DATA: Decomposed Attention-based Task Adaptation for Rehearsal-Free Continual Learning

Anonymous ACL submission

Abstract

Continual learning (CL) is essential for Large Language Models (LLMs) to adapt to evolving real-world demands, yet they are susceptible to catastrophic forgetting (CF). While traditional CF solutions rely on expensive data rehearsal, recent rehearsal-free methods employ model-based and regularization-based strategies to address this issue. However, these approaches often neglect the model's plasticity, which is crucial to achieving optimal performance on newly learned tasks. Consequently, a key challenge in CL is striking a balance between preserving plasticity and mitigating CF. To tackle this challenge, we propose the **D**ecomposed **A**ttention-based **T**ask **A**daptation (DATA), which explicitly decouples and learns both task-specific and task-shared knowledge using high-rank and low-rank task adapters (e.g., LoRAs). For new tasks, DATA dynamically adjusts the weights of adapters of different ranks based on their relevance and distinction from previous tasks, allowing the model to acquire new task-specific skills while effectively retaining previously learned knowledge. Specifically, we implement a decomposed component weighting strategy comprising learnable components that collectively generate attention-based weights, allowing the model to integrate and utilize diverse knowledge from each DATA. Extensive experiments on three widely used benchmarks demonstrate that our proposed method achieves state-of-the-art performance. Notably, our approach significantly enhances model plasticity and mitigates CF by extending learnable components and employing stochastic restoration during training iterations. Our code will be available at https://anonymous.4open.science/r/DATA1.

1 Introduction

Continual learning (CL) (Wang et al., 2023a; Zhou et al., 2024) is essential for Large Language Models (LLMs) (Dubey et al., 2024; Yang et al., 2024)
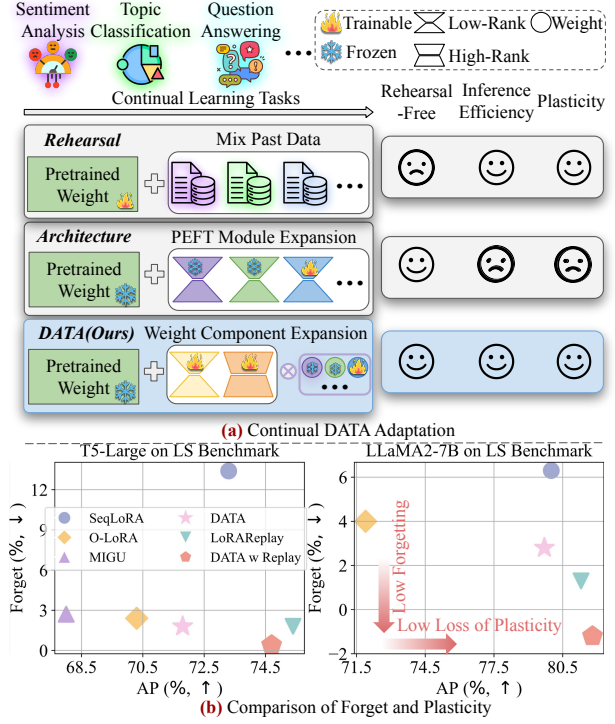

Figure 1: (a) Comparisons of previous CL methods with **DATA**. *Rehearsal-free* indicates that the methods do not require storing sample features from previous tasks. *Inference Efficiency* denotes the computational efficiency during the inference phase. *Plasticity* is the ability to adapt effectively to new tasks. (b) Comparison of **AP** and **Forget** (Sec. 4.1) across different CL methods.

to continuously evole, adapting to real-world demands and progressively improve regarding a series of new tasks. Unlike traditional supervised learning, which relies on independent and identically distributed (i.i.d.) data, CL focuses on the dynamic and evolving demands of real-world applications (Zhai et al., 2023; Wu et al., 2024), such as domain-specific adaptations (Roziere et al., 2023) and alignment with human preferences (Ouyang et al., 2022). Departing from the i.i.d. assumption presents two major challenges: 1) **Catastrophic Forgetting** (CF) — the tendency of a model to over-

write previously learned knowledge when acquiring new tasks (McCloskey and Cohen, 1989). 2) **Loss of Plasticity**—the model's diminishing ability to adapt effectively to distribution shifts over time (Dohare et al., 2021).

As illustrated in Figure 1 (a), an ideal CL method leveraging LLMs should possess three essential properties. A commonly used strategy in CL is the preservation (Wang et al., 2024b) or synthesis (Sun et al., 2019) of past training data, referred to as **rehearsal** (or *replay*). However, this approach presents notable challenges, particularly in scenarios involving *sensitive user data* where long-term storage is impractical (Touvron et al., 2023). Furthermore, rehearsal methods require *increased storage capacity and computational resources* during the learning process (Scialom et al., 2022). Architecture-based methods (Wang et al., 2024a) aim to mitigate interference between new and previously learned tasks by dynamically expanding the model's capacity or isolating existing model weights. Nevertheless, these approaches often necessitate training separate expert models for each task, which in turn requires selecting (Zhao et al., 2024) or combining (Wang et al., 2023b) modules during testing. This can limit their ability to generalize effectively to long sequence tasks. Moreover, as shown in Figure 1 (b), these methods mitigate CF by *sacrificing new task accuracy* (i.e., insufficient *plasticity*), prompting us to explore innovative perspectives for enhancing CL capacity.

To address these limitations while maintaining plasticity and mitigating CF, we propose a **D**ecomposed **A**ttention-based **T**ask **A**daptation (DATA), as depicted in Figure 1 (a), which explicitly manages high-rank and low-rank adapters for task-specific and task-shared knowledge during the CL. For new tasks, DATA dynamically adjusts the weights of different ranks adapters based on their relevance and distinction from previous tasks, enabling the model to acquire new task-specific skills while preserving previously learned capabilities. To extract knowledge across tasks more effectively, we introduce a *decomposed component weighting* strategy comprising learnable components that jointly generate attention-based weights, allowing dynamic fusion of knowledge from each rank DATA. During inference, low-rank and high-rank DATA can be reparameterized and projected into LLMs without increasing parameters, preserving the model plasticity (Liu et al., 2023). We use *stochastic restoration* (Wang et al., 2022; Dohare et al., 2024) technique to further protect source knowledge and mitigate CF.

