Continual Gradient Low-Rank Projection Fine-Tuning for LLMs

Anonymous ACL submission

Abstract

Continual fine-tuning of Large Language Models (LLMs) is hampered by the trade-off between efficiency and expressiveness. Low-Rank Adaptation (LoRA) offers efficiency but constrains the model's ability to learn new tasks and transfer knowledge due to its low-rank nature and reliance on explicit parameter constraints. We propose GORP (**G**radient L**O**w **R**ank **P**rojection for Continual Learning), a novel training strategy that overcomes these limitations by synergistically combining full and low-rank parameters and jointly updating within a unified low-rank gradient subspace. GORP expands the optimization space while preserving efficiency and mitigating catastrophic forgetting. Extensive experiments on continual learning benchmarks demonstrate GORP's superior performance compared to existing state-of-the-art approaches.

1 Introduction

Large Language Models (LLMs) have demonstrated remarkable capabilities in areas like in-context learning (Hendel et al., 2023; Liu et al., 2024b) and instruction following (Wei et al., 2022b,a). To adapt these large models to specific downstream tasks, traditional full fine-tuning imposes prohibitive computational costs and memory requirements, which has driven extensive research into parameter-efficient fine-tuning (PEFT) approaches (Houlsby et al., 2019; Hu et al., 2022; Ben Zaken et al., 2022). Low-Rank Adaptation (LoRA) (Hu et al., 2022), in particular, has become a popular PEFT technique, especially in continual learning scenarios (Chitale et al., 2023; Wistuba et al., 2024), due to its efficiency and ability to mitigate catastrophic forgetting (Biderman et al., 2024).

While LoRA significantly reduces training complexity and storage, the low-rank matrices inherently constrain the parameter space and, consequently, the model's expressiveness during optimization (Zhao et al., 2024). This restriction to a low-rank subspace can lead to suboptimal performance compared to full fine-tuning, a gap that often widens in continual learning settings (Xia et al., 2024; Mahla et al., 2025). Furthermore, LoRA updates are intertwined with shared parameter updates, potentially causing collisions in the parameter spaces of different tasks (Wang et al., 2023; Lu et al., 2024). Gradient projection has emerged as a promising mitigation strategy (Saha et al., 2021; Wang et al., 2021; Kong et al., 2022; Saha and Roy, 2023). Common approaches involve calculating the hidden feature space and projecting it onto the orthogonal gradient space of the old task. However, gradient spaces for different tasks are heterogeneous and dynamically evolving. Existing methods that impose explicit constraints (e.g., parameter regularization) on LoRA's low-rank parameters (Wang et al., 2023; Du et al., 2024; Yang et al., 2025) can only approximate the ideal parameter space and fail to adapt dynamically to the changing gradient space of new tasks (Liu et al., 2024a). Moreover, these explicit constraints often struggle to capture shared features across tasks, hindering knowledge transfer.

To address these limitations, we introduce GORP(**G**radient L**O**w **R**ank **P**rojection for Continual Learning), a novel training strategy for continual fine-tuning of LLMs that synergistically integrates full and low-rank parameter updates within a low-rank gradient subspace. GORP effectively balances the *stability-plasticity dilemma* inherent in continual learning (see Table 1 for a comparison with other methods). From a *plasticity* perspective, GORP enhances LoRA by incorporating learnable full-rank parameters for the current task. Crucially, we exploit the observation that gradients tend to adopt a low-rank structure during training (Zhao et al., 2024). Therefore, we project the gradients of these full-rank parameters into a low-rank

Method	Parameters		Parameter Constraints		Gradient Space	
	Full-rank	Low-rank	Explicit	Implicit	Low-rank	Adaptability
O-LoRA (Wang et al., 2023)	✗	✓	✓	✗	✗	Static
MIGU (Du et al., 2024)	✗	✓	✗	✓	✗	Static
N-LoRA (Yang et al., 2025)	✗	✓	✓	✗	✗	Static
GORP(Ours)	✓	✓	✗	✓	✓	Dynamic

Table 1: Comparison of continual fine-tuning methods on training parameters, parameter constraints and Gradient Space Adaptability.

space, maintaining fine-tuning efficiency while significantly expanding the search space for optimal solutions. From a *stability* perspective, GORP departs from prior methods that rely on explicit constraints. Recognizing the limitations of directly sampling subspaces from large-scale models, we leverage the first-order moment of gradients to implicitly capture the dynamic properties of the gradient space. This approach provides a more robust and comprehensive representation of the gradient, reducing computational complexity compared to methods that directly manipulate the hidden feature space (Saha et al., 2021; Zheng et al., 2024a; Qiao et al., 2024). We evaluate GORP on several continual fine-tuning evaluations, demonstrating its superior performance compared to existing state-of-the-art methods. Our results confirm that GORP provides a more effective approach for continual fine-tuning of LLMs.

Our main contributions are summarized as follows:

- We leverage the complementary strengths of full and low-rank parameters by jointly updating them within a unified low-rank gradient subspace. This expands the search space for optimal solutions while retaining the efficiency of low-rank adaptation.

- We utilize the first-order moment of gradients to approximate the hidden feature space, providing a more robust and efficient way to construct a gradient subspace. This mitigates catastrophic forgetting and minimizes computational overhead.

- We introduce GORP, a novel training strategy that effectively balances stability and plasticity in continual learning, outperforming existing methods while maintaining fine-tuning efficiency.

2 Related Works

2.1 Parameter-efficient Fine Tuning of LLMs

Various efficient parameter fine-tuning methods include adapters (Houlsby et al., 2019), Low-Rank Adaptation (LoRA) (Hu et al., 2022), and parameter subset techniques (Ben Zaken et al., 2022). These methods have tackled the challenges including large number of parameters and substantial memory requirements by fine-tuning selective model parameters rather than the entire model. Among these, LoRA has become one of the most widely used methods, which is achieved by freezing pre-trained weights and introducing low-rank trainable matrices, effectively reducing the computational burden. Building on LoRA, Lialin et al. (2023) proposed a series of low-rank aggregation updates for learning network parameters. Xia et al. (2024) employed a residual LoRA module at each fixed step, and eventually merging it with the pre-trained model parameters for chained updates. Hao et al. (2024) used random projection sampling to approximate LoRA, enabling high-rank weight updates, and optimizing memory usage.

