Making LoRA Identifiable: Orthogonal Alignment for Continual Learning

Pratik Rakesh Singh* **Mohammadi Zaki*** **Akash Saha**
Aneesh Mukkamala **Pankaj Wasnik**
Sony Research India
{pratik.singh,pankaj.wasnik}@sony.com

Abstract

There is growing interest in applying Parameter-Efficient Fine-Tuning (PEFT) to Continual Learning, as it enables faster training, mitigates catastrophic forgetting, and facilitates better adaptation to new tasks. Prior approaches have explored a range of strategies, including extending new LoRA branches for each incoming task, training under orthogonal loss objectives across all task-specific LoRA modules, and employing gating mechanisms to integrate new and existing LoRA modules. However, these methods incur memory costs that grow linearly with the number of tasks, limiting their scalability. More recent approaches investigate training a single LoRA by leveraging asymmetries in its parameterization; however, they often suffer from insufficient representational alignment across the LoRA parameter space. To address these limitations, we propose Stiefel Optimization and Aligned Rotation (SOAR), a two-stage framework for continual LoRA learning. First, we constrain the $\mathbf{A}$ matrix to lie on the Stiefel manifold, thereby reducing coordinate misalignment to the space of orthogonal rotation matrices. Second, we introduce an Orthogonal Procrustes alignment procedure for the $\mathbf{B}$ matrix to estimate the optimal rotation, coupled with a per-rank retention merging strategy that quantifies directional agreement between representations of new and previously learned tasks. We conduct extensive experiments across multiple models and benchmarks, demonstrating that SOAR achieves strong continual learning performance without incurring additional memory overhead.

1 Introduction

Large language models are increasingly deployed in settings where they must continuously acquire new capabilities without degrading previously learned knowledge [18, 24]. This has driven growing interest in continual learning (CL) under parameter-efficient fine-tuning frameworks such as LoRA, where task-specific updates are represented as low-rank modifications to a frozen backbone [25, 10, 14, 11]. While maintaining a separate adapter per task is effective, it results in linear growth in memory and inference costs. This motivates the more challenging setting of *single-adapter continual learning*, where all task updates must be consolidated into a single shared LoRA module [14, 1]. A common strategy in this setting is to iteratively merge task-specific LoRA updates. Concretely, for a pretrained weight matrix $\mathbf{W}$, LoRA models task adaptation as $\mathbf{W} \leftarrow \mathbf{W} + \mathbf{BA}$. Prior work has observed that the two LoRA factors play different roles, with the input factor $\mathbf{A} \in \mathbb{R}^{r\times n}$ defining a low-dimensional subspace and the output factor $\mathbf{B} \in \mathbb{R}^{m\times r}$ capturing task-specific adaptations [14, 4, 33]. However, effectively leveraging this structure in continual learning remains challenging. In particular, existing methods operate on LoRA factors that are not expressed in a consistent coordinate system, making merging fundamentally unstable.

*These authors contributed equally to this work.

40th Conference on Neural Information Processing Systems (NeurIPS 2026).

This instability arises from a geometric property of the LoRA parameterization. The factorization $\Delta\mathbf{W} = \mathbf{BA}$ is non-identifiable: equivalent updates can differ by any *arbitrary invertible transformation* in the rank space. In particular, for any $\mathbf{R} \in GL(r)$, $\mathbf{BA} = (\mathbf{BR})(\mathbf{R}^{-1}\mathbf{A})$, leading to infinitely many equivalent factorizations.

As a result, LoRA adapters learned across tasks may reside in *misaligned* latent coordinate systems. Even if $\mathbf{A}$ and $\mathbf{B}$ are treated asymmetrically, merging them without resolving this ambiguity can introduce interference due to representational misalignment rather than true task conflict [3, 7].

In this work, we propose Stiefel Optimization and Aligned Rotation (**SOAR**), a principled framework for single-adapter continual LoRA learning that resolves these issues through a geometric treatment of the factorization. Our approach builds on the asymmetric roles of $\mathbf{A}$ and $\mathbf{B}$, but makes this structure well-posed for continual merging. First, we constrain the input factor $\mathbf{A}$ to lie on the Stiefel manifold, ensuring $\mathbf{AA}^\top = \mathbf{I}$. This removes the general linear invariance of the LoRA factorization, *reducing* the ambiguity from arbitrary invertible transformations $\mathbf{R} \in GL(r)$ to orthogonal transformations $\mathbf{R} \in \mathcal{O}(r)$, thereby inducing a consistent and non-redundant coordinate system across tasks (see Lemma 3.1).

This combination leads to a coherent consolidation strategy: $\mathbf{A}$ restricts the latent coordinate system to an orthonormal basis, reducing ambiguity to rotations, while $\mathbf{B}$ captures task-specific information that can then be explicitly aligned and merged via orthogonal transformations. Empirically, SOAR consistently outperforms existing single-adapter and merge-based baselines across multiple continual learning benchmarks, demonstrating that resolving the geometric inconsistency in LoRA parameterization is critical for mitigating forgetting.

While Orthogonal Procrustes alignment resolves the global rotational ambiguity between tasks, it operates at the full-matrix level and does not enforce consistent alignment across individual rank components. In practice, different directions of the low-rank decomposition can undergo varying degrees of change during task adaptation: some remain consistent and reusable, while others are substantially modified. As a result, a single global interpolation coefficient is insufficient to capture this heterogeneity. To address this, we introduce a rank-wise, forgetting-aware merging strategy that adaptively interpolates between aligned factors based on directional compatibility and relative change, enabling more fine-grained consolidation of shared and task-specific information.