We conduct extensive experiments to evaluate DATA on Standard CL Benchmark (Zhang et al., 2015), Long Sequence Benchmark (Razdaibiedina et al., 2023) and TRACE (Wang et al., 2023c) across T5 model (Raffel et al., 2019), Llama2 (Touvron et al., 2023), Llama3.1 (Dubey et al., 2024) and Qwen2.5 (Yang et al., 2024). DATA achieves state-of-the-art performance on both public benchmarks and in generalizing to unseen tasks. Notably, DATA can be integrated with existing mainstream methods to further enhance the capabilities of CL. In summary, our contributions are as follows:

- We propose a method DATA for rehearsal-free CL of LLMs that balances plasticity and mitigates CF by explicitly managing both *task-specific* and *task-shared* knowledge.

- We explore different task representations of high- and low-rank adapters, decomposing knowledge into shared and specific components. This approach allows for better modeling of the relevance and uniqueness of new tasks concerning previous ones.

- To address various distribution shifts for each target sample, we introduce a *decomposed component weighting* strategy for DATA, which dynamically fuses knowledge from low-rank and high-rank adapters, enhancing their task representations.

- Experiments on three benchmarks show that our approach outperforms most state-of-the-art techniques, significantly alleviating CF and improving model plasticity.

2 Related Work

Continual Learning (CL). CL aims to effectively acquire new task knowledge during ongoing training while preserving knowledge from previously learned tasks. Traditional CL methods can be broadly classified into three categories: *rehearsal-based*, *regularization-based*, and *architecture-based* approaches. Rehearsal-based methods mitigate catastrophic forgetting (CF) by selectively retaining samples (Tiwari et al., 2021; Wang et al., 2024b; He et al., 2024) or pseudo-generative examples (Shin et al., 2017; Sun et al., 2019) from prior tasks. In contrast, regularization-based methods employ quadratic regularization

terms to constrain the update of weights critical to previous tasks (Zhu et al., 2024; Du et al., 2024), thereby stabilizing learned knowledge. Architecture-based methods introduce extra task-specific parameters for each new task (Wang et al., 2023b; Zhao et al., 2024), ensuring that previously acquired knowledge remains intact. These methods typically rely on access to old task data or involve discrete component learning, which presents significant challenges for the CL. In contrast, DATA leverages representations from diverse rank spaces and harnesses the inherent capabilities of LLMs to enable more effective CL (Wang et al., 2022).

Parameter Efficient Fine-Tuning (PEFT). LLMs have shown remarkable adaptability across a wide range of downstream tasks. However, traditional full fine-tuning for each task can incur *significant computational and storage overheads*, often resulting in overfitting (Wu et al., 2025). To mitigate these challenges, researchers have increasingly turned to PEFT methods, which can achieve comparable or even superior performance and generalization by fine-tuning only a small subset of parameters. For instance, *adapters* (Houlsby et al., 2019) incorporate additional modules into various layers of the Transformer architecture, while *prompt-tuning* (Qin and Eisner, 2021) and *prefix-tuning* (Li and Liang, 2021) introduce learnable soft tokens at different layers of the Transformer input. *Low-rank adaptation* (LoRA) (Hu et al., 2022) inserts low-rank branches into pre-trained weights and fine-tunes only those branches. Although these common PEFT techniques offer certain advantages, they are generally confined to static single-task or multi-task learning scenarios. In contrast, DATA leveraging the effectiveness and efficiency of LoRA enables continual adaptation by dynamically combining LoRAs of two different ranks (Liu et al., 2023) utilizing attention-based weights.

3 Methodology

Preliminary. Continual learning (CL) seeks to tackle challenges within ongoing sequences, specifically addressing how to adapt to new tasks without forgetting previously learned knowledge. Formally, suppose there are $\mathcal{N}$ sequential tasks $\{\mathcal{T}_1, \mathcal{T}_2, ..., \mathcal{T}_{\mathcal{N}}\}$, where each task $\mathcal{T}_i = \{\boldsymbol{x}_n^i, \boldsymbol{y}_n^i\}_{n=1}^{\mathcal{N}_i}$ consists of $\mathcal{N}_i$ training examples, with $\boldsymbol{x}_n^i$ representing the input and $\boldsymbol{y}_n^i$ is corresponding label. Let $\mathcal{L}_i(\cdot)$ denote the empirical risk on the i-th task $\mathcal{T}_i$, and f_θ represent the model with

parameters θ. The goal is for the model to perform well on both the current task $\mathcal{T}_j$ and all previously learned tasks $\mathcal{T}_i$, where $i \in \{1, 2, \dots, j-1\}$. The objective function of CL can be expressed as:

$$\mathcal{L}_{\text{CL}}(\theta) = \frac{1}{j} \sum_{i=1}^{j} \mathcal{L}_i(\theta) \qquad (1)$$

where j is the index of the current training task.

3.1 Motivation

Recent advances have highlighted the effectiveness of using adapters with varying ranks to model different tasks, offering a promising solution to the challenges of CL. This motivates us to further explore and validate the core principles behind utilizing both low-rank and high-rank adapters in CL.

To further explore this phenomenon, we conduct a t-SNE distribution analysis (van der Maaten and Hinton, 2008) to examine the feature distributions of Order1 (4 tasks, Sec. 4.1) within the 11th transformer block of LLaMA2-7B. The results as presented in Figure 2 (a) reveal that low-rank adapters maintain relatively consistent feature distributions across different target tasks. This consistency suggests that the low-rank embedding space effectively mitigates the effects of dynamic distribution shifts by concentrating on extracting task-shared knowledge. Conversely, high-rank adapters display noticeable variability across tasks. This variability underscores the enhanced ability of high-rank adapters to aggregate features within a single task, highlighting their suitability for capturing the unique data distribution of each target task.

Furthermore, we utilize the *H-divergence* metric to assess the representation consistency of adapters on Order 4 (15 tasks). A relatively small inter-task divergence indicates that feature representations remain stable and are less affected by cross-task shifts (Ganin et al., 2015), while a small intra-task divergence within a given task indicates that the model better understands the current distribution. We compare the divergence values obtained from the source model alone, injecting low-rank adapters and injecting high-rank adapters, as shown in Figure 2). Compared to the source model and high-rank adapters, low-rank adapters produce feature representations with lower inter-task divergence, particularly when addressing mid-range target tasks or significant task shifts between adjacent tasks (i.e., tasks 6-10). These findings underscore the efficacy of low-rank adapters in acquiring long-term,