2.2 Continual Fine Tuning for LLMs

Three widely used continual learning paradigms (Shi et al., 2024; Lu et al., 2024; Zheng et al., 2024b) for parameter fine-tuning are Replay-based methods (Zhao et al., 2022; Huang et al., 2024), Architecture-based methods (Badola et al., 2023; Song et al., 2023), and Learning-based methods (Farajtabar et al., 2020; Smith et al., 2024), which employ specific optimization strategies or introduce regularization penalties based on the original loss function to balance the trade-off between old and new knowledge. Many studies have demonstrated improved performance through learning-based methods. Qiao et al. (2024) proposed an overarching framework for continual fine-tuning, establishing diverse paradigms for efficient fine-tuning. However, due to the challenges in obtaining gradient spaces and the impracticality of using implicit feature spaces, Wang et al. (2023) suggested leveraging LoRA itself to represent the gradient space, ensuring orthogonality between gradient spaces of different tasks to mitigate forgetting. Subsequently, Du et al. (2024) focused on screening the normalized gradients of the hidden linear layer outputs and updating the selected parameters to minimize gradient conflicts. Yang et al. (2025) introduced parameter sparsification constraints, addressing parameter conflicts between tasks and ensuring that each task's vector space remains independent. Additionally, Lu et al. (2024) and Chen and Garner (2024) employed regularization matrices and introduced further constraints to enhance the ability of LLMs to learn new tasks.

2.3 Continual learning with Gradient Projection

Gradient projection methods in continual learning project the gradient into a subspace of the input's implicit feature space to mitigate catastrophic forgetting when learning new tasks. The Gradient Projection Memory proposed by Saha et al. (2021) leverages the relationship between the input and gradient spaces to form a gradient subspace for each layer, thereby retaining prior knowledge while accommodating new information. However, the gradient space can impose restrictive constraints on the optimization space for new tasks, potentially limiting their learning performance. To facilitate both forward and backward knowledge transfer, Lin et al. (2022c)(2022b) proposed a scaling matrix based on the similarity between new and previous tasks, using the frozen weights from the old task to scale and update the current task's weights. In response to the continuous expansion of the gradient subspace, Liang and Li (2023) introduced the dual gradient projection memory method, which reduces memory consumption and adaptively expands the dimensionality of the layer, enhancing the model's plasticity for new tasks. Other studies (Kong et al., 2022; Wang et al., 2021; Lin et al., 2022a) also improved continual learning performance by refining the gradient space.

3 Gradient Low Rank Projection

We introduce GORP, a novel training strategy that combines full and low-rank parameters with low-rank gradient updates to strike a balance between plasticity and stability. The framework, illustrated

Algorithm 1: Algorithm for GORP

Input : Old task weight W, gradient G_t, step t, rank r, scale factor α, decay rates β_1, β_2, learning rate η, subspace change frequency T, num steps N.

Output : New task weight W

Initialize gradient subspace $\mathcal{S} \leftarrow [\,]$
Initialize first-order moment $M_t \leftarrow 0$
Initialize second-order moment $V_t \leftarrow 0$
Initialize step $t \leftarrow 1$
while $t \leq N$ **do**
  if *Full-rank Parameters* **then**
    if $t \bmod T = 0$ **then** `// via Equation 6`
      $USV \leftarrow \mathrm{SVD}(G_t)$
      $G_t' \leftarrow U_r^\top G_t$
    else
      $G_t' \rightarrow G_{t-1}'$
    end
  end
  if *LoRA Parameters* **then**
    $G_t' \leftarrow G_t$
  end
  $P_t \leftarrow \texttt{Project}(G_t')$ `// via Equation 7`
  $M_t \leftarrow \beta_1 M_{t-1} + (1-\beta_1) P_t$
  $V_t \leftarrow \beta_2 V_{t-1} + (1-\beta_2) P_t^2$
  $P_t' \leftarrow M_t / \sqrt{V_t + \epsilon}$
  $W_t \leftarrow W_{t-1} + \eta \cdot \alpha U_r P_t'$
end
Update $\mathcal{S}$ with M_t `// via Equation 2,3,4`
return *New task weight* W

in Figure 1, consists of two main components: (1) the Gradient Shared Space Construction, which employs low-rank moment with distinct parameters to construct a shared gradient space, and (2) the Low-Rank Projection Optimization, which projects the gradient space of both full and low-rank parameters. The pseudo-code of our method is provided in Algorithm 1.

3.1 Gradient Shared Space Construction

In this section, we construct a gradient shared space. A common approach for building gradient spaces in continual learning is to randomly sample from hidden layer input features. However, for LLMs trained on vast amounts of data, the limited number

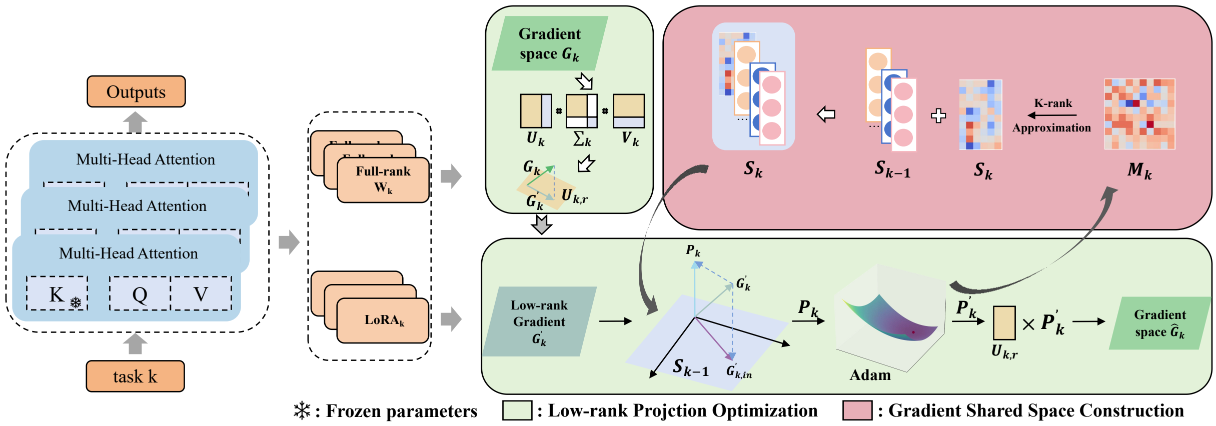

Figure 1: The framework of our Gradient Low Rank Projection (GORP) method. During k-th task training, we reduce the dimensions of full-rank parameters and project both full and low-rank parameters into the space $\mathcal{S}_{k-1}$. Then, we use the first-order moment M_k and a k-rank approximation to construct the Gradient Shared Space $\mathcal{S}_k$.

of sampled features may fail to accurately represent the overall data distribution. Consequently, the resulting gradients may not align with the overall gradient direction during gradient space computation.

