Spectral Disentanglement: Rank-Aware Task Adaptation for Rehearsal-free Continual Learning in LLMs

Huanxuan Liao[1,2], **Shizhu He**[1,2,3*], **Yupu Hao**[1,2], **Yequan Wang**[3*], **Wenhao Teng**[4], **Xiangwen Liao**[5], **Jun Zhao**[1,2], **Kang Liu**[1,2]

[1] The Key Laboratory of Cognition and Decision Intelligence for Complex Systems, Institute of Automation, Chinese Academy of Sciences, Beijing, China

[2] School of Artificial Intelligence, University of Chinese Academy of Sciences

[3] Beijing Academy of Artificial Intelligence, Beijing, China

[4] Department of Gastrointestinal Surgery Fujian Provincial Cancer Hospital

[5] College of Computer and Data Science, Fuzhou University

{liaohuanxuan2023, haoyupu2023}@ia.ac.cn {shizhu.he, jzhao, kliu}@nlpr.ia.ac.cn

Abstract

Continual Learning (CL) for Large Language Models (LLMs) faces a fundamental **Stability-Plasticity Dilemma**: balancing the plasticity to acquire new capabilities with the stability to preserve prior knowledge. While Parameter-Efficient Fine-Tuning methods, such as LoRA, enable efficient adaptation, we identify a critical flaw in current approaches termed **Rank-Blindness**: the enforcement of a single rank constraint across diverse tasks, which entangles task-shared and task-specific knowledge, leading to catastrophic forgetting of earlier tasks and underfitting on complex new ones. To address this, we propose SPARTA, a novel rehearsal-free framework guided by a rank-spectrum perspective that explicitly disentangles knowledge into two orthogonal subspaces. Specifically, SPARTA employs a low-rank branch to capture task-shared representations and a high-rank branch to model task-specific features. To integrate these complementary representations, we introduce a context-aware dynamic router that adaptively fuses the two branches based on input semantics, while an explicit orthogonality constraint minimizes interference between shared and specific parameter subspaces. This design effectively isolates task-specific updates from shared knowledge, preventing the overwriting of prior capabilities while preserving strong adaptation capacity. Extensive experiments demonstrate that SPARTA achieves a superior stability-plasticity balance compared to single-rank baselines. Notably, the proposed spectral disentanglement strategy substantially reduces inter-task interference and yields strong zero-shot generalization on unseen tasks. Our code will be available at https://github.com/Xnhyacinth/SpaRTA.

*Corresponding authors.

1 Introduction

As Large Language Models (LLMs) (Dubey et al., 2024; Yang et al., 2024) are increasingly deployed in dynamic, open-world environments, the capacity for Continual Learning (CL)—acquiring new capabilities continuously without erasing prior ones—has become paramount (Wang et al., 2023a; Liu et al., 2025). Unlike traditional supervised learning (Roziere et al., 2023), CL necessitates that models sequentially adapt to evolving tasks while maintaining proficiency on historical ones (Zhai et al., 2023; Wu et al., 2024). This requirement engenders a fundamental conflict known as the **Stability-Plasticity Dilemma**: the model must be sufficiently plastic to assimilate new, specific knowledge (Dohare et al., 2021), yet stable enough to preserve generalized linguistic structures.

As illustrated in Figure 1 (b), CL faces an inherent dilemma: rigid prioritization of stability constrains new learning (plasticity), while aggressive plasticity compromises memory retention (stability). While Rehearsal-based methods (Wang et al., 2024b; Sun et al., 2019) mitigate forgetting by replaying history, they fundamentally violate privacy protocols and incur prohibitive storage costs, rendering Rehearsal-Free approaches imperative for LLMs. Recent advances in Parameter-Efficient Fine-Tuning (PEFT), particularly LoRA (Hu et al., 2022), have shown promising by freezing the backbone (Wang et al., 2023b). However, current solutions remain limited. As summarized in Figure 1 (a), beyond the privacy risks of rehearsal and the inference inefficiency of architecture-based expansions (Wang et al., 2024a), existing PEFT methods predominantly suffer from ***Rank-Blindness***. They typically enforce a static, uniform rank constraint (e.g., $r = 8$) across all tasks. This implicitly assuming that all knowledge, whether a broad linguistic

Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), pages 7360–7376

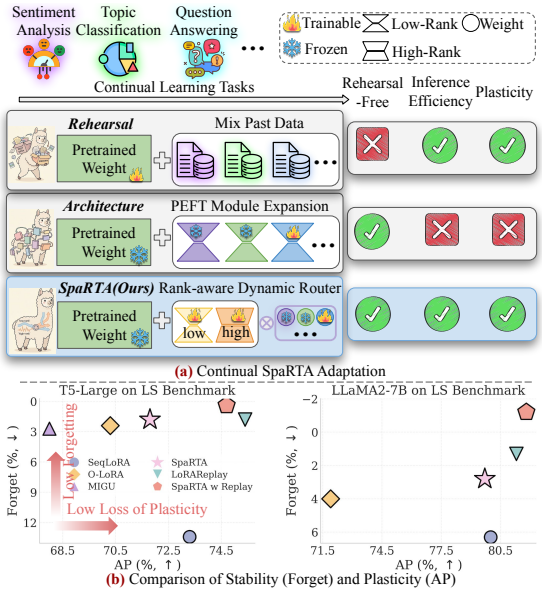


Figure 1: **Paradigms of Continual Learning and the Spectral Sweet Spot.** (a) Evolution of CL paradigms: Unlike Rehearsal (memory-heavy) or Architecture-based expansion (inference-inefficient), our proposed SPARTA employs a Rank-Aware Disentanglement strategy. It utilizes dual-rank components to physically separate shared (low-rank) and specific (high-rank) knowledge without replaying data. (b) Performance trade-off on T5 and LLaMA2. While baselines struggle to balance **Stability (Low Forget)** and **Plasticity (High AP)**, SPARTA breaks this dilemma, achieving the optimal balance in the top-right corner (Sec. 3 for metrics).

rule or a specific domain fact, is uniformly compressible into the same low-rank subspace. Consequently, constraining task-specific adaptation to a low rank often leads to underfitting (*plasticity loss*), while globally increasing the rank disrupts shared parameters, will accelerate catastrophic forgetting (*stability loss*) (Wang et al., 2022).

To address this, we advance a **rank-spectrum perspective** on knowledge representation in LLMs. Drawing from the properties of Singular Value Decomposition (Zhang, 2015), we hypothesize that neural network updates are *spectrally stratified*: Generalizable, task-invariant knowledge (e.g., syntax, reasoning patterns) typically resides in the dominant, low-rank principal components, whereas task-specific, idiosyncratic knowledge (e.g., domain entities, rote facts) requires high-rank updates to capture fine-grained variations (Liao et al., 2025b). Existing methods fail because they entangle these distinct knowledge types into a single subspace. As a result, *spectral interference* will be caused where new specific facts overwrite old general structures.

Motivated by this, we introduce **SPARTA** (**Sp**ectrum-**a**ware **R**ank **T**ask **A**daptation), a novel framework designed to *structurally disentangle* shared and specific knowledge in parameters. Unlike previous mixtures of adapters (Zhao et al., 2024), SPARTA explicitly constructs dual orthogonal subspaces: 1) A **Low-Rank Subspace** (Shared Branch) dedicated to consolidating universal representations that are robust to forgetting. 2) A **High-Rank Subspace** (Specific Branch) dedicated to high-fidelity adaptation for distinct task distributions. Crucially, we introduce a **Spectrum-Aware Dynamic Router** (formerly *decomposed weighting*) that learns to direct input tokens to the appropriate subspace based on their semantic context. Specifically, to ensure true disentanglement, we enforce an orthogonality constraint that minimizes the projection overlap between these subspaces, effectively *routing* common patterns to the stable low-rank component and specific details to the plastic high-rank component. Furthermore, we employ stochastic restoration to protect source knowledge (Dohare et al., 2024; Liu et al., 2023).

We conduct extensive experiments to evaluate SPARTA on the Standard CL Benchmark (Zhang et al., 2015), Long Sequence Benchmark (Razdaibiedina et al., 2023), and TRACE (Wang et al., 2023c) using T5 (Raffel et al., 2019), LLaMA2 (Touvron et al., 2023), LLaMA3.1 (Dubey et al., 2024), and Qwen2.5 (Yang et al., 2024). SPARTA achieves better performance on both public benchmarks and in zero-shot generalization to unseen tasks, validating the effectiveness of spectral disentanglement in mitigating catastrophic forgetting. In summary, our contributions are as follows:

- We formulate the CL challenge through a spectral lens, identifying the **rank-blindness** of existing adapters and proposing rank-based knowledge disentanglement as a fundamental solution.
- We propose SPARTA, a rehearsal-free framework that combines dual-rank adapters with context-aware routing and orthogonal regularization to balance stability and plasticity dynamically.
- Extensive experiments on public benchmarks across diverse LLMs demonstrate that SPARTA consistently outperforms baselines. Notably, SPARTA yields superior zero-shot generalization on unseen tasks, confirming the effective preservation of general capabilities.

