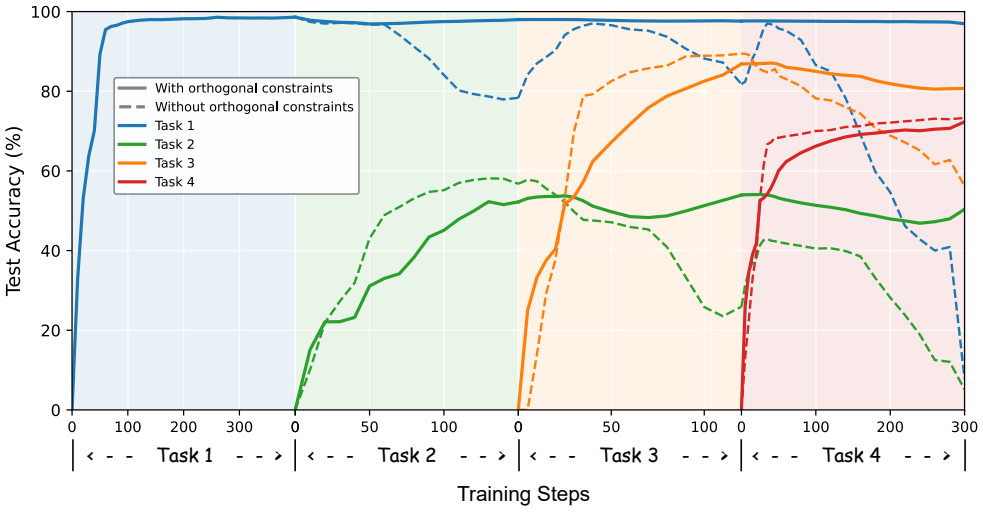

Figure 6: Occurrence and mitigation of catastrophic forgetting during sequential training following Order-2 across multiple tasks.

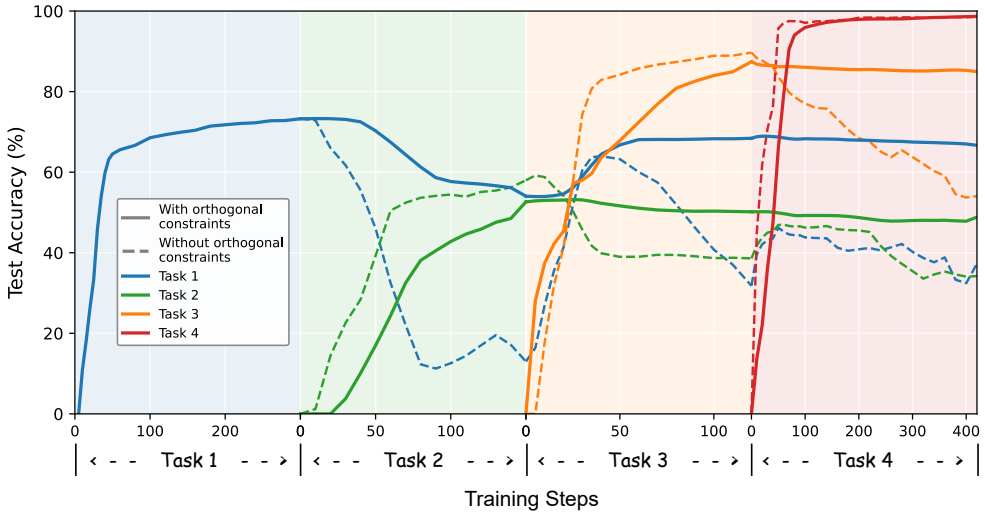

Figure 7: Occurrence and mitigation of catastrophic forgetting during sequential training following Order-3 across multiple tasks.

A.6 Compute and Memory Cost Analysis

Training cost and GPU memory. We benchmark the training cost of OA-Adapter against O-LoRA under Task Order 1 on the Standard CL Benchmark. OA-Adapter demonstrated slightly lower GPU memory usage, averaging 68.9% compared to 70.4% for O-LoRA. OA-Adapter achieved a significantly faster average training time per step: 0.357 seconds versus 0.582 seconds for O-LoRA, suggesting a clear advantage in training efficiency.

Overhead from orthogonality regularization. We further isolate the per-step time consumed by orthogonality computations for OA-Adapter (bottleneck dimension 24) on the Standard CL Benchmark, as shown in Table 13. The per-step overhead grows with the number of previously seen tasks, resulting in an average of 0.0179 s per step across the first four tasks, i.e., a 4.6% relative overhead compared to the total per-step time of 0.357 s.

Table 13: Per-step time spent on orthogonality computations for OA-Adapter.

Task	1	2	3	4	avg
Time (s)	0.0061	0.0130	0.0199	0.0269	0.0179