To address this issue, we employ low rank moment to more accurately represent the overall gradient space. Specifically, using Adam as an example, for the parameter gradient $G_t \in \mathbb{R}^{m\times n}$, there exists a first-order moment $M_t \in \mathbb{R}^{m\times n}$. Since Adam incorporates historical gradient information at each iteration, its moment term can theoretically help the optimization algorithm better approximate the optimal gradient direction for the overall task, particularly when the task's loss function exhibits a flat or irregular landscape. Thus, after training, we can leverage first-order moment information to capture the gradient direction of the current task and calculate the gradient sharing space. Let L denote the number of parameter layers to be trained.

For the first task, we utilize first-order moments of each layer's parameters, denoted as $M_1 = \{M_1^1, M_1^2, \ldots, M_1^l, \ldots, M_1^L\}$. We then perform singular value decomposition (SVD) on each layer, yielding $M_1^l = U_1^l \sum_1^l V_1^{l\top}$. Finally we execute a k-rank approximation under the specified constraints:

$$\|(M_1^l)_k\|_F^2 > \epsilon_t^l \|M_1^l\|_F^2 \tag{1}$$

where ϵ_t^l is an approximation threshold. We select the first k vectors from U_1^l to form layer gradient space, denoted as $\mathcal{S}_1^l = [u_{1,1}^l, u_{1,2}^l, \ldots, u_{1,k}^l]$, and aggregate the layer-wise gradient spaces to obtain overall gradient space $\mathcal{S} = \{\{\mathcal{S}_1^l\}_{l=1}^L\}$ for the current task.

For task 2 to T, we use the second task as an example to illustrate our method. After completing training, we use the first-order moment $M_2 = \{M_2^1, M_2^2, \ldots, M_2^l, \ldots, M_2^L\}$ obtained from the second task to calculate the component that is orthogonal to the previously gradient space:

$$\hat{M_2^l} = M_2^l - \mathcal{S}^l(\mathcal{S}^l)^\top M_2^l = M_2^l - M_{2,Proj}^l \tag{2}$$

We perform SVD decomposition on the first-order moment of each layer, obtaining $\hat{M}_2^l = U_2^l \Sigma_2^l V_2^{l\top}$. Then we apply the updated constraints and the approximation threshold ϵ_t^l to perform a k-rank approximation:

$$\|(\hat{M}_2^l)_k\|_F^2 + \|\hat{M}_{2,Proj}^l\|_F^2 \geq \epsilon_t^l \|\hat{M}_2^l\|_F^2 \tag{3}$$

Finally, we update the gradient space as follows:

$$\mathcal{S} = [\mathcal{S}, u_{2,1}^l, u_{2,2}^l, \ldots, u_{2,k}^l] \tag{4}$$

As the number of tasks increases, the gradient space will gradually expand. Therefore, it is necessary to constrain the gradient space and control its size by filtering the singular values to maintain a fixed dimension.

3.2 Low Rank Projection Optimization

In this section, we leverage the gradient shared space to project the training parameters effectively. Our training parameters consist of both LoRA and the full-rank parameters. The core idea behind low-rank projection is to reduce redundant information by constraining updates within the low-rank gradient space, ensuring learning focuses on critical direction updates. This approach mitigates overfitting and improves the model's generalization ability in high-dimensional data, resulting in a more stable training process, while maintaining fine-tuning efficiency.

Specifically, for LoRA parameters, the projection is applied to parameter A, which is projected into the gradient shared space. Given the gradient $G_{A,l} \in \mathbb{R}^{m\times n}$ of parameter A and the gradient space $\mathcal{S}^{A,l}_{t-1}$:

$$G^{'}_{A,l} = G_{A,l} - \mathcal{S}^{A,l}_{t-1}(\mathcal{S}^{A,l}_{t-1})^{\top} G_{A,l} \tag{5}$$

For full-rank parameters, following Zhao et al. (2024), we apply low-rank updates during Adam optimization rather than full-rank updates. Since full-parameter training introduces additional memory overhead and given that parameter gradients tend to exhibit a low-rank structure over the course of training, it is essential to preserve their low-rank nature as much as possible throughout the optimization. Given a full-rank parameter gradient $G_{t,l} \in \mathbb{R}^{m\times n}$, we decompose it into a low-rank structure using $G_{t,l} = U_l \sum_l V_l^{\top}$, then we select first k vectors $U_{l,k}$ and $V_{l,k}$, and project them into $G_{t,l}$ as follows:

$$G^{'}_{t,l} = U^{\top}_{l,k} G_{t,l} V_{l,k} \tag{6}$$

The original gradient information is compressed by projecting $G_{t,l}$ into a low-rank representation $G^{'}_{t,l}$. This reduces the dimensionality of the data while preserving its most significant features. Then $G^{'}_{t,l}$ is projected into gradient space $\mathcal{S}^{l}_{t-1}$ as follows:

$$P_{t,l} = G^{'}_{t,l} - \mathcal{S}^{l}_{t-1}(\mathcal{S}^{l}_{t-1})^{\top} G^{'}_{t,l} \tag{7}$$

The projected gradient $G^{'}_{t,l}$ of LoRA and the low-rank projected gradient $P_{t,l}$ are then optimized by Adam:

$$M_{t,l} = \beta_1 M_{t-1,l} + (1-\beta_1) P_{t,l} \tag{8}$$
$$V_{t,l} = \beta_2 V_{t-1,l} + (1-\beta_2) P^{2}_{t,l} \tag{9}$$
$$P^{'}_{t,l} = M_{t,l} / \sqrt{V_{t,l} + \epsilon} \tag{10}$$

Finally, the low-rank projected gradient is scaled back to the original gradient dimension:

$$\hat{G_{t,l}} = \alpha U_{l,k} P^{'}_{t,l} V^{\top}_{l,k} \tag{11}$$
$$W_{t,l} \leftarrow W_{t-1,l} + \eta \hat{G_{t,l}} \tag{12}$$

where α is the scaling factor and η is the learning rate. LoRA gradients do not require dimensional expansion and directly update the weights with Equation 12. However, frequent low-rank operations can introduce additional computational overhead. Therefore, we minimize the low-rank operations for full-rank parameters by updating them at fixed intervals. Simultaneously, the projection process in Equation 6 is simplified by projecting the gradients into a subspace, denoted as $G^{'}_{t,l} = U^{\top}_{l,k} G_{t,l}$.

4 Experiments

In this section, we present the experimental setup and evaluate the performance of the proposed GORP method across multiple tasks. The focus is on assessing the advantages of GORP in terms of model performance and adaptability, while also comparing it with existing mainstream methods.