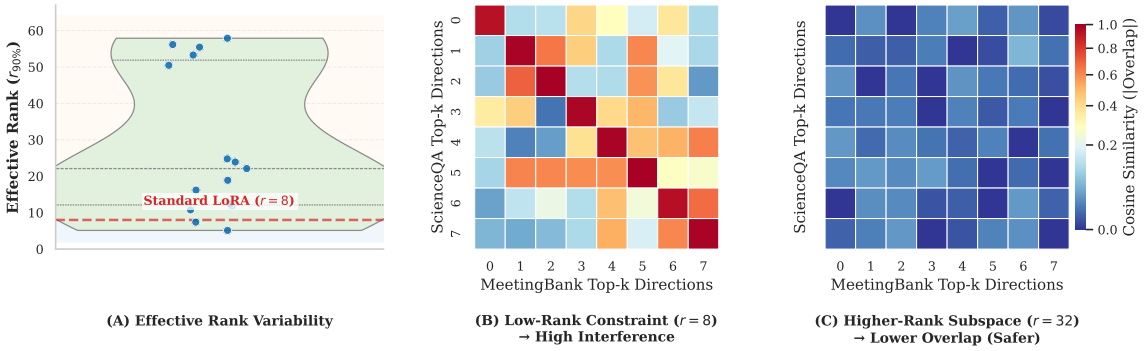


Figure 2: **Empirical Analysis of Rank Limitations and Subspace Interference.**(A) **Heterogeneity of Intrinsic Dimensionality:** The effective rank ($r_{90\%}$) varies significantly across 15 tasks. The standard setting ($r = 8$, red line) creates a severe bottleneck for knowledge-intensive tasks.(B) **Subspace Collapse ($r = 8$):** High cosine similarity (red diagonal) between ScienceQA and MeetingBank updates reveals that constrained ranks force distinct tasks to compete for identical optimization directions.(C) **Orthogonal Separation ($r = 32$):** With relaxed constraints, update subspaces naturally decouple (blue), demonstrating that interference stems from aggressive rank compression.

2 Motivation: Rank-Blindness and Subspace Interference

Current Parameter-Efficient Fine-Tuning (PEFT) methods (Houlsby et al., 2019) predominantly impose a static and uniform low-rank constraint across tasks, as exemplified by LoRA (Hu et al., 2022), which fixes the adaptation rank (e.g., $r = 8$). This design implicitly assumes that the intrinsic dimensionality of adaptation is universally low and task-invariant. In this work, we revisit this assumption through a systematic empirical analysis of task-specific weight updates (ΔW), revealing substantial variability in their effective rank across tasks.

The Heterogeneity of Effective Rank. We first examined the spectral properties of task-specific updates across 15 diverse tasks (refer to Sec. B). By performing Singular Value Decomposition (Zhang, 2015) on the update matrices, we calculated the **Effective Rank** ($r_{90\%}$) needed to capture 90% of the spectral energy. As shown in Figure 2 (A), the results reveal a significant disparity: **Task-Dependent Complexity:** The required rank spans a wide spectrum, from $r \approx 2$ for simple style alignment to $r > 60$ for complex reasoning tasks. **The Information Bottleneck:** The widely used setting of $r = 8$ (red dashed line) falls well below the requirement for the majority of tasks. This suggests that static low-rank adapters impose a severe *compression loss*, preventing the model from fully encoding the necessary knowledge for complex tasks (limiting Plasticity) (Zhao et al., 2024).

Rank-Induced Subspace Interference. Does this aggressive compression come at a cost to stability? We hypothesize that constraining the rank artificially forces the optimization trajectories of distinct tasks to collide in a crowded subspace.To verify this, we analyzed two semantically distinct tasks: **ScienceQA** (reasoning) (Lu et al., 2022) and **MeetingBank** (summarization) (Hu et al., 2023). We extracted the principal directions (top singular vectors) of their respective ΔW and computed the pairwise cosine similarity. **Forced Collision at Low Rank ($r = 8$):** As visualized in Figure 2 (B), the heatmap exhibits a strong diagonal alignment. This indicates that due to the scarcity of available dimensions, the update vector for MeetingBank is forced to align with the primary directions of ScienceQA. Mathematically, this high cosine similarity implies that new learning directly overwrites the parameters critical for the old task, mechanistically explaining catastrophic forgetting. **Emergent Orthogonality at Higher Rank ($r = 32$):** Conversely, when the rank capacity is relaxed to $r = 32$ (Figure 2 (C)), the subspace overlap vanishes (predominantly blue). This demonstrates that *interference is not inherent to the tasks themselves, but is an artifact of rank-blind compression*. With sufficient degrees of freedom, the model naturally finds orthogonal paths to accommodate new skills without disrupting historical knowledge.

These findings expose a structural dilemma: low-rank adaptation induces underfitting and subspace interference, whereas increasing rank conflicts with PEFT efficiency. SPARTA addresses this by a dual-branch design that disentangles shared low-rank representations from task-specific high-rank orthogonality. Further analysis in Appendix C.1

shows that the low-rank branch consolidates shared feature representations, while the high-rank branch captures task-specific variations. t-SNE visualizations and $\mathcal{H}$-divergence measurements further confirm that SPARTA effectively disentangles task-invariant stability from task-specific plasticity.

3 Methodology

Guided by the *Spectral Hypothesis* (Sec. 2), which posits that knowledge is stratified into low-rank general patterns and high-rank specific nuances, we introduce **SPARTA**. This rehearsal-free framework structurally disentangles knowledge acquisition into spectrally distinct subspaces. As shown in Figure 3, SPARTA is built upon three pillars: (1) A **Dual-Branch Spectral Architecture** (§3.2) that physically isolates shared structures from specific mappings; (2) A **Spectrum-Aware Dynamic Router** (§3.3) that acts as a hyper-network to contextually orchestrate information flow; (3) A **Progressive Isolation & Regularization** strategy (§3.4, §3.5) that ensures geometric separation and prevents catastrophic interference over time.

3.1 Preliminary

Formulation. We focus on the Rehearsal-Free Continual Learning (CL) setting for LLMs. Consider a stream of $\mathcal{N}$ sequential tasks $\mathcal{S} = \{\mathcal{T}_1, \ldots, \mathcal{T}_{\mathcal{N}}\}$, where the t-th task $\mathcal{T}_t = \{(\boldsymbol{x}_i, \boldsymbol{y}_i)\}_{i=1}^{N_t}$ becomes available only at step t. Let $\boldsymbol{\theta}_0$ denote the frozen pre-trained backbone weights. Our objective is to learn a set of adaptive parameters $\Delta\boldsymbol{\theta}$ such that the model $\mathcal{F}_{\boldsymbol{\theta}_0+\Delta\boldsymbol{\theta}}$ minimizes the empirical risk on the current task $\mathcal{T}_t$ while maintaining performance on all previous tasks $\mathcal{T}_{<t}$, without accessing any historical data or replay buffers.

Metrics. We adopt the following metrics to quantify various performances: 1) $\mathbf{FP} = \frac{1}{N}\sum_{j=1}^{N} a_N^{T_j}$ is the average zero-shot performance across all N tasks after tuning on the final N-th task. Here, a_m^q denotes the zero-shot performance on task q after sequentially tuning the m-th task, and T_j refers to the j-th task in the sequence. 2) $\mathbf{AP} = \frac{1}{N}\sum_{j=1}^{N} a_j^{T_j}$ is the average zero-shot performance when learning each j-th task, which measures the plasticity of the model. 3) $\mathbf{Forget} = \mathbf{AP} - \mathbf{FP}$ is calculated as the difference between $\mathbf{AP}$ and $\mathbf{FP}$, as commonly used in previous studies (Wu et al., 2022; Jiang et al., 2025) to quantify forgetting.

3.2 Dual-Branch Spectral Architecture

To resolve the *Rank-Blindness* of standard adapters, we structurally decouple the adaptation process into two orthogonal subspaces. We integrate a tri-branch structure into the linear layers (e.g., Query/Value projections) of the Transformer. For a given input $\boldsymbol{x} \in \mathbb{R}^{d_{in}}$, the output representation $\boldsymbol{h} \in \mathbb{R}^{d_{out}}$ is computed as:

$$\boldsymbol{h} = \underbrace{\mathbf{W}_0\boldsymbol{x}}_{\text{Original}} + \underbrace{\boldsymbol{\lambda}_l(\mathbf{W}_l\boldsymbol{x})}_{\text{Shared Subspace}} + \underbrace{\boldsymbol{\lambda}_h(\mathbf{W}_h\boldsymbol{x})}_{\text{Specific Subspace}} \quad (1)$$

where $\mathbf{W}_0$ represents the frozen backbone linear weights. The modulation vectors $\boldsymbol{\lambda}_l, \boldsymbol{\lambda}_h \in \mathbb{R}^{d_{out}}$ (generated by the router) perform channel-wise scaling to dynamically fuse features.

The Low-Rank (Shared) Branch: Designed to capture task-invariant capabilities (e.g., logical reasoning, syntax), this branch is constrained to a low intrinsic dimension. It is parameterized by $\mathbf{W}_l = \mathbf{B}_l\mathbf{A}_l$, where $\mathbf{A}_l \in \mathbb{R}^{r_l \times d_{in}}$ and $\mathbf{B}_l \in \mathbb{R}^{d_{out} \times r_l}$. We enforce a tight rank constraint $r_l \ll \min(d_{in}, d_{out})$ to encourage the learning of compact, transferable structures.

The High-Rank (Specific) Branch: Designed to accommodate task-idiosyncratic mappings (e.g., domain facts, rote memorization), this branch operates in a higher-dimensional subspace. It is parameterized by $\mathbf{W}_h = \mathbf{B}_h\mathbf{A}_h$ with rank r_h. Crucially, we set $r_h > r_l$ (e.g., $r_h = 4r_l$) to provide sufficient geometric capacity for high-entropy updates, preventing the information bottleneck.