(a) Feature representations of different branches   **(b)** Divergence of different branches
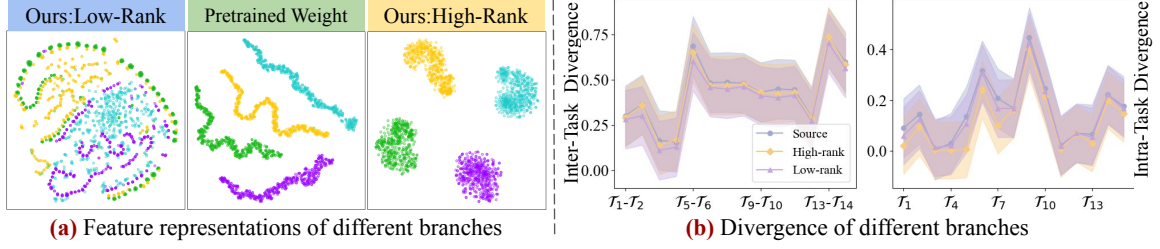
Figure 2: (a) We perform a t-SNE distribution analysis of different adapter representations on Order 1(4 tasks). The low-rank branch shows a consistent distribution across the target tasks and the high-rank branch exhibits substantial distribution differences across the target tasks. (b) We calculate the divergence of different branches in Order 4 (15 tasks). In comparison to the source model, low-rank adapters effectively alleviates inter-task divergence across all 14 task transitions, while the high-rank adapters significantly enhances intra-task feature aggregation.

task-shared knowledge during the CL. Furthermore, high-rank adapters effectively reduce intra-task divergence across nearly all tasks, indicating their superior capability in adapting to the current task distribution and extracting task-specific knowledge in consecutive target tasks.

In summary, the structure of low-rank DATA reduces feature redundancy, placing the model in an underfitting state and enhancing its plasticity. This design enables the model to acquire general information across consecutive target tasks and effectively extract task-shared knowledge. In contrast, high-rank DATA aligns more closely with the target data distribution, allowing it to focus on learning task-specific knowledge and mitigating CF.

3.2 High- and Low-Rank Adapter

The above observations prompted us to introduce high- and low-rank Adapters into the model, aiming to simultaneously adapt current task distribution while maintaining task-shared knowledge.

Similar to LoRA, DATA's design principle is simple and effective, as illustrated in Figure 3 (b). The architecture consists of three sub-branches: the central branch, which originates from the original model's linear layer, and the right and left branches, which are bottleneck structures representing high-rank and low-rank DATA, respectively. Specifically, the right branch (high-rank) consists of an up-projection layer with parameters $\mathbf{A}_2 \in \mathbb{R}^{d \times d_h}$, and a down-projection layer with parameters $\mathbf{B}_2 \in \mathbb{R}^{d_h \times d}$, where d_h (e.g., $d_h = 32$) represents the middle dimension of the high-rank feature. The same principle applies to the low-rank branch, except it satisfies $d_l < d_h$. For an input $\boldsymbol{x}$, the resulting high-rank features $\boldsymbol{f}_h$ and low-rank features $\boldsymbol{f}_l$ from the DATA are formulated as:

$$\boldsymbol{f}_h = \mathbf{A}_2 \cdot (\mathbf{B}_2 \cdot \boldsymbol{x}); \boldsymbol{f}_l = \mathbf{A}_1 \cdot (\mathbf{B}_1 \cdot \boldsymbol{x}) \quad (2)$$

The two-branch bottleneck is connected to the output feature of the original linear $\boldsymbol{f}_o$ through a residual connection, with scale weights $\boldsymbol{\lambda}_h$ and $\boldsymbol{\lambda}_l$. The fusion of knowledge $\boldsymbol{f_x}$ is then expressed as:

$$\boldsymbol{f_x} = \boldsymbol{f}_o + \boldsymbol{\lambda}_h \times \boldsymbol{f}_h + \boldsymbol{\lambda}_l \times \boldsymbol{f}_l \quad (3)$$

The task knowledge scale weights $\boldsymbol{\lambda}_h$ and $\boldsymbol{\lambda}_l$ can be adaptively computed through *decomposed component weighting* strategy as detailed in Sec. 3.3. During the inference process, the distinct task representations of the DATA are reparameterized and projected into the original model, enabling dynamic knowledge adjustments.

3.3 Decomposed Component Weighting

To effectively mitigate CF across various tasks and samples, it is crucial to extract and manage different types of knowledge. While the distinct structure of low-rank and high-rank DATA facilitates learning diverse task representations, the continual adaptation process also necessitates the normalization of knowledge fusion weights. This normalization ensures the effective capture of relevant task-specific knowledge while preserving long-term task-specific knowledge.

As illustrated in Figure 3 (a), we draw inspiration from (Smith et al., 2022) and propose a *decomposed component weighting* strategy. Specifically, we introduce a set of weight components that form a decomposed weight through weighted summation, which is then passed to the corresponding DATA layer. Furthermore, in new tasks, the weights will essentially reuse previously acquired knowledge about past tasks instead of initializing new task weights from scratch:

$$\boldsymbol{\lambda} = \sum_m \boldsymbol{\alpha}_m \mathbf{W}_m \quad (4)$$
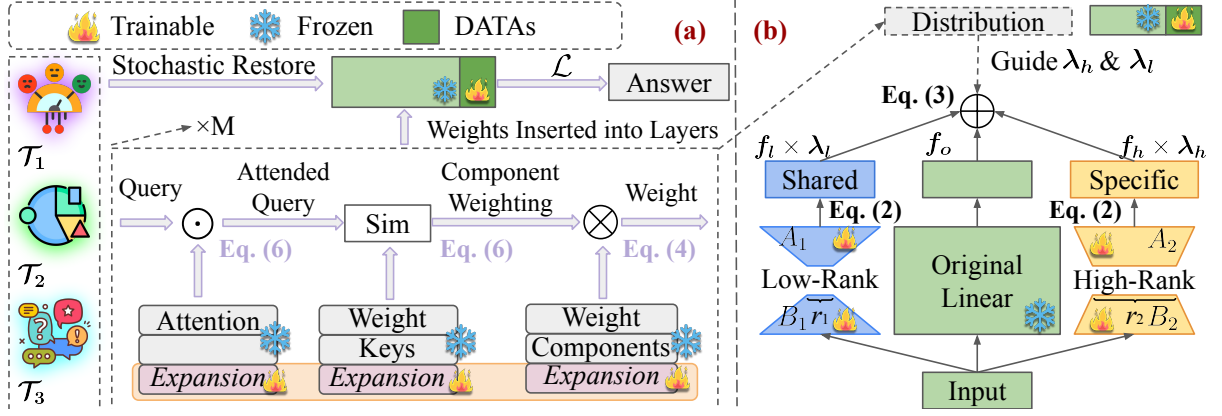
4