4.1 Experimental Setups

Models and Datasets. To evaluate the proposed method, we employ two widely adopted language models: the encoder-decoder T5-large model (Raffel et al., 2020) with 770M parameters and the decoder-only Llama2 model (Touvron et al., 2023) with 7B parameters. For datasets, we utilize the standard CL benchmarks (Zhang et al., 2015) and the large number of tasks (Razdaibiedina et al., 2023) as our experimental datasets. The standard CL benchmarks consist of classification datasets with 4 tasks and 5 categories, while the large number of tasks dataset includes a long-sequence CL dataset with 15 tasks, comprising the GLUE benchmark (Wang et al., 2018), SuperGLUE benchmark (Wang et al., 2019), and the IMDB movie reviews dataset (Maas et al., 2011). Following the experimental setup of Qin and Joty (2022) and Wang et al. (2023), we shuffle the tasks in the datasets and establish three different task orders. Detailed information is provided in Appendix B.

Evaluation Metrics. We evaluate the effectiveness of our GORP method from multiple perspectives using various evaluation metrics, including Average Accuracy, Backward Transfer (BWT), Parameter Orthogonality, and Gradient Orthogonality. The detailed calculation methods are provided in Appendix C.

Baselines. To demonstrate the effectiveness of our method, we compare it with various CL baseline approaches, including both non-continual learning methods and non-continual learning methods.

	Standard CL Benchmark				Large Number of Tasks			
	Order-1	**Order-2**	**Order-3**	**Avg**	**Order-4**	**Order-5**	**Order-6**	**Avg**
ProgPrompt	75.2	75.1	75.1	75.1	78.3	77.9	77.9	78.0
PerTaskFT	70.0	70.0	70.0	70.0	78.1	78.1	78.1	78.1
MTL	80.0	80.0	80.0	80.0	76.5	76.5	76.5	76.5
SeqFT	18.9	24.9	41.7	28.5	7.5	7.4	7.5	7.4
SeqLoRA	44.6	32.7	53.7	43.7	2.0	1.9	1.6	1.8
IncLoRA	66.0	64.9	68.3	66.4	54.7	53.2	62.2	56.7
Replay	55.2	56.9	61.3	57.8	44.5	46.5	45.1	45.4
EWC	48.7	47.7	54.5	50.3	46.9	45.6	45.6	46.0
LwF	50.2	52.0	64.3	55.5	49.9	50.5	49.5	49.9
L2P	60.3	61.7	61.1	61.0	56.9	56.9	56.1	56.6
LFPT5	65.3	68.0	71.5	68.3	70.0	73.0	73.8	72.3
O-LoRA	75.4	75.7	76.3	75.8	72.3	64.8	71.6	69.6
MIGU	77.1	77.0	75.6	76.6	67.3	68.5	74.2	70.0
N-LoRA	79.2	78.4	78.8	78.8	73.6	70.3	73.2	72.4
GORP	**79.7**	**79.9**	**79.7**	**79.8**	**76.1**	**76.2**	**75.6**	**76.0**

Table 2: Performance comparison across different methods on Standard CL Benchmark and Large Number of Tasks. The average accuracy after training on the final task is reported.

- ***Non-Continual Learning Methods***: **MTL** (Multi Task Learning), which involves jointly training on multiple task datasets, typically represents the upper bound of continual learning. **PerTaskFT** trains an independent model for each task, **SeqFT** (d'Autume et al., 2019) entails continual training of all parameters, **SeqLoRA** focuses on training only one LoRA, and **IncLoRA** involves training a new LoRA for each task.

- ***Continual Learning Methods***: **Replay** involves merging old task data to train new tasks, while **EWC** (Kirkpatrick et al., 2017) and **LwF** (Li and Hoiem, 2018) adjust model parameters using regularization losses. **L2P** (Wang et al., 2022) and **LFPT5** (Qin and Joty, 2022) dynamically design prompts to adapt to new tasks, and **O-LoRA** (Wang et al., 2023) constrains LoRA parameters to be orthogonal in a subspace to learn new tasks. **MIGU** (Du et al., 2024) considers output gradient normalization distributions to filter parameter updates, and **N-LoRA** (Yang et al., 2025) reduces collisions by sparsifying parameter updates.

4.2 Main Results

We compare the performance of GORP with baseline methods on two types of CL benchmarks. The experimental results across different task orders are summarized in Table 2.

	Order-1	**Order-2**	**Order-3**	**Avg**
O-LoRA	76.8	75.7	75.7	76.1
N-LoRA	77.2	77.3	**78.4**	77.6
GORP	**78.7**	**78.8**	78.2	**78.6**

Table 3: Performance comparison of various methods implemented on the Llama2-7B model, reporting average accuracy across all task orders and evaluated across multiple task orders within the Standard CL Benchmark.

Performance on standard CL benchmarks. On T5 model, GORP demonstrates consistent superiority over all prior methods across various task sequences, achieving significant improvements on standard continual learning benchmarks. Specifically, GORP improves performance by 4% over baseline methods while closely approaching MTL performance. As shown in Table 3, GORP also significantly outperforms baseline methods on LLaMA2-7B, achieving a 2.5% performance gain. These results highlight the effectiveness of our approach, even with larger model parameters.

Performance on a Large Number of Tasks. Continual learning tasks with long sequences are generally more challenging. As shown in Table 2, GORP consistently outperforms the baseline methods, achieving a 6.1% performance improvement. It also surpasses other state-of-the-art methods, with GORP's performance approaching that of MTL. Additionally, GORP performs more simi-