3.3 Spectrum-Aware Dynamic Router

Ideally, the model should dynamically decide whether to invoke shared skills or specific memories for each input token. To achieve this, we propose a **Spectrum-Aware Dynamic Router** that predicts the modulation vectors $\boldsymbol{\lambda}_l, \boldsymbol{\lambda}_h$ in Eq. (1).

Instead of using static scalars, we employ a **hyper-network approach** to generate context-aware weights (Liao et al., 2024). We maintain a pool of learnable routing components $\mathcal{M} = \{(\mathbf{K}_m, \mathbf{V}_m, \mathbf{A}_m)\}_{m=1}^{M}$, where M is a hyperparameter that controls the capacity of the hyper-network (i.e., the number of routing components). Each component consists of: (i) a key vector $\mathbf{K}_m$ for context matching, (ii) a value vector $\mathbf{V}_m$ that encodes the routing signature and maps to the modulation vector $\boldsymbol{\lambda}$, and (iii) an attention vector $\mathbf{A}_m$ that acts as a spectral filter over the input representation.

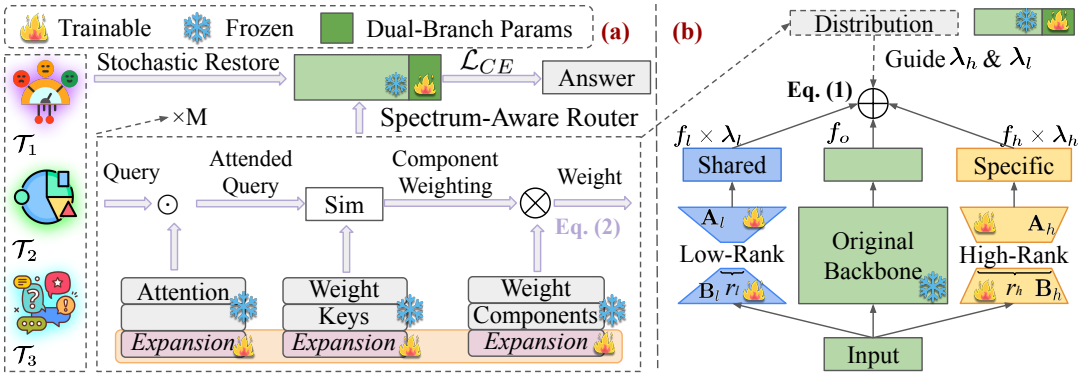


Figure 3: **The SPARTA Framework.** (a) **Spectrum-Aware Dynamic Router:** An expandable memory bank serves as a hyper-network to generate context-aware modulation signals. It takes the input query and acts as a *prism*, decomposing the adaptation requirement into shared ($\boldsymbol{\lambda}_l$) and specific ($\boldsymbol{\lambda}_h$) coefficients. An orthogonality constraint ($\mathcal{L}_{ortho}$) enforces separation between these subspaces, while the router dynamically fuses them via $\boldsymbol{\lambda}_l$ and $\boldsymbol{\lambda}_h$.

For an input query $\boldsymbol{x}$, we first compute a relevance score vector $\boldsymbol{\alpha} \in \mathbb{R}^M$. To enhance feature selection, each query is modulated by component-specific attention vectors $\mathbf{A} \in \mathbb{R}^{d_{in} \times M}$ via element-wise multiplication before similarity computation:

$$\boldsymbol{\alpha} = \text{Softmax}\left(\frac{\text{Sim}(\boldsymbol{x} \odot \mathbf{A}, \mathbf{K})}{\tau}\right) \tag{2}$$

where $\mathbf{K} \in \mathbb{R}^{d_{in} \times M}$ stacks the routing keys, $\odot$ denotes the Hadamard product, and τ is a temperature parameter. This design enables the router to attend to different **spectral bands** of the input embedding for different routing components, effectively suppressing task-irrelevant features prior to similarity computation. As a result, routing decisions become more stable under domain shifts and better aligned with task semantics.

The final routing weights $\boldsymbol{\lambda}$ are synthesized as a weighted combination of the value components:

$$\boldsymbol{\lambda} = \sum_{m=1}^{M} \alpha_m \mathbf{V}_m \tag{3}$$

where $\mathbf{V}_m \in \mathbb{R}^{d_{\text{out}}}$. This hyper-network formulation allows SPARTA to dynamically route each input to an appropriate combination of low-rank and high-rank adaptation subspaces, thereby balancing task-shared generalization and domain-specific specialization in a unified and input-adaptive manner.

3.4 Progressive Subspace Isolation

To enable lifelong learning without catastrophic forgetting, we implement a **Progressive Subspace Isolation** strategy. Given a sequence of tasks, we partition the component pool $\mathcal{M}$ into task-specific subsets. When learning a new task $\mathcal{T}_t$, we unlock only a fraction ($M/\mathcal{N}$) of new components $\{\mathbf{K}_{new}, \mathbf{V}_{new}, \mathbf{A}_{new}\}$, while strictly **freezing** all previously learned components.

This strategy serves two purposes: (1) **Forward Transfer:** The router can still attend to frozen old components ($\alpha_{old} > 0$) if the current input shares similarity with past tasks, enabling knowledge reuse. (2) **Backward Stability:** By physically isolating the parameters for different temporal stages, we structurally eliminate the risk of overwriting prior knowledge.

3.5 Orthogonal Subspace Regularization

Structural separation alone does not guarantee semantic disentanglement. To prevent the new components from redundantly learning old patterns (which leads to *subspace collision*), we impose an **Orthogonality Constraint**. We enforce the projection matrices of different components to be orthogonal, thereby promoting more effective knowledge partitioning and improving long-term retention:

$$\mathcal{L}_{\text{ortho}} = \sum_{\mathbf{P} \in \{\mathbf{K}, \mathbf{V}, \mathbf{A}\}} \|\mathbf{P}^\top \mathbf{P} - \mathbf{I}\|_F^2 \tag{4}$$

where $\|\cdot\|_F$ is the Frobenius norm. This regularization pushes the routing components to span diverse directions in the optimization landscape, ensuring that new tasks occupy unoccupied subspaces (as visualized in Figure 2(C)).

Stochastic Plasticity Restoration. Finally, to balance stability with plasticity, we adopt a **Stochastic Restoration** strategy. During training,

	Methods	Standard CL Benchmark (SC)			Long Sequence Benchmark (LS)			TRACE		
		FP ↑	AP ↑	Forget ↓	FP ↑	AP ↑	Forget ↓	FP ↑	AP ↑	Forget ↓
T5-Large	L2P* (Wang et al., 2021)	60.7	-	-	56.1	-	-	-	-	-
	LFPT5* (Qin and Joty, 2021)	72.7	-	-	69.2	-	-	-	-	-
	ProgPrompt* (Razdaibiedina et al., 2023)	75.1	-	-	**77.9**	-	-	-	-	-
	IncLoRA	65.7	68.1	2.4	59.7	66.3	6.6	-	-	-
	SeqLoRA	$70.7_{\pm.39}$	**76.7**$_{\pm.43}$	6.0	$59.9_{\pm.56}$	$73.3_{\pm.28}$	13.4	$12.1_{\pm.82}$	$44.5_{\pm.94}$	32.4
	LoRAReplay	$73.3_{\pm.42}$	$76.6_{\pm.51}$	3.3	$73.6_{\pm.36}$	**75.4**$_{\pm.59}$	1.8	$34.0_{\pm.62}$	**46.8**$_{\pm.63}$	12.8
	O-LoRA (Wang et al., 2023b)	$72.0_{\pm.63}$	$74.4_{\pm.47}$	2.4	$67.9_{\pm.82}$	$70.3_{\pm.65}$	2.4	-	-	-
	+ MIGU (Du et al., 2024)	$71.6_{\pm.45}$	$73.9_{\pm.67}$	2.3	$65.3_{\pm.35}$	$68.0_{\pm.47}$	2.7	-	-	-
	SPARTA (*ours*)	$72.7_{\pm.58}$	$74.8_{\pm.68}$	2.1	$70.0_{\pm.44}$	$71.8_{\pm.36}$	1.8	$16.7_{\pm.21}$	$41.3_{\pm.58}$	24.6
	+ Replay	**75.9**$_{\pm.24}$	$76.5_{\pm.75}$	**0.6**	$74.3_{\pm.35}$	$74.7_{\pm.91}$	**0.4**	**36.5**$_{\pm.32}$	$45.2_{\pm.61}$	**8.7**
	MTL	80.0	-	-	76.5	-	-	39.8	-	-
LLaMA2-7B	SeqLoRA	$74.9_{\pm.42}$	**80.7**$_{\pm.38}$	5.8	$73.7_{\pm.66}$	$80.0_{\pm.65}$	6.3	$64.1_{\pm.49}$	$77.9_{\pm.54}$	13.8
	LoRAReplay	$79.2_{\pm.53}$	**80.7**$_{\pm.61}$	1.5	$80.0_{\pm.45}$	**81.3**$_{\pm.57}$	1.3	$71.9_{\pm.62}$	**78.6**$_{\pm.75}$	6.7
	O-LoRA (Wang et al., 2023b)	$76.4_{\pm.74}$	$78.7_{\pm.59}$	2.3	$67.9_{\pm.74}$	$71.9_{\pm.49}$	4.0	$35.0_{\pm.34}$	$47.0_{\pm.42}$	12.0
	SPARTA (*ours*)	$79.8_{\pm.54}$	$80.3_{\pm.49}$	0.5	$76.9_{\pm.35}$	$79.7_{\pm.36}$	2.8	$66.0_{\pm.33}$	$74.6_{\pm.38}$	8.6
	+ Replay	**80.0**$_{\pm.24}$	$80.4_{\pm.45}$	**0.4**	**81.8**$_{\pm.61}$	$80.6_{\pm.84}$	**-1.2**	**73.1**$_{\pm.46}$	$77.5_{\pm.98}$	**4.4**
	MTL	83.6	-	-	85.1	-	-	80.8	-	-
LLaMA3.1-8B	SeqLoRA	$79.6_{\pm.62}$	$80.8_{\pm.49}$	5.8	$74.8_{\pm.58}$	$83.8_{\pm.52}$	9.0	$65.1_{\pm.69}$	$82.4_{\pm.54}$	17.3
	LoRAReplay	$80.3_{\pm.71}$	**80.9**$_{\pm.56}$	0.6	$82.0_{\pm.69}$	**85.0**$_{\pm.75}$	3.0	**78.7**$_{\pm.83}$	**85.7**$_{\pm.68}$	7.0
	O-LoRA (Wang et al., 2023b)	$72.3_{\pm.86}$	$73.9_{\pm.68}$	1.6	$71.4_{\pm.64}$	$74.8_{\pm.64}$	3.7	$36.7_{\pm.57}$	$50.1_{\pm.37}$	13.4
	SPARTA (*ours*)	**80.9**$_{\pm.40}$	$80.6_{\pm.35}$	-0.3	$80.0_{\pm.53}$	$82.3_{\pm.48}$	2.3	$72.7_{\pm.94}$	$80.4_{\pm.88}$	7.7
	+ Replay	$80.8_{\pm.33}$	$80.4_{\pm.47}$	**-0.4**	**82.2**$_{\pm.66}$	$82.6_{\pm.77}$	**0.4**	$77.6_{\pm.37}$	$81.0_{\pm.72}$	**3.4**
	MTL	84.2	-	-	86.6	-	-	81.4	-	-