Figure 3: **Overview of the DATA framework.** (a) We introduce a novel *decomposed component weighting* strategy for generating **attention-based weights**, parameterized by a set of *extended* weight components, each associated with a corresponding key and attention vector. Only the weighting and adapter parameters are trainable which is *parameter efficient* and no training data is stored for the replay which is *memory-efficient* and *privacy-preserving*. (b) We integrate low-rank and high-rank DATA into the linear layers of the pre-trained model guided by the generated weights, allowing for the dynamic fusion of knowledge from each DATA with different task representations.

where $W \in \mathbb{R}^{L_w \times d \times M}$ represents our set of weight components, with M denoting the length of the set (i.e., the extra axis of additional capacity), L_w is the weight length (chosen as a hyperparameter) and $\boldsymbol{\alpha}$ is a *weighting vector* that determines the contribution of each component, which is the cosine similarity γ between a query $q(\boldsymbol{x})$ and keys:

$$\boldsymbol{\alpha} = \gamma(q(\boldsymbol{x}), \mathbf{K}) \quad (5)$$

where $\mathbf{K} \in \mathbb{R}^{d \times M} = \{\mathbf{K}_1, \mathbf{K}_2, \ldots, \mathbf{K}_M\}$ contains keys corresponding to weight components. The intuition behind this is that the contribution of each weight component $\mathbf{W}_m$ to the final weight $\boldsymbol{\lambda}$ is weighted according to the similarity.

To address the challenge of query matching, we introduce an additional component: *attention*. Alongside $\mathbf{K}_m$, each weight component $\mathbf{W}_m$ is paired with a corresponding attention vector $\mathbf{A}_m$, enabling the query to focus on specific features within the high-dimensional query. We utilize a simple feature selection attention mechanism, where the query vector is element-wise multiplied by the attention vector to generate an *attended query* which is then used to compute the similarity with the key. Specifically, our updated approach to producing the weighting vector is as follows:

$$\boldsymbol{\alpha} = \gamma(q(\boldsymbol{x}) \odot \mathbf{A}, \mathbf{K}) \quad (6)$$

where $\mathbf{A} \in \mathbb{R}^{d \times M} = \{\mathbf{A}_1, \mathbf{A}_2, \ldots, \mathbf{A}_M\}$ represents the learnable attention vectors corresponding to each weight component, and $\odot$ denotes the element-wise (Hadamard) product. It is important to highlight that our attention vectors act

as learnable feature weightings rather than input-conditioned modules. We observe that this fixed representation with its simple design is less susceptible to forgetting, similar to the behavior of our weight component keys.

3.4 Expansion & Orthogonality

The key to alleviating CF is to prevent overwriting the knowledge acquired in previous tasks. When encountering a new task, we freeze the existing components and expand the collection by updating only the new components. This is illustrated at the bottom of Figure 3, where the existing parameters are *frozen*, and only the newly added parameters are *trainable*. Specifically, for task $\mathcal{T}_t$, we learn $\frac{M}{N}$ components, where N denotes the number of tasks and M is the hyperparameter, while the previously learned $\frac{(t-1) \cdot M}{N}$ components are kept frozen. This extension is achieved through our attention-based component weighting scheme, ensuring that the expansion of parameters does not affect the computation of the weights $\boldsymbol{\alpha}$ corresponding to the previously learned components.

To further mitigate CF, we introduce orthogonality constraints for $\mathbf{W}$, $\mathbf{K}$, and $\mathbf{A}$. The underlying intuition is that interference is minimized when vectors are orthogonal. For instance, we aim to prevent the keys and weights learned in task $\mathcal{T}_2$ from influencing data from task $\mathcal{T}_1$. To achieve this, we initialize the vectors orthogonally and incorporate a simple orthogonal penalty loss as:

$$\mathcal{L}_{\text{ortho}}(B) = \|BB^\top - I\|_2^2 \quad (7)$$

Methods	Standard CL Benchmark (SC)			Long Sequence Benchmark (LS)			TRACE		
	FP ↑	AP ↑	Forget ↓	FP ↑	AP ↑	Forget ↓	FP ↑	AP ↑	Forget ↓
T5-Large									
SeqLoRA	$70.7_{\pm.39}$	$\mathbf{76.7}_{\pm.43}$	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	$73.6_{\pm.36}$	$\mathbf{75.4}_{\pm.59}$	1.8	$34.0_{\pm.62}$	$\mathbf{46.8}_{\pm.63}$	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	-	-	
DATA (*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	$\mathbf{73.9}_{\pm.24}$	$74.5_{\pm.75}$	**0.6**	$\mathbf{74.3}_{\pm.35}$	$74.7_{\pm.91}$	**0.4**	$36.5_{\pm.32}$	$45.2_{\pm.61}$	**8.7**
LLaMA-2-7B									
SeqLoRA	$74.9_{\pm.42}$	$\mathbf{80.7}_{\pm.38}$	5.8	$73.7_{\pm.66}$	$80.0_{\pm.65}$	6.3	$64.1_{\pm.49}$	$77.9_{\pm.54}$	13.8
LoRAReplay	$79.2_{\pm.53}$	$\mathbf{80.7}_{\pm.61}$	1.5	$80.0_{\pm.45}$	$\mathbf{81.3}_{\pm.57}$	1.3	$71.9_{\pm.62}$	$\mathbf{78.6}_{\pm.75}$	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
DATA (*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	$\mathbf{80.0}_{\pm.24}$	$80.4_{\pm.45}$	**0.4**	$\mathbf{81.8}_{\pm.61}$	$80.6_{\pm.84}$	**-1.2**	$\mathbf{73.1}_{\pm.46}$	$77.5_{\pm.98}$	**4.4**
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	$80.3_{\pm.71}$	$\mathbf{80.9}_{\pm.56}$	0.6	$82.0_{\pm.69}$	$\mathbf{85.0}_{\pm.75}$	3.0	$\mathbf{78.7}_{\pm.83}$	$\mathbf{85.7}_{\pm.68}$	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
DATA (*ours*)	$\mathbf{80.9}_{\pm.40}$	$80.6_{\pm.35}$	-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**	$\mathbf{82.2}_{\pm.66}$	$82.6_{\pm.77}$	**0.4**	$77.6_{\pm.37}$	$81.0_{\pm.72}$	**3.4**

Table 1: Performance of baselines and ours **DATA** 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. We report the mean and standard deviation of results with 3 different runs.

where B is an arbitrary matrix, I is identity matrix, and $\|\cdot\|_2$ denotes Frobenius norm, which measures the squared sum of all entries of the matrix.