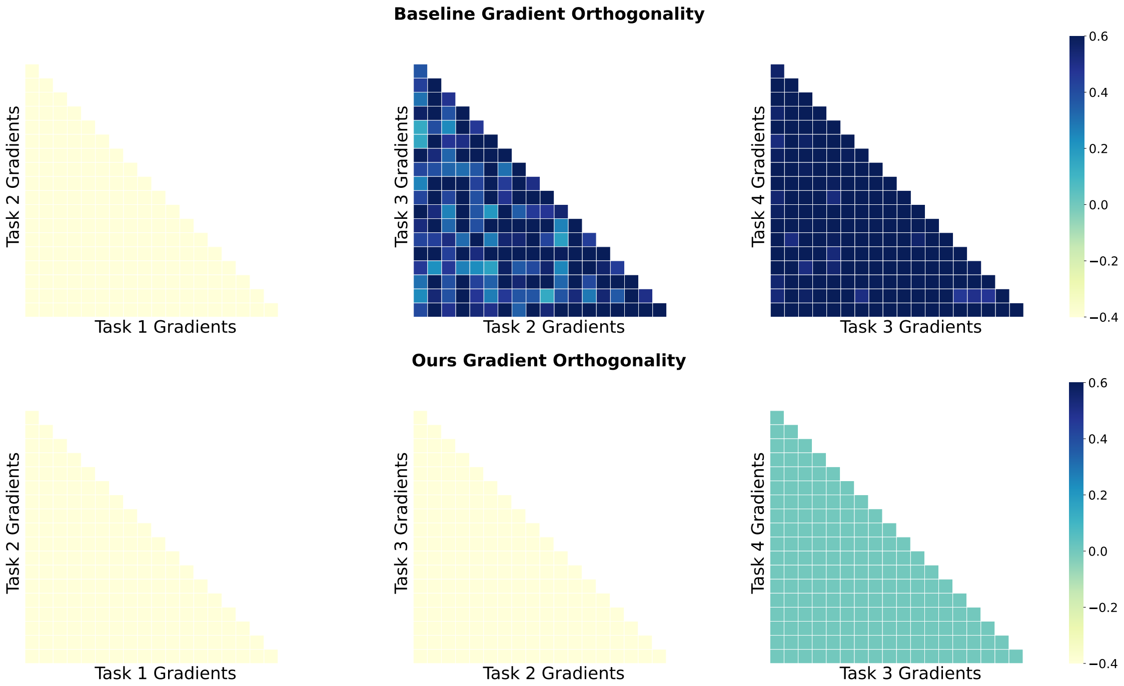

Figure 2: The visualization comparison of gradient orthogonality between Baseline and our method. Although the first two tasks maintain orthogonality, gradient interference between parameters gradually increases as more tasks are added, while our method consistently preserves orthogonality.

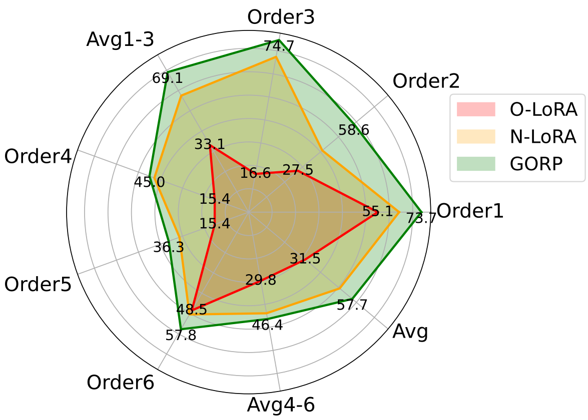

Figure 3: Performance comparison of generalization on unseen tasks. GORP consistently outperforms other methods across all task orders.

	BWT (%)	
	Avg Order 1-3	**Avg Order 4-6**
O-LoRA	-7.8	-16.4
GORP	**-0.8**	**-4.3**

Table 4: The forgetting rate comparison between the baseline and our proposed method, quantified using Backward Transfer (BWT) as the evaluation metric. As evidenced by the comparative results presented in the table, our method demonstrates a 7% and 12.1% reduction in forgetting rate compared to the baseline.

larly to PerTaskFT than other methods, suggesting that combining low-rank parameters with full parameters helps narrow the performance gap.

Generalization of LLMs. This part explores the generalization ability of our proposed GORP. We train on the first T-1 tasks, and test on the unseen t-th task, evaluating directly on the unseen task for comparison. As shown in Figure 3, the generative capability of pre-trained LLMs on unseen tasks is nearly zero. Although O-LoRA and its improved version, N-LoRA, outperform the pre-trained model on unseen tasks, the GORP method surpasses these comparative methods in generative ability. Across all task order configurations, GORP surpasses N-LoRA and O-LoRA, achieving average performance improvements of 7.0% and 26.2%, respectively. The results demonstrate the superior generative capability of GORP on unseen tasks.

4.3 Ablation Study

In this section, we conduct ablation experiments to assess the contribution of each component to GORP. As shown in Figure 4, adding low-rank projections to LoRA improves performance by an average of 0.7% compared to the baseline. Combining LoRA with full-rank parameters and low-rank projection results in an average improvement of 2.0%, while the overall improvement reaches 3.9%. The results

	Method		
	O-LoRA	**N-LoRA**	**GORP**
FLOPs	68.4	84.3	0.125
($\times 10^{12}$)	$1\times$	$1.23\times$	1.8e-3$\times$
Time/task	128.5	97.7	128.1
	$1\times$	$0.76\times$	$0.99\times$

Table 5: Time complexity comparison of different methods on Standard CL Benchmark.

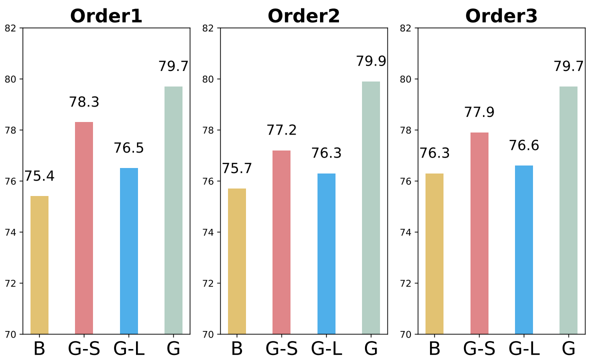

Figure 4: Ablation study of our method. B refers to the baseline method, L refers to low-rank projection for full-rank parameters, S refers to projection for LoRA, and G refers to our GORP method, which outperforms other components.

suggest that the incorporating both full-rank and low-rank parameters produces a complementary effect. The full-rank parameters enhance model flexibility and enable finer-grained adjustments, leading to improved performance. The ablation results confirm the effectiveness of each component.

4.4 Model Forgetting

Forgetting is a critical challenge in continual learning. To address this, we compare the forgetting rate of GORP with baseline methods. As shown in Table 4, GORP achieves a forgetting rate of just 0.8%, while baseline methods exhibit a rate of 7.8%, representing a 7.0% reduction. This result highlights the strong anti-forgetting capability of GORP.

Gradient space plays a crucial role in mitigating forgetting. While O-LoRA explicitly enforces orthogonality constraints on LoRA weights, GORP applies implicit constraints to regulate gradients. We compare the updates of parameter A in GORP and O-LoRA from both parameter and gradient perspectives, visualizing the weight distribution of A and the orthogonality of gradient distributions. As shown in Figure 5, the baseline method maintains parameter orthogonality throughout. Although GORP exhibits slightly weaker parameter

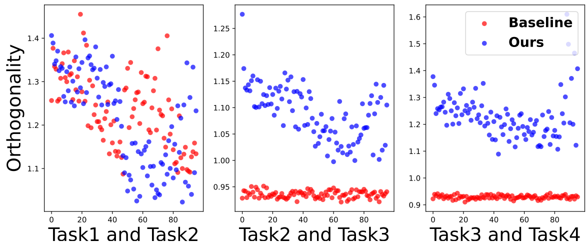

Figure 5: The visualization comparison of parameter orthogonality between baseline and our method. Although the parameter orthogonality of our method is higher compared to the baseline, the difference is not significant.

orthogonality, the difference is minimal. However, GORP demonstrates highly stable gradient orthogonality in Figure 2, enabling better gradient direction control while allowing parameters to update within a larger space, thereby increasing their degrees of freedom.