Table 1: Performance of baselines and ours **SPARTA** on standard CL benchmark (Order 1,2,3) and long sequence benchmark (Order 4,5,6) and TRACE (Order 7). **Bold** indicates the best in each setting and $*$ means that those results are from their papers. We report the mean and standard deviation of results with 3 different runs.

we randomly revert a subset of trainable parameters to their initial states:

$$\boldsymbol{\theta}_{t+1} \leftarrow \mathbf{M} \odot \boldsymbol{\theta}_{init} + (1 - \mathbf{M}) \odot \boldsymbol{\theta}_{t+1} \quad (5)$$

where $\mathbf{M} \sim \text{Bernoulli}(\boldsymbol{p})$ and $\boldsymbol{p}$ is a small probability. This acts as a regularization akin to Dropout, preventing the model from becoming overly rigid to the current task's specific distribution.

Total Objective. The final optimization objective for task $\mathcal{T}_t$ minimizes the Cross-Entropy loss $\mathcal{L}_{CE}$ alongside the orthogonality penalty:

$$\mathcal{L}_{total} = \mathcal{L}_{CE}(\mathcal{T}_t) + \beta \cdot \mathcal{L}_{\text{ortho}} \quad (6)$$

where β controls the subspace separation strength.

4 Experiments

4.1 Experimental Setup

We evaluate SPARTA on three rehearsal-free continual-learning benchmarks with increasing interference difficulty: the Standard CL Benchmark (SC) (Zhang et al., 2015; Wang et al., 2023b), the Long Sequence Benchmark (LS) (Razdaibiedina et al., 2023), and TRACE (Wang et al., 2023c). SC contains four classification tasks, LS extends the stream to 15 tasks, and TRACE stresses broad capability retention with QA, multilingual understanding, code, and mathematical reasoning. We experiment with T5-Large, LLaMA2-7B, LLaMA3.1-8B, and Qwen2.5-7B. The baselines span the main design families in the literature, including prompt-based methods, regularization-based methods, rehearsal-free PEFT baselines, and replay-enhanced variants: L2P, LFPT5, ProgPrompt, IncLoRA, SeqLoRA, LoRAReplay, MIGU, and O-LoRA. We average our main results over three runs. O-LoRA is the strongest rehearsal-free single-rank baseline in our setting, while LoRAReplay serves as a reference point when memory is allowed. This distinction matters because our claim is not that replay is unnecessary, but that rank-aware structure strengthens the rehearsal-free frontier under a comparable budget. Additional dataset, task-order, and hyperparameter details are deferred to Appendix B.

4.2 Main Results: Continual Learning Performance

Table 1 reports the main continual-learning results. The overall pattern is consistent across scales and backbones: SPARTA improves the rehearsal-free retention–plasticity trade-off, with the largest gains appearing where single-rank methods are most brittle. On T5-Large, the advantage is already visible on the harder long-stream setting: compared with O-LoRA, SPARTA improves LS final performance from 67.9 to 70.0 and reduces forgetting from 2.4 to 1.8. The gains become much larger on decoder-

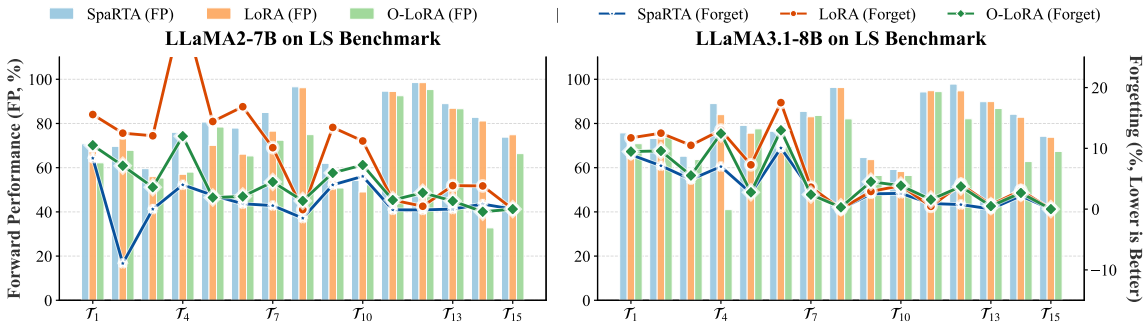


Figure 4: **Learning dynamics on the long-sequence benchmark.** We track final performance (**FP**, bars) and forgetting (**Forget**, lines) throughout Order 4. SPARTA follows a more stable trajectory than prior rehearsal-free baselines, consistent with the view that spectral disentanglement reduces forgetting without sacrificing downstream performance.

only LLMs, where interference is stronger and capacity allocation matters more. On LLaMA2-7B, SPARTA cuts forgetting from 2.3 to 0.5 on SC and from 4.0 to 2.8 on LS, while raising TRACE final performance from 35.0 to 66.0. On LLaMA3.1-8B, it lifts SC final performance from 72.3 to 80.9, LS final performance from 71.4 to 80.0, and TRACE final performance from 36.7 to 72.7, while reducing TRACE forgetting from 13.4 to 7.7.

Two observations matter most. First, the relative gain of SPARTA grows with stream difficulty. The gap is modest on SC, larger on LS, and largest on TRACE, which is consistent with the spectral-interference hypothesis: as the stream becomes longer and more heterogeneous, forcing all tasks into one shared low-rank geometry becomes increasingly harmful. Second, SPARTA improves retention without simply turning conservative. Its **AP** remains competitive with strong baselines, making it less likely that the gains come from suppressing adaptation. Taken together, the results support our main interpretation that the key bottleneck lies in the *sharing scheme* of the adapter space, not only in the total parameter count.

We also observe supplementary evidence beyond the main tables on a distinct multimodal backbone. In the RoboBrain2.5-4B study (Appendix C.3), SPARTA outperforms a single-rank LoRA baseline across all three task orders, achieving final performance of 80.6, 78.0, and 80.0 with forgetting of only 0.2, 2.6, and 0.3. We treat this result as supporting evidence rather than as a standalone claim, but it is consistent with the view that the same rank-aware split transfers beyond a single model family without changing the core algorithm.

4.3 Unseen-Task Generalization

To test whether SPARTA preserves task-invariant capabilities rather than merely fitting the observed stream, we evaluate the final model on unseen benchmarks spanning reasoning and classification. Table 2 shows a clear pattern. Relative to O-LoRA, SPARTA improves MMLU from 62.79 to 64.47 and GSM8K from 1.56 to 3.63, while also raising final in-domain performance from 71.83 to 81.39. It also remains stronger than LoRAReplay on MMLU and GSM8K despite using no replay storage. These results matter because they help distinguish two explanations for good continual performance: a method can look stable either because it preserves reusable structure or because it overfits the observed stream in a way that merely delays forgetting. The unseen-task gains favor the first interpretation. More cautiously, they suggest that the shared low-rank branch preserves capabilities that remain useful beyond the training trajectory itself.