Additionally, we propose a stochastic restoration method to restore knowledge from the source pre-trained model, thereby enhancing plasticity. The update of the weight W at step t is as follows:

$$Mask \sim \text{Bernoulli}(p) \quad (8)$$

$$W_{t+1} = Mask \odot W_0 + (1 - Mask) \odot W_{t+1} \quad (9)$$

where p is a small restoration probability, and $Mask$ is a mask tensor of the same shape as W_{t+1}. The mask determines which elements within W_{t+1} will be restored to the source weights W_0.

3.5 Full Optimization

Combining all objectives, our full optimization is:

$$\min_{\mathbf{W}_n, \mathbf{K}_n, \mathbf{A}_n} \mathcal{L}_{\text{CL}}(f_{\theta, \mathbf{W}, \mathbf{K}, \mathbf{A}, \boldsymbol{\lambda}_h, \boldsymbol{\lambda}_l}(\boldsymbol{x}), \boldsymbol{y}) + \\ \beta(\mathcal{L}_{\text{ortho}}(\mathbf{W}) + \mathcal{L}_{\text{ortho}}(\mathbf{K}) + \mathcal{L}_{\text{ortho}}(\mathbf{A})) \quad (10)$$

where $\mathbf{W}_n, \mathbf{K}_n, \mathbf{A}_n$ refer to the weight components and corresponding keys/attention vectors that are unfrozen and trained during task $\mathcal{T}_n$ and β is a hyperparameter balancing the orthogonality loss.

4 Experiment

In Section 4.4, we compare our method with other SOTA methods on three public benchmarks. In Section 4.6, we further evaluate the task generalization ability of the proposed method on unseen target tasks. Comprehensive ablation studies are conducted in Section 4.7.

4.1 Datasets

Standard CL Benchmark (SC) is a CL benchmark for language models, which consists of five text classification datasets introduced by (Zhang et al., 2015). We follow Wang et al. (2023b) to pick four datasets (AG News, Amazon reviews, DBpedia and Yahoo Answers) and shuffle the tasks into three different orders to form orders 1, 2, and 3.

Long Sequence Benchmark (LS) is an extended version of the standard CL benchmark with 15 datasets (five classification tasks, nine GLUE and SuperGLUE tasks, and the IMDB dataset) (Razdaibiedina et al., 2023). Following Razdaibiedina et al. (2023), we select 1,000 random samples for training each task and hold out 500 samples per class for validation and testing. Similarly, we shuffle them to form orders 4, 5, and 6.

TRACE is a CL benchmark for LLMs that includes 8 datasets that cover multichoice QA, multilingual capabilities, code generation, and mathematical reasoning (Wang et al., 2023c).

Metrics. We adopt the following three 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 differ-
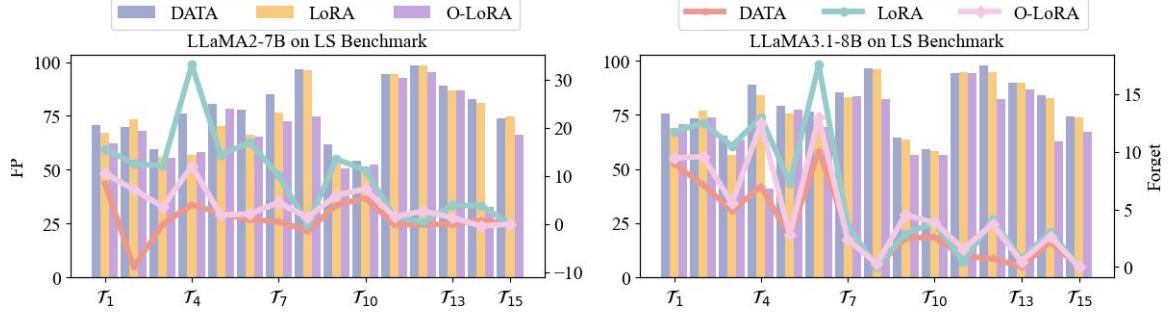
Figure 4: The shifts in CL methods with **FP** and **Forget** on LS Order 4. DATA prevents the shift (blue bar) and thus mitigates forgetting (orange line).

ence between **AP** and **FP**, as commonly used in previous studies (Wu et al., 2022; Jiang et al., 2025) to quantify forgetting. More detailed information on the datasets about orders and evaluation metrics is presented in the Appendix B.1.

4.2 Baselines

We compare our method with the following baselines: 1) **SeqLoRA**: Trains fixed-size LoRA on a sequence of tasks. 2) **LoRAReplay**: Trains new tasks on LoRA with mixing a 2% past task. 3) **O-LoRA** (Wang et al., 2023b): Learns tasks in different LoRA subspaces that are kept orthogonal to each other and sums all LoRA weights up at testing time. 4) **MIGU** (Du et al., 2024): Only updates the model parameters with large magnitudes.

4.3 Implementations

We implement DATA with the LLaMA3.1-8B (Dubey et al., 2024), LLaMA2-7B (Touvron et al., 2023), Qwen2.5-7B (Yang et al., 2024) and T5-Large (Raffel et al., 2019). All experiments are conducted on 2 A100 GPUs with 80GB using LLaMA-Factory (Zheng et al., 2024). All experimental results are reported as the average of 3 runs. Please refer to the Appendix B for more detailed settings.

4.4 Main Results

To demonstrate the effectiveness of the proposed DATA method, we perform experiments on three CL benchmarks, as summarized in Table 1. Following O-LoRA (Wang et al., 2023b), we present the results of three independent runs with different task orders on the two previous CL benchmarks. Detailed results for each order and each task within a specific order are provided in Appendix C.

Our DATA significantly reduces the *CF* of abilities and knowledge in CL. Traditional CL approaches often result in relatively high levels of

forgetting. While O-LoRA demonstrates slightly better performance yet still experiences over 12% forgetting on the TRACE benchmark in **FP**. In contrast, DATA consistently and significantly mitigates this issue by reducing forgetting across various models and benchmarks. For instance, compared to O-LoRA, DATA increases **FP** by an average of 14.4 and decreases **Forget** by an average of 2.1 in LLaMA2. Furthermore, compared to replay methods, DATA further increases **FP** by an average of 1.3 and reduces **Forget** by an average of 1.1, highlighting its compatibility with other methods to effectively improve capabilities. DATA provides a viable solution for addressing the problem of commonsense forgetting in existing models.

Our DATA method does not excessively compromise the plasticity needed to learn new tasks. Compared to other CL methods, this technique demonstrates a negligible reduction in the **FP** and **AP** metric particularly in longer sequence tasks (LS, 15 tasks) and more challenging task instruction generalizations (TRACE, 8tasks), signifying a well-maintained balance between plasticity and stability. DATA outperforms O-LoRA in LLaMA2 by an average of 12.3 in **AP**, while the difference with SeqLoRA, known for not restricting parameter updates and theoretically offering the most optimal plasticity, is less than 1. This indicates that DATA is more suitable for practical applications in LLMs.