4.5 Time complexity Analysis

We present in Table 5 the floating point operations per second (FLOPs) and total running times (in seconds) of different methods on the standard CL benchmarks. Compared to O-LORA, our proposed GORP method requires nearly the same amount of time but significantly reduces computational cost. In contrast, N-LoRA reduces training time but increases computational demand. This indicates that our GORP method does not introduce significant computational delays and optimizes efficiency, making it a more resource-efficient alternative to O-LORA. While N-LoRA offers desirable speedup, it may result in higher computational burden. Therefore, GORP may be more suitable for scenarios where both time and computational resources are critical.

5 Conclusion

In this work, we propose GORP, a novel training strategy that overcomes these limitations by synergistically combining full and low-rank parameters and jointly updating within a unified low-rank gradient subspace. GORP is enable to expand the search space for optimal solutions while preserving the essential properties of continual fine-tuning. Through extensive empirical evaluations, we show that GORP effectively addresses the stability-plasticity dilemma in continual learning, all while maintaining computational efficiency during the fine-tuning.

Limitations

While GORP outperforms existing methods on continual learning benchmarks, several limitations should be considered. First, as task sequences expand, continuously updating task vectors within the gradient subspace becomes necessary. Therefore, effectively capturing increasing task diversity within constrained dimensional boundaries is a key challenge. Additionally, while GORP has shown strong performance in known continual data environments, its effectiveness in more complex real-world scenarios remains to be further validated.

References

Kartikeya Badola, Shachi Dave, and Partha Talukdar. 2023. Parameter-efficient finetuning for robust continual multilingual learning. In *Findings of the Association for Computational Linguistics: ACL 2023*, pages 9763–9780, Toronto, Canada. Association for Computational Linguistics.

Elad Ben Zaken, Yoav Goldberg, and Shauli Ravfogel. 2022. BitFit: Simple parameter-efficient fine-tuning for transformer-based masked language-models. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 1–9, Dublin, Ireland. Association for Computational Linguistics.

Dan Biderman, Jacob Portes, Jose Javier Gonzalez Ortiz, Mansheej Paul, Philip Greengard, Connor Jennings, Daniel King, Sam Havens, Vitaliy Chiley, Jonathan Frankle, Cody Blakeney, and John Patrick Cunningham. 2024. LoRA learns less and forgets less. *Transactions on Machine Learning Research*. Featured Certification.

Haolin Chen and Philip N. Garner. 2024. Bayesian parameter-efficient fine-tuning for overcoming catastrophic forgetting. *Preprint*, arXiv:2402.12220.

Rajas Chitale, Ankit Vaidya, Aditya Kane, and Archana Santosh Ghotkar. 2023. Task arithmetic with loRA for continual learning. In *Workshop on Advancing Neural Network Training: Computational Efficiency, Scalability, and Resource Optimization (WANT@NeurIPS 2023)*.

Cyprien de Masson d'Autume, Sebastian Ruder, Lingpeng Kong, and Dani Yogatama. 2019. *Episodic memory in lifelong language learning*. Curran Associates Inc., Red Hook, NY, USA.

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 *Findings of the Association for Computational Linguistics: EMNLP 2024*, pages 6503–6522, Miami, Florida, USA. Association for Computational Linguistics.

Mehrdad Farajtabar, Navid Azizan, Alex Mott, and Ang Li. 2020. Orthogonal gradient descent for continual learning. In *Proceedings of the Twenty Third International Conference on Artificial Intelligence and Statistics*, volume 108 of *Proceedings of Machine Learning Research*, pages 3762–3773. PMLR.

Yongchang Hao, Yanshuai Cao, and Lili Mou. 2024. Flora: Low-rank adapters are secretly gradient compressors. In *Forty-first International Conference on Machine Learning*.

Roee Hendel, Mor Geva, and Amir Globerson. 2023. In-context learning creates task vectors. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 9318–9333, Singapore. 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. In *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pages 2790–2799. PMLR.

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*.

Jianheng Huang, Leyang Cui, Ante Wang, Chengyi Yang, Xinting Liao, Linfeng Song, Junfeng Yao, and Jinsong Su. 2024. Mitigating catastrophic forgetting in large language models with self-synthesized rehearsal. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1416–1428, Bangkok, Thailand. Association for Computational Linguistics.

James Kirkpatrick, Razvan Pascanu, Neil Rabinowitz, Joel Veness, Guillaume Desjardins, Andrei A. Rusu, Kieran Milan, John Quan, Tiago Ramalho, Agnieszka Grabska-Barwinska, Demis Hassabis, Claudia Clopath, Dharshan Kumaran, and Raia Hadsell. 2017. Overcoming catastrophic forgetting in neural networks. *Proceedings of the National Academy of Sciences*, 114(13):3521–3526.

Yajing Kong, Liu Liu, Zhen Wang, and Dacheng Tao. 2022. Balancing stability and plasticity through advanced null space in continual learning. In *Computer Vision – ECCV 2022: 17th European Conference, Tel Aviv, Israel, October 23–27, 2022, Proceedings, Part XXVI*, page 219–236, Berlin, Heidelberg. Springer-Verlag.

Zhizhong Li and Derek Hoiem. 2018. Learning without forgetting. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 40(12):2935–2947.

Vladislav Lialin, Namrata Shivagunde, Sherin Muckatira, and Anna Rumshisky. 2023. Relora: High-rank training through low-rank updates. *Preprint*, arXiv:2307.05695.

Yan-Shuo Liang and Wu-Jun Li. 2023. Adaptive plasticity improvement for continual learning. In *2023 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 7816–7825.

Guoliang Lin, Hanlu Chu, and Hanjiang Lai. 2022a. Towards better plasticity-stability trade-off in incremental learning: A simple linear connector. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 89–98.

Sen Lin, Li Yang, Deliang Fan, and Junshan Zhang. 2022b. Beyond not-forgetting: Continual learning with backward knowledge transfer. In *Advances in Neural Information Processing Systems*, volume 35, pages 16165–16177. Curran Associates, Inc.

Sen Lin, Li Yang, Deliang Fan, and Junshan Zhang. 2022c. TRGP: trust region gradient projection for continual learning. In *The Tenth International Conference on Learning Representations, ICLR 2022, Virtual Event, April 25-29, 2022*. OpenReview.net.