4.4 Efficiency and Overhead

An effective continual-learning method must also remain practical. Table 3 shows that SPARTA maintains a favorable efficiency–performance trade-off. Relative to O-LoRA, SPARTA reduces training FLOPs from 8.8 to 4.6 ($\times 10^{16}$), lowers the trainable-parameter ratio from 0.46% to 0.38%, and cuts prediction latency from 196 ms to 141 ms, all while remaining strictly rehearsal-free. SeqLoRA is still slightly cheaper at inference time, but it pays for that simplicity with markedly worse forgetting on the harder benchmarks. This comparison matters because it narrows an alternative explanation for the gains: they are not attributable

Methods	MMLU	BBH	GSM8K	AGIEval	FP
Zero-Shot	65.65	62.12	56.33	17.72	-
SeqLoRA	63.58	**11.90**	0.00	**20.60**	79.92
LoRAReplay	60.24	5.99	1.82	10.69	80.13
O-LoRA	62.79	6.31	1.56	13.87	71.83
SPARTA	**64.47**	10.42	**3.63**	16.94	**81.39**
MTL	66.48	28.87	22.59	22.18	84.12

Table 2: Task generalization comparisons on unseen tasks based the LLaMA3.1-8B after training in Order 1.

Method	FLOPs (10^{16}) ↓	Trainable Parameters (%) ↓	Stored Features (%) ↓	Predict Time (ms) ↓
SeqLoRA	2.6	0.30	0	89
LoRARepaly	4.8	0.30	2%	90
O-LoRA	8.8	0.46	0	196
SPARTA	4.6	0.38	0	141

Table 3: Comparison of the number of trainable parameters and FLOPs for Order 4 with LLaMA2-7B.

to substantially more optimization budget, replay storage, or hidden parameter growth, but to a more effective use of a similar budget.

4.5 Mechanistic Analysis

Learning Dynamics and Forward Transfer. Figure 4 visualizes the training trajectory on a representative long-stream order. The main takeaway is dynamical rather than only final-state: SPARTA exhibits smaller performance drops at task boundaries and a flatter forgetting curve than O-LoRA. This is the pattern we would expect if a new task can recruit fresh task-specific directions without rewriting the reusable subspace.

Table 9 provides a more fine-grained view of retained knowledge after the full Order 1 stream. After training on all four tasks, SPARTA retains 97.76 on DBPedia versus 93.77 for O-LoRA, 57.22 on Amazon versus 55.65, 71.41 on Yahoo versus 68.43, and 91.61 on AGNews versus 87.0. The gains are not isolated to one task; they appear across the full task trajectory. This broad retention pattern is consistent with the paper's mechanistic account: spectral disentanglement does not merely protect one early task, but preserves a larger fraction of the learned trajectory.

4.6 Ablation Study

Table 4 isolates the contribution of each component. The pattern is highly structured. Starting from the minimal baseline (E_1, 73.0 FP), enabling only the high-rank branch reaches 73.7, while enabling only the low-rank branch reaches 75.4. This asymmetry is informative: the shared low-rank branch already captures more broadly useful structure than the high-rank branch in isolation. However, neither branch alone is sufficient. Combining them in E_4 raises final performance to 77.6, a gain of 2.2 points over the best single-branch variant.

	SPARTA$_h$	SPARTA$_l$	Weight	Attention	Ortho.	Rest.	FP ↑
E_1	-	-	-	-	-	-	73.0
E_2	✓	-	-	-	-	-	73.7
E_3	-	✓	-	-	-	-	75.4
E_4	✓	✓	-	-	-	-	77.6
E_5	✓	✓	-	-	-	✓	78.1
E_6	✓	✓	✓	-	-	✓	78.6
E_7	✓	✓	✓	✓	-	✓	79.2
E_8	✓	✓	✓	✓	✓	✓	**79.5**

Table 4: Ablation studies on different components.

The remaining gains then align with the rest of our design. Stochastic restoration improves $E_4 \rightarrow E_5$ from 77.6 to 78.1, showing that mild parameter reset helps sustain plasticity. Learned modulation weights improve $E_5 \rightarrow E_6$ to 78.6, and the full dynamic router improves $E_6 \rightarrow E_7$ to 79.2, indicating that input-dependent branch allocation matters beyond static branch coexistence. Orthogonality further improves $E_7 \rightarrow E_8$ from 79.2 to 79.5, confirming that explicit subspace separation remains beneficial even after architectural disentanglement. In short, the ablation study suggests that SPARTA benefits from the interaction of its components rather than from one isolated trick.

5 Related Work

Continual Learning paradigms. Existing CL methods generally fall into three categories. *Rehearsal-based* approaches (Tiwari et al., 2021; Wang et al., 2024b; He et al., 2024) retain historical samples, fundamentally compromising privacy and storage efficiency (Sun et al., 2019). *Regularization-based* methods (Zhu et al., 2024; Du et al., 2024) constrain parameter updates but frequently struggle with the stability-plasticity trade-off. While *architecture-based* strategies (Wang et al., 2023b; Zhao et al., 2024) expand parameters to reduce interference, they typically rely on discrete, expert-based routing that fails to guarantee semantic separation. In contrast, **SPARTA** introduces a spectral perspective, leveraging orthogonal subspaces to physically disentangle task-invariant structures from idiosyncratic mappings without data replay.

PEFT in Continual Learning. While PEFT techniques like LoRA (Hu et al., 2022) have been adapted for CL (Wang et al., 2023b; Zhao et al., 2024), current methods predominantly suffer from

Rank-Blindness. By enforcing a uniform rank across all tasks, they ignore the varying intrinsic dimensionality of knowledge, causing information bottlenecks for complex tasks or subspace interference for simple ones. **SPARTA** resolves this by dynamically orchestrating a *spectrum-aware* interplay between low-rank (shared) and high-rank (specific) adapters (Liu et al., 2023), ensuring robust adaptation across diverse task complexities.

Structured Reasoning Transfer Beyond CL. Recent efficient-reasoning work increasingly avoids monolithic transfer and instead decomposes what should be distilled, internalized, or generated. Representative directions include rationale-based reasoning distillation (Ho et al., 2023; Hsieh et al., 2023; Magister et al., 2023), knowledge-augmented or neural-symbolic transfer (Kang et al., 2023; Liao et al., 2025b), progressive internalization of external prompts or symbolic cues (Zou et al., 2024; Liao et al., 2025a), and instruction-conditioned generation of task-specific adapters (Liao et al., 2024). Although these methods target reasoning distillation, inference efficiency, or cross-task generalization rather than rehearsal-free continual learning, they reinforce the same high-level intuition behind SPARTA: heterogeneous knowledge is better handled through structured factorization than through a single uniform adaptation channel.

6 Conclusion

We study rehearsal-free continual learning through the lens of rank allocation and identify *rank blindness* as the central failure mode revealed by our analysis of long, heterogeneous task streams. A single fixed-rank adapter forces reusable structure and task-specific detail into the same update space, which weakens plasticity on hard tasks and intensifies forgetting as task interference accumulates. SPARTA addresses this mismatch by separating low-rank shared adaptation from high-rank task-specific adaptation, modulating both with a spectrum-aware router, and explicitly discouraging cross-task subspace collision. The evidence is coherent at three levels: spectral analysis exposes heterogeneous rank demand and subspace collision, ablations show that the proposed components contribute complementary gains, and benchmark results show stronger retention, competitive plasticity, and better unseen-task generalization across multiple backbones. Taken together, these findings suggest that continual adaptation benefits from organizing capacity by *functional role* rather than by a single global rank budget. At the same time, our evidence is limited to the task streams and model scales studied here. Testing whether the same principle extends to larger multimodal models and more open-ended lifelong settings remains an important direction for future work.

Acknowledgements

This work was supported by the Beijing Major Science and Technology Project (No. Z251100008125025), the National Natural Science Foundation of China (No.62376270, No.62476060) and the independent research project of the Key Laboratory of Cognition and Decision Intelligence for Complex Systems.

References

BIG bench authors. 2023. Beyond the imitation game: Quantifying and extrapolating the capabilities of language models. *Transactions on Machine Learning Research*.

Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, Christopher Hesse, and John Schulman. 2021. Training verifiers to solve math word problems. *arXiv preprint arXiv:2110.14168*.

OpenCompass Contributors. 2023. Opencompass: A universal evaluation platform for foundation models. https://github.com/open-compass/opencompass.

Shibhansh Dohare, J. Fernando Hernandez-Garcia, Qingfeng Lan, Parash Rahman, A. Ruapm Mahmood, and Richard S. Sutton. 2024. Loss of plasticity in deep continual learning. *Nature*, 632:768—774.

Shibhansh Dohare, Ashique Rupam Mahmood, and Richard S. Sutton. 2021. Continual backprop: Stochastic gradient descent with persistent randomness. *ArXiv*, abs/2108.06325.

Wenyu Du, Shuang Cheng, Tongxu Luo, Zihan Qiu, Zeyu Huang, Ka Chun Cheung, Reynold Cheng, and Jie Fu. 2024. Unlocking continual learning abilities in language models. In *Conference on Empirical Methods in Natural Language Processing*.

Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Amy Yang, Angela Fan, Anirudh Goyal, Anthony S. Hartshorn, Aobo Yang, Archi Mitra, Archie Sravankumar, Artem Korenev, Arthur Hinsvark, Arun Rao, Aston Zhang, and et al. 2024. The llama 3 herd of models. *ArXiv*, abs/2407.21783.

Jinghan He, Haiyun Guo, Kuan Zhu, Zihan Zhao, Ming Tang, and Jinqiao Wang. 2024. SEEKR: Selective attention-guided knowledge retention for continual learning of large language models. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 3254–3266, Miami, Florida, USA. Association for Computational Linguistics.

Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. 2021. Measuring massive multitask language understanding. In *International Conference on Learning Representations*.

Namgyu Ho, Laura Schmid, and Se-Young Yun. 2023. Large language models are reasoning teachers. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 14852–14882, Toronto, Canada. Association for Computational Linguistics.

Neil Houlsby, Andrei Giurgiu, Stanislaw Jastrzebski, Bruna Morrone, Quentin de Laroussilhe, Andrea Gesmundo, Mona Attariyan, and Sylvain Gelly. 2019. Parameter-efficient transfer learning for nlp. *ArXiv*, abs/1902.00751.

Cheng-Yu Hsieh, Chun-Liang Li, Chih-kuan Yeh, Hootan Nakhost, Yasuhisa Fujii, Alex Ratner, Ranjay Krishna, Chen-Yu Lee, and Tomas Pfister. 2023. Distilling step-by-step! outperforming larger language models with less training data and smaller model sizes. In *Findings of the Association for Computational Linguistics: ACL 2023*, pages 8003–8017, Toronto, Canada. Association for Computational Linguistics.

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

Yebowen Hu, Tim Ganter, Hanieh Deilamsalehy, Franck Dernoncourt, Hassan Foroosh, and Fei Liu. 2023. Meetingbank: A benchmark dataset for meeting summarization. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (ACL)*, Toronto, Canada. Association for Computational Linguistics.

Gangwei Jiang, Caigao JIANG, Zhaoyi Li, Siqiao Xue, JUN ZHOU, Linqi Song, Defu Lian, and Ying Wei. 2025. Unlocking the power of function vectors for characterizing and mitigating catastrophic forgetting in continual instruction tuning. In *The Thirteenth International Conference on Learning Representations*.

Minki Kang, Seanie Lee, Jinheon Baek, Kenji Kawaguchi, and Sung Ju Hwang. 2023. Knowledge-augmented reasoning distillation for small language models in knowledge-intensive tasks. In *Advances in Neural Information Processing Systems 37: Annual Conference on Neural Information Processing Systems 2023, NeurIPS 2023, December 10-16, 2023, New Orleans*.

Huanxuan Liao, Shizhu He, Yupu Hao, Xiang Li, Yuanzhe Zhang, Jun Zhao, and Kang Liu. 2025a. Skintern: Internalizing symbolic knowledge for distilling better cot capabilities into small language models. In *Proceedings of the 31st International Conference on Computational Linguistics*, pages 3203–3221.

Huanxuan Liao, Shizhu He, Yao Xu, Yuanzhe Zhang, Yanchao Hao, Shengping Liu, Kang Liu, and Jun Zhao. 2024. From instance training to instruction learning: Task adapters generation from instructions. *Advances in Neural Information Processing Systems*, 37:45552–45577.

Huanxuan Liao, Shizhu He, Yao Xu, Yuanzhe Zhang, Kang Liu, and Jun Zhao. 2025b. Neural-symbolic collaborative distillation: Advancing small language models for complex reasoning tasks. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 39, pages 24567–24575.

Huanxuan Liao, Zhongtao Jiang, Yupu Hao, Yuqiao Tan, Shizhu He, Ben Wang, Jun Zhao, Kun Xu, and Kang Liu. 2026. Resadapt: Adaptive resolution for efficient multimodal reasoning. *arXiv preprint*.

Jiaheng Liu, Dawei Zhu, Zhiqi Bai, Yancheng He, Huanxuan Liao, Haoran Que, Zekun Wang, Chenchen Zhang, Ge Zhang, Jiebin Zhang, and 1 others. 2025. A comprehensive survey on long context language modeling. *arXiv preprint arXiv:2503.17407*.

Jiaming Liu, Senqiao Yang, Peidong Jia, Ming Lu, Yandong Guo, Wei Xue, and Shanghang Zhang. 2023. Vida: Homeostatic visual domain adapter for continual test time adaptation. *ArXiv*, abs/2306.04344.

Pan Lu, Swaroop Mishra, Tony Xia, Liang Qiu, Kai-Wei Chang, Song-Chun Zhu, Oyvind Tafjord, Peter Clark, and Ashwin Kalyan. 2022. Learn to explain: Multimodal reasoning via thought chains for science question answering. In *The 36th Conference on Neural Information Processing Systems (NeurIPS)*.

Lucie Charlotte Magister, Jonathan Mallinson, Jakub Adamek, Eric Malmi, and Aliaksei Severyn. 2023. Teaching small language models to reason. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 1773–1781, Toronto, Canada. Association for Computational Linguistics.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Köpf, Edward Yang, Zach DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, and 2 others. 2019. Pytorch: an imperative style, high-performance deep learning library. In *Proceedings*

of the 33rd International Conference on Neural Information Processing Systems, Red Hook, NY, USA. Curran Associates Inc.

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

Colin Raffel, Noam M. Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. 2019. Exploring the limits of transfer learning with a unified text-to-text transformer. *J. Mach. Learn. Res.*, 21:140:1–140:67.

Anastasia Razdaibiedina, Yuning Mao, Rui Hou, Madian Khabsa, Mike Lewis, and Amjad Almahairi. 2023. Progressive prompts: Continual learning for language models. In *International Conference on Learning Representations*.

Baptiste Roziere, Jonas Gehring, Fabian Gloeckle, Sten Sootla, Itai Gat, Xiaoqing Ellen Tan, Yossi Adi, Jingyu Liu, Romain Sauvestre, Tal Remez, and 1 others. 2023. Code llama: Open foundation models for code. *arXiv preprint arXiv:2308.12950*.

Fan-Keng Sun, Cheng-Hao Ho, and Hung yi Lee. 2019. Lamol: Language modeling for lifelong language learning. In *International Conference on Learning Representations*.

Mirac Suzgun, Nathan Scales, Nathanael Scharli, Sebastian Gehrmann, Yi Tay, Hyung Won Chung, Aakanksha Chowdhery, Quoc V. Le, Ed Huai hsin Chi, Denny Zhou, and Jason Wei. 2022. Challenging big-bench tasks and whether chain-of-thought can solve them. In *Annual Meeting of the Association for Computational Linguistics*.

Huajie Tan, Enshen Zhou, Zhiyu Li, Yijie Xu, Yuheng Ji, Xiansheng Chen, Cheng Chi, Pengwei Wang, Huizhu Jia, Yulong Ao, and 1 others. 2026. Robobrain 2.5: Depth in sight, time in mind. *arXiv preprint arXiv:2601.14352*.

Rishabh Tiwari, Krishnateja Killamsetty, Rishabh K. Iyer, and Pradeep Shenoy. 2021. Gcr: Gradient coreset based replay buffer selection for continual learning. *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 99–108.

Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, and 1 others. 2023. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*.

Laurens van der Maaten and Geoffrey E. Hinton. 2008. Visualizing data using t-sne. *Journal of Machine Learning Research*, 9:2579–2605.

Mingyang Wang, Heike Adel, Lukas Lange, Jannik Strotgen, and Hinrich Schütze. 2024a. Rehearsal-free modular and compositional continual learning for language models. In *North American Chapter of the Association for Computational Linguistics*.

Qin Wang, Olga Fink, Luc Van Gool, and Dengxin Dai. 2022. Continual test-time domain adaptation. *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 7191–7201.

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

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

Xiao Wang, Yuansen Zhang, Tianze Chen, Songyang Gao, Senjie Jin, Xianjun Yang, Zhiheng Xi, Rui Zheng, Yicheng Zou, Tao Gui, and 1 others. 2023c. Trace: A comprehensive benchmark for continual learning in large language models. *arXiv preprint arXiv:2310.06762*.

Yifan Wang, Yafei Liu, Chufan Shi, Haoling Li, Chen Chen, Haonan Lu, and Yujiu Yang. 2024b. InsCL: A data-efficient continual learning paradigm for fine-tuning large language models with instructions. In *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pages 663–677, Mexico City, Mexico. Association for Computational Linguistics.

Zifeng Wang, Zizhao Zhang, Chen-Yu Lee, Han Zhang, Ruoxi Sun, Xiaoqi Ren, Guolong Su, Vincent Perot, Jennifer G. Dy, and Tomas Pfister. 2021. Learning to prompt for continual learning. *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 139–149.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Remi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, and 3 others. 2020. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45, Online. Association for Computational Linguistics.

Tongtong Wu, Massimo Caccia, Zhuang Li, Yuan-Fang Li, Guilin Qi, and Gholamreza Haffari. 2022. Pretrained language model in continual learning: A comparative study. In *International Conference on Learning Representations*.

Tongtong Wu, Linhao Luo, Yuan-Fang Li, Shirui Pan, Thuy-Trang Vu, and Gholamreza Haffari. 2024. Continual learning for large language models: A survey. *arXiv preprint arXiv:2402.01364*.

An Yang, Baosong Yang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Zhou, Chengpeng Li, Chengyuan Li, Dayiheng Liu, Fei Huang, Guanting Dong, Haoran Wei, Huan Lin, Jialong Tang, Jialin Wang, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Ma, and 39 others. 2024. Qwen2 technical report. *ArXiv*.

Yuexiang Zhai, Shengbang Tong, Xiao Li, Mu Cai, Qing Qu, Yong Jae Lee, and Yi Ma. 2023. Investigating the catastrophic forgetting in multimodal large language model fine-tuning. In *Conference on Parsimony and Learning (Proceedings Track)*.