Longer task sequences and more challenging task instructions contribute to increased forgetting. A comparative analysis of the **Forget** metrics for LS and TRACE with SC reveals a significant rise in forgetting. For example, in the LLaMA3.1-8B model, the decline in performance escalates from 5.8 in SC to 9.0 in LC, and further to 17.3 in TRACE. Furthermore, across all tasks, the forgetting scores **Forget** for both LLaMA2 and LLaMA3 remain below 20, demonstrating that, compared to

7

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
DATA	**64.47**	10.42	**3.63**	16.94	**81.39**

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

	$DATA_h$	$DATA_l$	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 3: Ablation studies on different components.

the T5 model, LLMs exhibit a stronger capacity to mitigate CF (Jiang et al., 2025).

Forgetting in CL is model-dependent. A comparison of **Forget** across different models reveals that stronger foundational models tend to mitigate CF more effectively. For instance, on the LS benchmark, SeqLoRA with LLaMA2-7B exhibits 2.7 less **Forget** than SeqLoRA with LLaMA3.1-8B and 0.8 less **Forget** than LoRAReplay. This suggests that forgetting is influenced not only by the nature of the tasks but also significantly by model-related factors such as size, architecture, and the diversity of pre-training data.

4.5 Performance Shifts

In Figure 4, we visually depict the performance changes observed during the training process of DATA compared to other CL methods. Our proposed method clearly demonstrates its effectiveness in mitigating forgetting and enhancing plasticity. By maintaining a stable structure during the fine-tuning phase, DATA ensures consistent performance on general tasks. In sequential task transitions, it is evident that DATA demonstrates superior **FP** and reduce **Forget** compared to other methods. The line shifts further illustrate that LLaMA3.1-8B exhibits less forgetting than LLaMA2-7B, emphasizing that **forgetting is model-dependent**.

4.6 Task Generalization

We select four benchmarks to evaluate the cross-task generalization capability of DATA, an essential dimension for assessing CL algorithms. As demonstrated in Table 2, DATA effectively balances generalization and CL ability by efficiently extracting task-shared knowledge using low-rank adapters. This finding suggests that actively promoting task-shared knowledge and source capabilities between different tasks is beneficial. However, all methods experience a notable decline in performance on reasoning tasks (GSM8K and BBH), which may be attributed to the simple classification tasks potentially impairing the ability to perform generative

reasoning required for instruction following.

4.7 Ablation Study

We conduct an ablation study in Order 1 using LLaMA2-7B to evaluate the contributions of various components in our method DATA, including high-rank DATA ($DATA_h$), low-rank DATA ($DATA_l$), decomposed component weighting strategy, attention keys, orthogonality regularization (Ortho.), and stochastic restoration (Rest.). As shown in Table 3 (E_2), incorporating high-rank DATA increases the **FP** by 0.7, indicating that high-rank features can effectively extract more task-specific knowledge for adaptation to the target task. In E_3, low-rank DATA improves performance by 2.4 compared to E_1. The overall improvement in E_4 reaches 4.6, suggesting that both types of DATA complement each other during continual adaptation. The stochastic restoration in E_5 allows the model to maintain plasticity in CL, enhancing its adaptability to new tasks. E_6-E_8 achieves an additional 2 improvements, demonstrating the effectiveness of the decomposed component weighting strategy in enhancing the task representations for each DATA.

5 Conclusion

In this paper, we revisit existing methods for leveraging Large Language Models (LLMs) in continual learning (CL) and propose three ideal characteristics for such systems: *rehearsal-free*, *inference efficiency*, and *plasticity*. To address the challenges associated with these characteristics, we introduce Decomposed Attention-based Task Adaptation (DATA) as a solution to catastrophic forgetting and plasticity loss. Moreover, we propose a *decomposed component weighting* strategy to dynamically integrate the knowledge from both low-rank and high-rank DATA, thereby enhancing the unique task representation. Extensive experiments across multiple CL benchmarks and LLMs consistently validate the effectiveness of DATA.

Limitations

Method The DATA method introduces a decomposed component weighting strategy and employs both high-rank and low-rank adapters, which increases the complexity of the model architecture. This complexity may lead to higher computational costs during training and inference, particularly when scaling to larger models or more tasks. Additionally, the need for dynamic weight adjustments based on task relevance and distinction may require more sophisticated optimization techniques, potentially limiting its applicability in resource-constrained environments. Furthermore, our current approach to stochastic recovery involves a step-level method, where a small proportion of parameters is recovered every 200 steps. There is significant potential to enhance this process by exploring dynamically adaptive methods that can more effectively select saturated or less important parameters for recovery. Additionally, establishing criteria to determine when recovery is necessary could optimize the process further, potentially improving model performance and efficiency.

Task Although DATA is designed to be rehearsal-free, it still relies on the availability of diverse and high-quality task-specific data for effective adaptation. In scenarios where task-specific data is scarce or of low quality, the method's ability to adapt and generalize may be compromised. Additionally, the method's performance on tasks with significant domain shifts or out-of-distribution data remains to be fully explored.

Large Language Models The effectiveness of DATA is highly dependent on the underlying LLM architecture. While the method shows promising results on models like LLaMA2, LLaMA3.1, and Qwen2.5, its performance may vary across different LLMs, especially those with significantly different architectures or pre-training objectives. Furthermore, we do not experiment with larger models like 13B and 72B due to computational or financial constraints.

Ethical Considerations

Our approach does not introduce ethical concerns. The datasets we used are public, and there are no privacy issues.

AI writing statement

This paper utilized AI assistance for language polishing of the manuscript, including vocabulary correction and spell checking.

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.

Yaroslav Ganin, E. Ustinova, Hana Ajakan, Pascal Germain, H. Larochelle, François Laviolette, Mario Marchand, and Victor S. Lempitsky. 2015. Domain-adversarial training of neural networks. In *Journal of machine learning research*.

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*.

9

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.

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*.

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*.

Xiang Lisa Li and Percy Liang. 2021. Prefix-tuning: Optimizing continuous prompts for generation. 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)*, pages 4582–4597, Online. Association for Computational Linguistics.

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.

Michael McCloskey and Neal J. Cohen. 1989. Catastrophic interference in connectionist networks: The sequential learning problem. volume 24 of *Psychology of Learning and Motivation*, pages 109–165. Academic Press.

Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, et al. 2022. Training language models to follow instructions with human feedback. *Advances in neural information processing systems*, 35:27730–27744.