Jialin Liu, Jianhua Wu, Jie Liu, and Yutai Duan. 2024a. Learning attentional mixture of loras for language model continual learning. *Preprint*, arXiv:2409.19611.

Sheng Liu, Haotian Ye, Lei Xing, and James Y. Zou. 2024b. In-context vectors: Making in context learning more effective and controllable through latent space steering. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pages 32287–32307. PMLR.

Yuheng Lu, Bingshuo Qian, Caixia Yuan, Huixing Jiang, and Xiaojie Wang. 2024. Controlled low-rank adaptation with subspace regularization for continued training on large language models. *Preprint*, arXiv:2410.16801.

Andrew L. Maas, Raymond E. Daly, Peter T. Pham, Dan Huang, Andrew Y. Ng, and Christopher Potts. 2011. Learning word vectors for sentiment analysis. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 142–150, Portland, Oregon, USA. Association for Computational Linguistics.

Navyansh Mahla, Kshitij Sharad Jadhav, and Ganesh Ramakrishnan. 2025. Exploring gradient subspaces: Addressing and overcoming lora's limitations in federated fine-tuning of large language models. *Preprint*, arXiv:2410.23111.

Jingyang Qiao, Zhizhong Zhang, Xin Tan, Yanyun Qu, Wensheng Zhang, Zhi Han, and Yuan Xie. 2024. Gradient projection for continual parameter-efficient tuning. *Preprint*, arXiv:2405.13383.

Chengwei Qin and Shafiq R. Joty. 2022. LFPT5: A unified framework for lifelong few-shot language learning based on prompt tuning of T5. In *The Tenth International Conference on Learning Representations, ICLR 2022, Virtual Event, April 25-29, 2022*. OpenReview.net.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *J. Mach. Learn. Res.*, 21(1).

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

Gobinda Saha, Isha Garg, and Kaushik Roy. 2021. Gradient projection memory for continual learning. In *International Conference on Learning Representations*.

Gobinda Saha and Kaushik Roy. 2023. Continual learning with scaled gradient projection. *Proceedings of the AAAI Conference on Artificial Intelligence*, 37(8):9677–9685.

Haizhou Shi, Zihao Xu, Hengyi Wang, Weiyi Qin, Wenyuan Wang, Yibin Wang, Zifeng Wang, Sayna Ebrahimi, and Hao Wang. 2024. Continual learning of large language models: A comprehensive survey. *Preprint*, arXiv:2404.16789.

James Seale Smith, Yen-Chang Hsu, Lingyu Zhang, Ting Hua, Zsolt Kira, Yilin Shen, and Hongxia Jin. 2024. Continual diffusion: Continual customization of text-to-image diffusion with c-lora. *Trans. Mach. Learn. Res.*, 2024.

Chenyang Song, Xu Han, Zheni Zeng, Kuai Li, Chen Chen, Zhiyuan Liu, Maosong Sun, and Tao Yang. 2023. Conpet: Continual parameter-efficient tuning for large language models. *Preprint*, arXiv:2309.14763.

Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, Aurelien Rodriguez, Armand Joulin, Edouard Grave, and Guillaume Lample. 2023. Llama: Open and efficient foundation language models. *Preprint*, arXiv:2302.13971.

Alex Wang, Yada Pruksachatkun, Nikita Nangia, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel Bowman. 2019. Superglue: A stickier benchmark for general-purpose language understanding systems. In *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc.

Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel Bowman. 2018. GLUE: A multi-task benchmark and analysis platform for natural language understanding. In *Proceedings of the 2018 EMNLP Workshop BlackboxNLP: Analyzing *and Interpreting Neural Networks for NLP*, pages 353–355, Brussels, Belgium. Association for Computational Linguistics.

Shipeng Wang, Xiaorong Li, Jian Sun, and Zongben Xu. 2021. Training networks in null space of feature covariance for continual learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 184–193.

Xiao Wang, Tianze Chen, Qiming Ge, Han Xia, Rong Bao, Rui Zheng, Qi Zhang, Tao Gui, and Xuanjing Huang. 2023. 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.

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

Jason Wei, Yi Tay, Rishi Bommasani, Colin Raffel, Barret Zoph, Sebastian Borgeaud, Dani Yogatama, Maarten Bosma, Denny Zhou, Donald Metzler, Ed H. Chi, Tatsunori Hashimoto, Oriol Vinyals, Percy Liang, Jeff Dean, and William Fedus. 2022a. Emergent abilities of large language models. *Transactions on Machine Learning Research*. Survey Certification.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Brian Ichter, Fei Xia, Ed H. Chi, Quoc V. Le, and Denny Zhou. 2022b. Chain-of-thought prompting elicits reasoning in large language models. In *Advances in Neural Information Processing Systems 35: Annual Conference on Neural Information Processing Systems 2022, NeurIPS 2022, New Orleans, LA, USA, November 28 - December 9, 2022*.

Martin Wistuba, Prabhu Teja S, Lukas Balles, and Giovanni Zappella. 2024. Continual learning with low rank adaptation. In *NeurIPS 2023 Workshop on Distribution Shifts: New Frontiers with Foundation Models*.

Wenhan Xia, Chengwei Qin, and Elad Hazan. 2024. Chain of lora: Efficient fine-tuning of language models via residual learning. *Preprint*, arXiv:2401.04151.

Shuo Yang, Kun-Peng Ning, Yu-Yang Liu, Jia-Yu Yao, Yong-Hong Tian, Yi-Bing Song, and Li Yuan. 2025. Is parameter collision hindering continual learning in LLMs? In *Proceedings of the 31st International Conference on Computational Linguistics*, pages 4243–4259, Abu Dhabi, UAE. Association for Computational Linguistics.

Xiang Zhang, Junbo Zhao, and Yann LeCun. 2015. Character-level convolutional networks for text classification. In *Advances in Neural Information Processing Systems*, volume 28. Curran Associates, Inc.

Jiawei Zhao, Zhenyu Zhang, Beidi Chen, Zhangyang Wang, Anima Anandkumar, and Yuandong Tian. 2024. Galore: Memory-efficient LLM training by gradient low-rank projection. In *Forty-first International Conference on Machine Learning, ICML 2024, Vienna, Austria, July 21-27, 2024*.

Yingxiu Zhao, Yinhe Zheng, Zhiliang Tian, Chang Gao, Jian Sun, and Nevin L. Zhang. 2022. Prompt conditioned VAE: Enhancing generative replay for lifelong learning in task-oriented dialogue. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 11153–11169, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

Junhao Zheng, Qianli Ma, Zhen Liu, Binquan Wu, and Huawen Feng. 2024a. Beyond anti-forgetting: Multimodal continual instruction tuning with positive forward transfer. *Preprint*, arXiv:2401.09181.