Xiang Zhang, Junbo Jake Zhao, and Yann LeCun. 2015. Character-level convolutional networks for text classification. In *Neural Information Processing Systems*.

Zhihua Zhang. 2015. The singular value decomposition, applications and beyond. *arXiv preprint arXiv:1510.08532*.

Weixiang Zhao, Shilong Wang, Yulin Hu, Yanyan Zhao, Bing Qin, Xuanyu Zhang, Qing Yang, Dongliang Xu, and Wanxiang Che. 2024. Sapt: A shared attention framework for parameter-efficient continual learning of large language models. In *Annual Meeting of the Association for Computational Linguistics*.

Yaowei Zheng, Richong Zhang, Junhao Zhang, Yanhan Ye, Zheyan Luo, Zhangchi Feng, and Yongqiang Ma. 2024. Llamafactory: Unified efficient fine-tuning of 100+ language models. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 3: System Demonstrations)*, Bangkok, Thailand. Association for Computational Linguistics.

Wanjun Zhong, Ruixiang Cui, Yiduo Guo, Yaobo Liang, Shuai Lu, Yanlin Wang, Amin Saied, Weizhu Chen, and Nan Duan. 2023. Agieval: A human-centric benchmark for evaluating foundation models. *arXiv preprint arXiv:2304.06364*.

Didi Zhu, Zhongyi Sun, Zexi Li, Tao Shen, Ke Yan, Shouhong Ding, Kun Kuang, and Chao Wu. 2024. Model tailor: Mitigating catastrophic forgetting in multi-modal large language models. *ArXiv*, abs/2402.12048.

Jiaru Zou, Meng Zhou, Tao Li, Shi Han, and Dongmei Zhang. 2024. Promptintern: Saving inference costs by internalizing recurrent prompt during large language model fine-tuning. *ArXiv*, abs/2407.02211.

A Method Details

This appendix provides supporting material for the main paper, including implementation details, dataset and task descriptions, mechanistic evidence, and fine-grained benchmark results.

We build on LoRA (Hu et al., 2022), which expresses parameter updates as a low-rank matrix factorization. In SPARTA, this idea is extended into coordinated low-rank and high-rank branches so that shared and task-specific adaptation need not compete within one single subspace.

For completeness, the standard LoRA update for a linear layer is

$$h' = W_0 x + \Delta W x = (W_0 + AB)x$$

where W_0 is frozen and only the low-rank factors are trainable. SPARTA retains this efficiency while allocating separate ranks to shared and specific adaptation roles.

B Additional Experimental Details

B.1 Datasets

Train Tasks. Tables 5 and 6 summarize the datasets used in our continual-learning experiments. Table 5 lists the 15 datasets in the Long Sequence Benchmark (Razdaibiedina et al., 2023), while Table 6 lists the 8 datasets in TRACE (Wang et al., 2023c). Both tables also report the evaluation metric used for each dataset.

Generalization. For unseen-task evaluation, we use Multitask Language Understanding (MMLU) (Hendrycks et al., 2021), which covers multiple-choice questions across 57 subjects; GSM8K (Cobbe et al., 2021), a linguistically diverse multi-step elementary math reasoning benchmark; BIG-Bench Hard (BBH) (Suzgun et al., 2022), which contains 27 challenging tasks spanning arithmetic, symbolic reasoning, and related skills and is derived from BIG-Bench (BB) (bench authors, 2023); and AGIEval (Zhong et al., 2023), which collects official entrance exams, qualifying exams, and advanced competitions designed for human test takers.

B.2 Task Sequence Orders

We report task orders used for our CL experiments in Table 7.

B.3 Implementation Details

Our implementation is based on Hugging Face Transformers v4.45.2 (Wolf et al., 2020), PyTorch v2.3.1 (Paszke et al., 2019), and LLaMAFactory (Zheng et al., 2024). All unseen-task generalization experiments are conducted with the OpenCompass toolkit (Contributors, 2023) under its default configuration.

Table 5: **Long Sequence benchmark datasets.** The 15 datasets used in the Long Sequence Benchmark (Razdaibiedina et al., 2023) are listed here, and the first five tasks coincide with the Standard CL Benchmark (Zhang et al., 2015).

Dataset Name	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	Natural Language Inference	Various	Accuracy
QQP	GLUE	Paragraph Detection	Quora	Accuracy
RTE	GLUE	Natural Language Inference	News, Wikipedia	Accuracy
SST-2	GLUE	Sentiment Analysis	Movie Reviews	Accuracy
WiC	SuperGLUE	Word Sense Disambiguation	Lexical Databases	Accuracy
CB	SuperGLUE	Natural Language Inference	Various	Accuracy
COPA	SuperGLUE	Question and Answering	Blogs, Encyclopedia	Accuracy
BoolQA	SuperGLUE	Boolean Question and Answering	Wikipedia	Accuracy
MultiRC	SuperGLUE	Question and Answering	Various	Accuracy
IMDB	SuperGLUE	Sentiment Analysis	Movie Reviews	Accuracy

For the Standard CL Benchmark and the Long Sequence Benchmark (Orders 1–6), we train each model for 1 epoch with a constant learning rate of 1e-4.

For TRACE Order 7 (C-STANCE, FOMC, MeetingBank, Py150, ScienceQA, NumGLUE-cm, NumGLUE-ds, and 20Minuten), we use 5000 samples per task with a constant learning rate of 1e-4 and train for 5, 3, 7, 5, 3, 5, 5, and 7 epochs, respectively.

In the main performance experiments, we set the LoRA rank to 8, following the evidence in Figure 2(a), and use a replay ratio of 2% for LoRAReplay. For SPARTA, the low-rank branch uses rank 2 and the high-rank branch uses rank 8. Table 8 shows that this configuration performs best. Compared with vanilla LoRA, SPARTA adds only one extra low-rank adapter with rank 2, yielding a favorable balance between parameter cost and performance on the studied benchmarks.

For the decomposed component weighting strategy, we use a weight length (L_w) of 8. The number of weight components allocated to each task is set to $\frac{N_{\text{layer}}}{4}$, which gives a total of $M = \frac{N \times N_{\text{layer}}}{4}$, where N_{layer} is the number of model layers and N is the number of tasks. We set the hyperparameter β to 10. For stochastic recovery, we apply a simple schedule that restores a small proportion of parameters every 200 training steps.

B.4 Additional Baselines

IncLoRA: An incremental baseline that appends new LoRA parameters as tasks arrive, without additional regularization or replay.
LFPT5 (Qin and Joty, 2021): A soft-prompt method that jointly learns task solving and sample generation, with the generated samples used for replay.
ProgPrompt (Razdaibiedina et al., 2023): A prompt-based continual-learning method that concatenates previously learned prompts to the current prompt during training and inference.
SAPT (Zhao et al., 2024): A routing-based PEFT method that uses the Shared Attentive Learning and Selection Module (SALS) to direct each instance to the most suitable task-specific PET block.

C Additional Analysis and Results

C.1 Mechanistic Evidence for Spectral Disentanglement

To examine whether SPARTA disentangles knowledge representations as intended, we visualize feature distributions and quantify task divergences.
Feature Space Separation (t-SNE). We perform t-SNE analysis (van der Maaten and Hinton, 2008) on the hidden states of LLaMA2-7B across four tasks. As shown in Figure 5 (a), the features processed by the low-rank branch exhibit highly overlapping clusters across tasks. This pattern is consistent with the idea that the low-rank subspace

Table 6: **TRACE dataset statistics.** We summarize the dataset sizes, context sources, average input lengths, and evaluation metrics for TRACE (Wang et al., 2023c).

Dataset	Source	Avg len	Metric	Language	#Data
Domain-specific					
ScienceQA	Science	210	Accuracy	English	5,000
FOMC	Finance	51	Accuracy	English	5,000
MeetingBank	Meeting	2853	ROUGE-L	English	5,000
Multi-lingual					
C-STANCE	Social media	127	Accuracy	Chinese	5,000
20Minuten	News	382	SARI	German	5,000
Code Completion					
Py150	Github	422	Edim Similarity	Python	5,000
Mathematical Reasoning					
NumGLUE-cm	Math	32	Accuracy	English	5,000
NumGLUE-ds	Math	21	Accuracy	English	5,000

Table 7: **Task orders used in continual-learning experiments.** Orders 1–3 follow the Standard CL Benchmark (Zhang et al., 2015), Orders 4–6 correspond to the 15-task Long Sequence Benchmark (Razdaibiedina et al., 2023), and Order 7 denotes TRACE (Wang et al., 2023c).

Benchmark	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 → dbpedia → ag → yelp → amazon → yahoo
	6	yelp → amazon → mnli → cb → copa → qqp → rte → imdb → sst-2 → dbpedia → ag → yahoo → multirc → boolqa → wic
TRACE	7	c-stance → fomc → meetingbank → py150 → scienceqa → numglue-cm → numglue-ds → 20minuten

Orders	2,16	2,8	4,8	4,16
1	79.4408	81.0921	80.8125	80.8387
2	78.5329	80.3684	80.4507	80.8585
3	78.9934	81.8882	80.2007	80.7500

Table 8: **Sensitivity to branch ranks on the Standard CL Benchmark.** We report the performance of SPARTA with LLaMA3.1-8B under different low-rank and high-rank branch configurations.