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, Junjie Bai, and Soumith Chintala. 2019. *PyTorch: an imperative style, high-performance deep learning library*. Curran Associates Inc., Red Hook, NY, USA.

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.

Guanghui Qin and Jas' Eisner. 2021. Learning how to ask: Querying lms with mixtures of soft prompts. *ArXiv*, abs/2104.06599.

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, et al. 2023. Code llama: Open foundation models for code. *arXiv preprint arXiv:2308.12950*.

Thomas Scialom, Tuhin Chakrabarty, and Smaranda Muresan. 2022. Fine-tuned language models are continual learners. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 6107–6122, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

Hanul Shin, Jung Kwon Lee, Jaehong Kim, and Jiwon Kim. 2017. Continual learning with deep generative replay. In *Neural Information Processing Systems*.

James Smith, Leonid Karlinsky, Vyshnavi Gutta, Paola Cascante-Bonilla, Donghyun Kim, Assaf Arbelle, Rameswar Panda, Rogério Schmidt Feris, and Zsolt Kira. 2022. Coda-prompt: Continual decomposed attention-based prompting for rehearsal-free continual learning. *2023 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 11909–11919.

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*.

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, et al. 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, et al. 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, Mariama Drame, Quentin Lhoest, and Alexander Rush. 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*.

Yichen Wu, Hongming Piao, Long-Kai Huang, Renzhen Wang, Wanhua Li, Hanspeter Pfister, Deyu Meng, Kede Ma, and Ying Wei. 2025. S-loRA: Scalable low-rank adaptation for class incremental learning. In *The Thirteenth International Conference on Learning Representations*.

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, Jin Xu, Jingren Zhou, Jinze Bai, Jinzheng He, Junyang Lin, Kai Dang, Keming Lu, Ke-Yang Chen, Kexin Yang, Mei Li, Min Xue, Na Ni, Pei Zhang, Peng Wang, Ru Peng, Rui Men, Ruize Gao, Runji Lin, Shijie Wang, Shuai Bai, Sinan Tan, Tianhang Zhu, Tianhao Li, Tianyu Liu, Wenbin Ge, Xiaodong Deng, Xiaohuan Zhou, Xingzhang Ren, Xinyu Zhang, Xipin Wei, Xuancheng Ren, Yang Fan, Yang Yao, Yichang Zhang, Yunyang Wan, Yunfei Chu, Zeyu Cui, Zhenru Zhang, and Zhi-Wei Fan. 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*.

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.

11

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*.

Da-Wei Zhou, Qi-Wei Wang, Zhi-Hong Qi, Han-Jia Ye, De-Chuan Zhan, and Ziwei Liu. 2024. Class-Incremental Learning: A Survey . *IEEE Transactions on Pattern Analysis & Machine Intelligence*, 46(12):9851–9873.

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.

A Overall Framework

Drawing from the insight that LoRA (Hu et al., 2022) has exhibited superior performance, we utilize a high- and low-rank framework to ensure stability during continual task adaptation. The overall framework and the details of our method DATA are shown in Figure 3.

LoRA (Low-Rank Adaptation), as proposed by (Hu et al., 2022), postulates that parameter changes (ΔW) during fine-tuning occur within a low-rank subspace. This is particularly applied to the layer weights $W_0 \in \mathbb{R}^{m \times n}$ of a model f_θ for a downstream task. The parameter update is formulated as $\Delta W = A \times B$, where $A \in \mathbb{R}^{m \times r}$ and $B \in \mathbb{R}^{r \times n}$ are two learnable matrices, and the rank r is significantly smaller than $\min\{m, n\}$.

For a specific layer in the model f_θ, the LoRA update is expressed as:

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

Here, h' represents the updated output, and x is the input to the layer. Importantly, the original weights W_0 are kept frozen during the fine-tuning process, and only the matrices A and B are trainable.

B Experimantal Settings

B.1 Datasets

Train Tasks. Tables 4 and 5 provide detailed information on the datasets utilized in our continual learning (CL) experiments. Table 4 presents the 15 datasets included in the Long Sequence Benchmark (Razdaibiedina et al., 2023), while Table 5 outlines the 8 datasets from TRACE (Wang et al., 2023c). Both tables include the corresponding evaluation metrics for each dataset.

Generalization. We select the 1) Multitask Language Understanding (MMLU) (Hendrycks et al., 2021), which includes multiple-choice questions across 57 subjects. 2) GSM8K (Cobbe et al., 2021), which is a high-quality linguistically diverse multi-step elementary math reasoning dataset. 3) BIG-Bench Hard (BBH) (Suzgun et al., 2022), which includes 27 challenging tasks spanning arithmetic, symbolic reasoning, and more, derived from BIG-Bench (BB) (bench authors, 2023). Most of the data consists of multiple-choice questions. 4) AGIEval (Zhong et al., 2023), which includes a wide range of high-quality official entrance exams, qualifying exams, and advanced competitions tailored to human participants.

B.2 Task Sequence Orders

We report task orders used for our CL experiments in Table 6.

B.3 Implementations

Our implementations are based on huggingface transformers v4.45.2 (Wolf et al., 2020) using PyTorch v2.3.1 (Paszke et al., 2019) and LlamaFactory (Zheng et al., 2024). All unseen tasks generalization evaluation conducted using the OpenCompass toolkit (Contributors, 2023), adopting its default configuration.

For Standard CL Benchmark and Long Sequence Benchmark (Order 1 - Order 6), We trained the models with 1 epoch, a constant learning rate of 1e-4.

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

In a series of performance experiments, we configured various parameters as follows: the LoRA rank was set to 8 refer to Figure 5, and the proportion of past task data mixed in LoRAReplay was set to 2%. For the DATA model, the low-rank configuration was set to 2 and the high-rank configuration to 8. From Figure 7, it can be observed that the performance of 2 and 8 is optimal. Additionally, compared to LoRA, the increase in parameters is limited, with only an additional set of LoRA with a rank of 2, achieving a balance between resources and performance.

In terms of the decomposed component weighting strategy, we used a weight length (L_w) of 8. The weight component for each task was set

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

Table 4: The details of 15 classification datasets in the Long Sequence Benchmark (Razdaibiedina et al., 2023). First five tasks correspond to the standard CL benchmark (Zhang et al., 2015).

to $\frac{N_{\text{layer}}}{4}$, leading to a total weight calculation of $M = \frac{N \times N_{\text{layer}}}{4}$, where N_{layer} is the number of model layers and N represents the number of tasks. The hyperparameter β was assigned a value of 10. For stochastic recovery, a simple strategy was applied where a small proportion of parameters was recovered every 200 training steps.