Junhao Zheng, Shengjie Qiu, Chengming Shi, and Qianli Ma. 2024b. Towards lifelong learning of large language models: A survey. *Preprint*, arXiv:2406.06391.

A Preliminary Knowledge

A.1 Continual learning setups

For consecutive tasks $\{T_1, T_2, \ldots, T_n\}$, each task T_t contains N_t samples $\{x_t, y_t\}_{t=1}^{N_t}$. In the t-th task, each step will sample n training samples $\mathcal{B}_n$ from the task for training, obtain parameter weights W_s^t, and then accumulate the weights to obtain the weight of the current task $W_t = \sum_s W_s^t$, and integrate with the previous task weight to get $W_t^{'} = W_{t-1}^{'} + W_t$. The model is able to retain its performance on previous tasks while progressively learning new ones, thereby minimizing the forgetting of earlier tasks.

A.2 Low Rank Adaptation

For a pre-trained weight $W_p \in \mathbb{R}^{m \times n}$, LoRA freezes the pretrained parameters and updates $W_{new} = W_p + \Delta W = W_p + AB$ by training low rank parameters, where $A \in \mathbb{R}^{m \times k}$ and $B \in \mathbb{R}^{k \times n}$, and rank $k \ll min(m, n)$. For a linear layer, the output can be written by Equation 13:

$$y = (W_p + \Delta W)x = W_p x + ABx \tag{13}$$

Through low rank updates, W_{new} retains the capabilities of pretrained models and also improves the generalization ability on downstream tasks.

B Datasets and Task Details

This part presents the datasets used in the experiments, along with the data categories and their corresponding tasks. The detailed information is provided in Table 6. CL benchmark includes Yelp, Amazon, Dbpedia, Yahoo and Agnews, GLUE dataset includes MNLI, QQP, RTE and SST-2, and SuperGLUE includes WiC, CB, COPA, BoolQA, MultiRC and IMDB. For the large number of tasks, we select 1000 random samples for training each task and 500 samples per class for validation and testing.

We report the task sequences used for CL experiments on the T5 and LLAMA2 models in Table 7. These datasets span diverse categories, including natural language inference (NLI), sentiment classification (SC), and topic classification (TC), ensuring diverse abilities of the model's generalization across multiple tasks. And the task instructions for different categories are shown in Table 8.

C Evaluation Metrics

Let $a_{i,j}$ be the test accuracy of the i-th task after training on the j-th task. A_i denotes the A matrix of LoRA, and $G_{A,i}$ denotes the gradient of A matrix on the i-th task. We evaluate the model using the following metrics:

- **Average Accuracy (ACC)**: The average accuracy of all tasks after training on the last task:

$$ACC = \frac{1}{T}\sum_{i=1}^{T} a_i, T \tag{14}$$

- **Backward Transfer (BWT)**: The average forgetting of all tasks after training on the last tasks:

$$BWT = \frac{1}{T-1}\sum_{i=1}^{T-1} a_{i,T} - a_{i,i} \tag{15}$$

- **Parameter Orthogonality (PO)**: We use this metric to quantify the orthogonal overlap between A_i and A_j, for the reason that O-LoRA use A to capture gradient subspaces of previous tasks. The metric is calculated as:

$$PO_{i,j} = \|A_i^\top A_j\|^2 \tag{16}$$

- **Gradient Orthogonality (GO)**: We use this metric to quantify the orthogonal overlap between $G_{A,i}$ and $G_{A,j}$, showing the difference between the gradient space and the parameter space, calculated as:

$$GO_{i,j} = \|G_{A,i}^\top G_{A,j}\|^2 \tag{17}$$

D Implementation Details

We adapted the code-base from O-LoRA (Wang et al., 2023). And our improved version of the code is available in the supplementary meterial and will be released upon acceptance. All experiments were conducted on the machine with 8 NVIDIA L20 and were implemented with Deepspeed.

For T5 model, we employed LoRA to replace the SelfAttention layers and full-rank parameter trainings for the EncDecAttention layers. For all orders, we trained the models with one epoch, a constant learning rate 1e-03 for LoRA and 1e-05 for full-rank parameters, a training batch size of 8 per device, a evaluation batch size of 64 per device, and a weight decay rate of 0, a value 0.05 of λ. We set different scale factors for order 1 to 6. For order 1 to 3, we set scale factor 1 and 0.25 for order 4 to 6. In our method, the low-rank updates are interval, and we set the update gap 10.

For LLaMA2 model, we employed LoRA to replace the Self-atten layers and full-rank parameter trainings for the MLP Gate layers. For order 1 to 3, we trained the models with one epoch, a constant learning rate 2e-04 for LoRA and 1e-06 for full-rank parameters, a training batch size of 1 per device, a evaluation batch size of 4 per device, and a weight decay rate of 0, a value 0 of λ. We set scale factor 0.25 for order 1 to 3 and the value 20 of the interval gap for low-rank updates.

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	NLI	Various	Accuracy
QQP	GLUE	Paragraph detection	Quora	Accuracy
RTE	GLUE	NLI	News, Wikipedia	Accuracy
SST-2	GLUE	Sentiment analysis	Movie reviews	Accuracy
WiC	SuperGLUE	Word sense disambiguation	Lexical databases	Accuracy
CB	SuperGLUE	NLI	Various	Accuracy
COPA	SuperGLUE	QA	Blogs, encyclopedia	Accuracy
BoolQA	SuperGLUE	Boolean QA	Wikipedia	Accuracy
MultiRC	SuperGLUE	QA	Various	Accuracy
IMDB	SuperGLUE	Sentiment analysis	Movie reviews	Accuracy

Table 6: Datasets, Categories, Domians and evaluation Metrics.

Model	Order	Task Sequence
T5-large, Llama2	**1**	dbpedia → amazon → yahoo → ag
T5-large, Llama2	**2**	dbpedia → amazon → ag → yahoo
T5-large, Llama2	**3**	yahoo → amazon → ag → dbpedia
T5-large	**4**	mnli → cb → wic → copa → qqp → boolqa → rte → imdb → yelp → amazon → sst-2 → dbpedia → ag → multirc → yahoo
T5-large	**5**	multirc → boolqa → wic → mnli → cb → copa → qqp → rte → imdb → sst-2 → dbpedia → ag → yelp → amazon → yahoo
T5-large	**6**	yelp → amazon → mnli → cb → copa → qqp → rte → imdb → sst-2 → dbpedia → ag → yahoo → multirc → boolqa → wic

Table 7: Task sequences used for CL experiments on the T5 and LLAMA2 models.

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

Table 8: Instructions for different tasks.