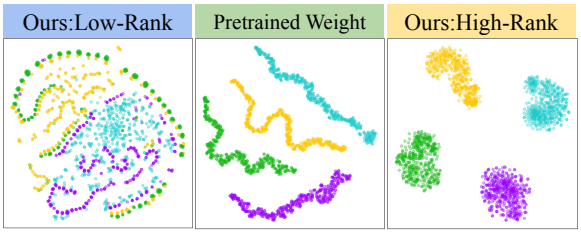
suppresses task-specific noise and captures more task-invariant structure. In contrast, the high-rank branch produces more distinct clusters, reflecting its greater capacity to encode task-specific variation.

Quantifying Stability (H-divergence). We further employ the $\mathcal{H}$-divergence metric to measure distribution shifts. Lower inter-task divergence implies better stability. As shown in Figure 5 (b), the low-rank branch yields lower inter-task divergence than the high-rank branch and the baseline. This result aligns with our spectral hypothesis: reusable capabilities are more stable and transferable, whereas task-specific knowledge introduces larger distributional shifts.

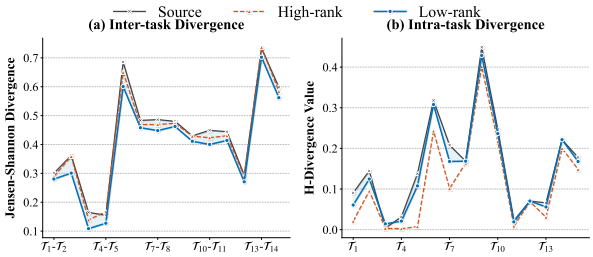
Taken together, these analyses are consistent with the intended functional split in SPARTA: stability is concentrated in the low-rank subspace, while plasticity is concentrated in the high-rank subspace.

C.2 Fine-grained Results for the Main Experiments

We report order-level results for all three benchmarks in Table 10. These results complement Ta-



(a) **t-SNE feature geometry.** The low-rank branch forms more task-invariant clusters than the high-rank branch.



(b) **H-divergence across task transitions.** The low-rank branch yields lower inter-task divergence than the high-rank branch.

Figure 5: **Mechanistic evidence for spectral disentanglement.** The low-rank branch preserves task-invariant geometry and lower inter-task divergence, whereas the high-rank branch remains more task-specific, matching the functional split proposed in SPARTA.

ble 1 in the main text and show that the gains of SPARTA are not driven by a single task order.

Method	Order 1	dbpedia	amazon	yahoo	agnews
SPARTA	dbpedia	99.04			
	amazon	99.01	59.62		
	yahoo	98.78	55.98	75.66	
	agnews	97.76	57.22	71.41	91.61
OLoRA	dbpedia	98.46			
	amazon	98.30	55.24		
	yahoo	96.94	51.50	69.05	
	agnews	93.77	55.65	68.43	87.0

Table 9: Performance comparison across different stages.

C.3 Supplementary Results from the Final Evaluation Logs

We conduct additional experiments on RoboBrain2.5-4B (Tan et al., 2026) to probe the cross-backbone robustness of SPARTA. This multimodal setting is also complementary to recent efficiency-oriented reasoning work such as ResAdapt (Liao et al., 2026), which improves multimodal reasoning from the input-allocation side rather than the continual-adaptation side. Table 11 compares SPARTA against a single-rank LoRA baseline across three task orders.

As shown in Table 11, SPARTA achieves consistently strong **AP** and **FP** across all orders (both around 80) with small forgetting (0.2–2.6 points). Relative to LoRA, SPARTA yields higher final performance and lower forgetting on Order 1 and Order 3, and remains competitive on Order 2. We therefore treat RoboBrain2.5-4B as supplementary evidence that the proposed rank-aware split is not tied to a single model family.

D Limitations

Method. SPARTA adds architectural structure beyond a standard single-rank adapter, including a dual-branch decomposition, a dynamic router, and stochastic restoration. Although the main experiments show a favorable efficiency–performance trade-off at the tested scales, this added structure may become harder to optimize or deploy as the model size and task horizon grow. In particular, the current stochastic restoration schedule is heuristic rather than adaptive, leaving room for more principled criteria for when and where recovery should be applied.

Task. Our evaluation focuses on benchmark streams with clear task boundaries and supervised objectives. This setting is appropriate for controlled comparison, but it does not cover task-free continual learning, severe domain shift, or settings where task-specific data is scarce or noisy. The extent to which the same rank-allocation principle transfers to such regimes remains open.

Model Scope. We validate SPARTA on T5, LLaMA, Qwen, and a supplementary RoboBrain2.5-4B study, but we do not test substantially larger models such as 13B or 72B systems. As a result, our claims should be interpreted at the level of the backbones and scales studied here rather than as evidence of universal architecture-agnostic behavior.

E Ethical Considerations and AI Writing Statement

Our method is evaluated on public datasets and does not require storing historical user data for rehearsal. This property makes the approach ap-

Table 10: **Order-level final performance across all benchmarks.** We report averaged final performance (**FP**) for T5-Large, LLaMA2-7B, LLaMA3.1-8B, and Qwen2.5-7B after the full task stream; $*$ marks numbers copied from prior work.

Methods	Standard CL Benchmark (SC)				Long Sequence Benchmark (LS)				TRACE
	Order 1	Order 2	Order 3	Avg	Order 4	Order 5	Order 6	Avg	Order 7
# T5-Large based									
SeqLoRA	72.1	66.8	73.3	70.7	66.4	63.9	19.5	59.9	12.1
LoRAReplay	74.0	73.1	73.0	73.3	74.2	72.7	73.9	73.6	34.0
L2P* (Wang et al., 2021)	60.3	61.7	61.1	60.7	57.5	53.8	56.9	56.1	-
LFPT5* (Qin and Joty, 2021)	67.6	72.6	**77.9**	72.7	70.4	68.2	69.1	69.2	-
ProgPrompt* (Razdaibiedina et al., 2023)	75.2	75.0	75.1	75.1	78.0	**77.7**	77.9	**77.9**	-
IncLoRA	66.5	64.6	66.1	65.7	59.1	60.7	59.4	59.7	-
O-LoRA (Wang et al., 2023b)	73.2	72.4	70.4	72.0	69.9	68.5	65.3	67.9	-
+ MIGU (Du et al., 2024)	73.5	71.4	70.0	71.6	65.4	65.2	65.2	65.3	-
SAPT-LoRA* (Zhao et al., 2024)	-	-	-	-	**83.4**	-	**80.6**	-	-
SPARTA (*ours*)	73.7	70.5	73.8	72.7	71.5	70.5	68.0	70.0	16.7
+ Replay	**77.0**	**75.6**	75.2	**75.9**	75.6	73.2	74.1	74.3	**36.5**
# LLaMA2-7B based									
SeqLoRA	73.0	73.2	78.4	74.9	74.7	73.7	72.5	73.7	64.1
LoRAReplay	80.3	80.4	76.7	79.2	80.3	79.5	80.5	80.0	71.9
O-LoRA (Wang et al., 2023b)	76.2	76.3	76.8	76.4	68.5	67.8	67.5	67.9	35.0
SPARTA (*ours*)	79.5	79.9	**80.0**	79.8	76.6	77.0	77.2	76.9	66.0
+ Replay	**80.4**	**81.3**	78.4	**80.0**	**83.2**	**82.5**	**81.8**	**81.8**	**73.1**
# LLaMA3.1-8B based									
SeqLoRA	79.9	79.0	80.0	79.6	74.2	73.7	76.5	74.8	65.1
LoRAReplay	80.1	80.6	80.1	80.3	**83.2**	80.7	**82.2**	82.0	78.7
O-LoRA (Wang et al., 2023b)	71.8	72.2	72.8	72.3	73.1	69.4	71.6	71.4	36.7
SPARTA (*ours*)	**81.4**	80.7	80.5	**80.9**	80.7	77.7	81.5	80.0	72.7
+ Replay	80.6	**81.0**	**80.7**	80.8	83.1	**81.7**	81.8	**82.2**	**80.1**
# Qwen2.5-7B based									
SeqLoRA	80.0	77.9	78.4	78.8	79.5	79.1	81.1	79.9	65.1
LoRAReplay	**80.7**	**80.6**	**80.1**	**80.5**	83.3	**83.2**	82.7	83.1	75.7
SPARTA (*ours*)	79.8	79.1	79.4	79.4	79.8	80.2	81.5	80.5	70.4
+ Replay	80.3	**80.6**	79.9	80.3	**83.7**	82.9	**82.9**	**83.2**	**77.3**

pealing for privacy-sensitive continual-learning settings, although broader deployment risks inherited from the underlying LLMs still remain.

AI tools were used only for surface-level language polishing, such as spelling correction and phrasing refinement. The technical content, experimental design, and scientific claims were authored and verified by the human authors.

Table 11: **Continual-learning results on RoboBrain2.5-4B.** We compare SPARTA with a single-rank LoRA baseline across three task orders, reporting **AP**, **FP**, and **Forget**.

Method	Order 1			Order 2			Order 3		
	AP↑	**FP↑**	**Forget↓**	**AP↑**	**FP↑**	**Forget↓**	**AP↑**	**FP↑**	**Forget↓**
LoRA	80.0	79.4	0.7	80.3	77.5	2.8	80.2	79.8	0.4
SPARTA	80.8	80.6	0.2	80.6	78.0	2.6	80.3	80.0	0.3