B.4 More Baselines

IncLoRA: Incremental learning of new LoRA parameters for a sequential series of tasks (without adding any regularization or replaying samples from previous tasks).

LFPT5 (Qin and Joty, 2021): Continuously train a soft prompt that simultaneously learns to solve tasks and generate training samples, which are subsequently used in experience replay.

ProgPrompt (Razdaibiedina et al., 2023): Sequentially concatenates previously learned prompts to the current one during training and testing.

SAPT (Zhao et al., 2024): In the SAPT method, a Shared Attentive Learning and Selection Module (SALS) is used to guide training samples through optimal PET blocks for task-specific learning, using a unique instance-level attention mechanism. This process ensures efficient continual learning for large language models.

C Extended Results

C.1 Fine-grained Results for the Main Experiments

We report the results of each task order on the 3 benchmarks in Table 8. Overall, our proposed DATA demonstrates excellent capabilities in addressing CF and Loss of plasticity.

C.2 Supplementary Motivation

In our analysis of LoRA's performance with varying ranks on LLaMA2-7B for Order 1, as shown in Figure 5, we observed that increasing the rank enhances Adaptation Plasticity (AP) for new tasks. However, this improvement comes at the cost of greater forgetting, as it tends to overlook the shared knowledge between tasks. Conversely, lower ranks do not lead to significant changes in forgetting, indicating that they are more effective at capturing the shared knowledge and skills across tasks. This observation suggests the necessity of integrating both high- and low-rank adapters. By doing so, we can better balance the modeling of task-specific and shared knowledge, potentially optimizing both AP and reducing forgetting.

C.3 Efficiency Analysis

In Table 9, we compare the FLOPs, trainable parameters, storage features, and average predict times of various CL methods. LoRA achieves the highest inference efficiency by bypassing module selection and expansion, allowing the learned

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 5: The overview of dataset statistics in TRACE (Wang et al., 2023c). The 'Source' indicates the origin of the context. 'Avg len' denotes the average length, measured in word count for English, German, and code datasets, and in character count for Chinese datasets. 'SARI' is a score that is specific to evaluating simplification tasks.

Benchmark	Order	Task Sequence
Standard CL Becnhmark	1	dbpedia → amazon → yahoo → ag
	2	dbpedia → amazon → ag → yahoo
	3	yahoo → amazon → ag → dbpedia
Long Sequence Becnhmark	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

Table 6: Seven distinct orders of task sequences were employed for the experiments in continual learning. Orders 1-3 align with the Standard CL Benchmarks, as adopted in previous studies (Zhang et al., 2015). Orders 4-6 pertain to the Long Sequence Benchmarks, which encompass a total of 15 tasks (Razdaibiedina et al., 2023). Order 7 refers to the TRACE benchmark, specifically designed for LLMs, and comprises eight datasets (Wang et al., 2023c).

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 7: The performance of DATA using LLaMA3.1-8B with different high and low ranks on the standard CL benchmark.

LoRA weights to be directly integrated with the original model weights during testing. Furthermore, our method utilizes only two sets of high- and low-rank adapters with a small number of parameters for decomposed component weighting and eliminates the need to store sample features. This leads to excellent performance when considering both trainable parameters and storage features.
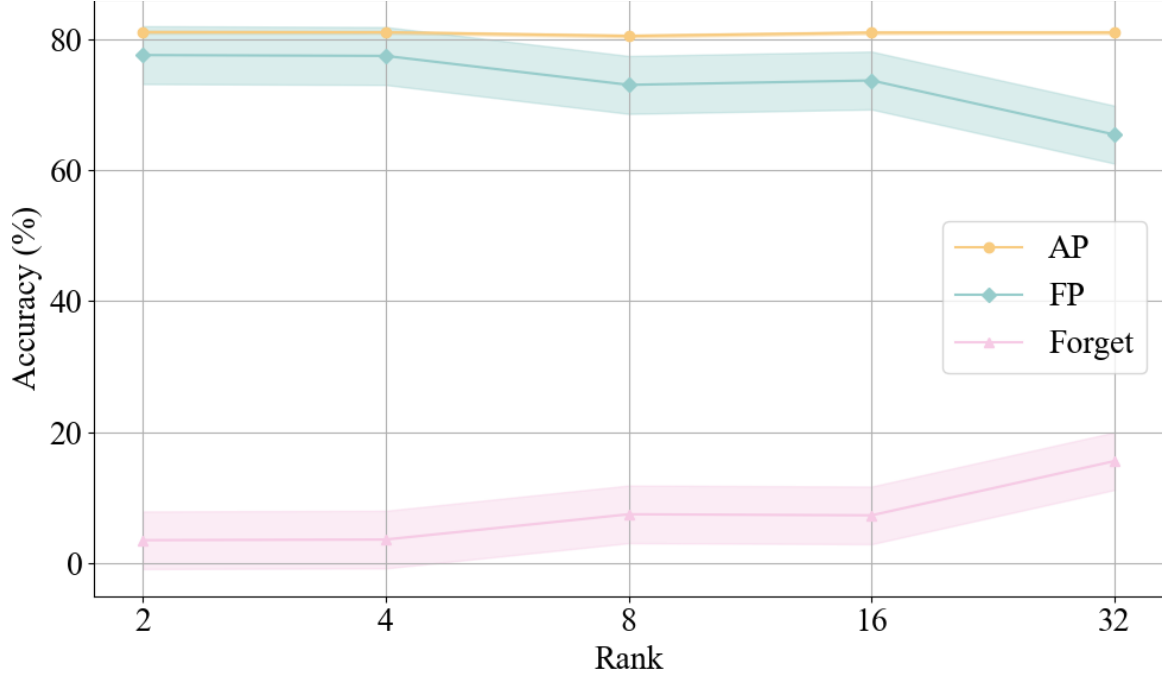
Figure 5: Performance of different LoRA ranks.

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	-
IncLoRA	66.5	64.6	66.1	65.7	59.1	60.7	59.4	59.7	-
ProgPrompt* (Razdaibiedina et al., 2023)	75.2	75.0	75.1	75.1	78.0	**77.7**	77.9	**77.9**	-
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	-	-
DATA (*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
DATA (*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
DATA (*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
DATA (*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**

Table 8: Summary of the results on 3 standard CL benchmarks with T5-Large, LLaMA2-7B, LLaMA3.1-8B and Qwen2.5-7B. Averaged accuracy after training on the last task (**FP**, Sec. 4.1) is reported.

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
DATA	4.6	0.38	0	141

Table 9: Comparison of the number of trainable parameters and FLOPs for Order 4 with LLaMA2-7B.