With both corrections in place, rank-wise interpolation of $\mathbf{B}$ becomes well-defined, enabling directional agreement and magnitude drift to be measured in a consistent coordinate system. This, in turn, allows prior information to be selectively retained in directions that exhibit strong geometric compatibility and limited drift. The following section formalizes each of these components and the design principles that connect them. We summarize our contributions below:

- We propose **SOAR**, a framework for single-adapter continual LoRA learning that enables knowledge consolidation at constant memory cost by explicitly addressing *representational misalignment* across tasks.
- We show that LoRA adapters trained on different tasks can be expressed in inconsistent latent coordinate systems, making direct merging unstable. To address this, we constrain the input factor $\mathbf{A}$ to the Stiefel manifold, which enforces orthonormal rank directions and yields a stable, shared coordinate system across tasks.
- We introduce an *alignment-aware merging strategy* for the output factor $\mathbf{B}$ that first resolves the remaining rotational ambiguity via Orthogonal Procrustes alignment, and then performs *rank-wise, forgetting-aware interpolation* based on directional agreement and magnitude drift.
- We empirically validate SOAR across multiple continual learning benchmarks and model families, demonstrating consistent improvements over sequential and merge-based baselines while maintaining the efficiency of a single adapter.

2 Related Work

Continual learning has been studied extensively in the deep learning literature (see [24]). In this work, we focus specifically on continual learning under parameter-efficient fine-tuning (PEFT), where task adaptation is performed through lightweight low-rank updates rather than full-model optimization.

Parameter-efficient continual learning. A growing body of work has explored continual learning with LoRA and related PEFT methods. O-LoRA [25] constrains new LoRA updates to remain orthogonal to previous ones in order to reduce interference. GainLoRA [32] learns task-specific scaling factors over shared LoRA components for efficient adaptation across tasks. InfLoRA [11] mitigates task interference through similarity-aware routing and selective adaptation of LoRA modules. SAPT-LoRA [31] introduces structured sparsity and parameter selection mechanisms to preserve important task-specific information during adaptation. Most closely related to our setting, SLAO [14] studies single-adapter continual learning by maintaining a shared low-rank subspace across tasks while enforcing orthogonality constraints on task-specific components.

Unlike prior methods, SOAR explicitly addresses the geometric non-identifiability of the LoRA factorization. Our approach stabilizes the latent coordinate system through Stiefel-constrained optimization and resolves residual rotational ambiguity via Procrustes-aligned consolidation.

Model merging and weight-space composition. Related ideas have also been explored in model merging, where independently trained models or LoRA adapters are combined directly in weight space. Unlike continual learning, these methods typically assume simultaneous access to all task-specific adapters and therefore avoid sequential forgetting by design.

LoRA-LEGO [32] treats LoRA adapters as composable modules for multi-task composition without joint training. KnOTS [19] aligns task-specific LoRA subspaces prior to merging in order to reduce interference. Task Arithmetic [5] represents tasks as parameter-space vectors that can be composed through addition and subtraction operations. TIES-Merging [27] resolves conflicting parameter updates through trimming and interference-aware merging. OPCM [21] proposes a sequential, training-free framework for incrementally composing task-specific models without retraining.

In contrast to these approaches, SOAR operates in the continual single-adapter setting, where past task-specific adapters are not retained. Our method therefore focuses on stable sequential consolidation under memory constraints, rather than offline merging of independently available task updates.

3 Motivation

3.1 Continual learning with single LoRA

In LoRA, a frozen weight matrix $\mathbf{W}_0 \in \mathbb{R}^{m\times n}$ is augmented with a low-rank residual:

$$\mathbf{W} = \mathbf{W}_0 + \Delta\mathbf{W}, \qquad \Delta\mathbf{W} = \mathbf{B}\mathbf{A}, \tag{1}$$

where $\mathbf{B} \in \mathbb{R}^{m\times r}$, $\mathbf{A} \in \mathbb{R}^{r\times n}$, and $r \ll \min(m, n)$. We consider a sequence of tasks $\{\mathcal{D}_t\}_{t=1}^{T}$, where fine-tuning on task t yields task-specific parameters $(\mathbf{B}_{\text{ft}}^{(t)}, \mathbf{A}_{\text{ft}}^{(t)})$. Our goal is to maintain a single consolidated adapter $(\mathbf{B}_{\text{merge}}^{(t)}, \mathbf{A}_{\text{merge}}^{(t)})$ after each task, so that inference uses:

$$\mathbf{W}^{(t)} = \mathbf{W}_0 + \mathbf{B}_{\text{merge}}^{(t)}\,\mathbf{A}_{\text{merge}}^{(t)}, \tag{2}$$

Thus, the model stores only one consolidated LoRA update, keeping adapter memory fixed at $\mathcal{O}((m + n)r)$ rather than growing as $\mathcal{O}(T(m + n)r)$. The main challenge, therefore, shifts from storing past adapters to consolidating their information: the merged adapter must preserve useful knowledge from earlier tasks while adapting to the current one.

This consolidation should respect the asymmetric roles of the LoRA factors [14, 33]. Directly merging the full product $\mathbf{BA}$ discards the two-factor structure that makes LoRA parameter-efficient and interpretable. Likewise, applying the same merge rule to $\mathbf{A}$ and $\mathbf{B}$ ignores the distinction between input-side subspace selection and output-side task adaptation.

A natural approach to consolidation is to directly merge the fine-tuned factors $(\mathbf{B}_{\text{ft}}^{(t)}, \mathbf{A}_{\text{ft}}^{(t)})$ and $(\mathbf{B}_{\text{ft}}^{(t-1)}, \mathbf{A}_{\text{ft}}^{(t-1)})$ obtained from sequential fine-tuning. However, this is fundamentally ill-posed. Although both adapters are optimized using the same procedure, the LoRA factorization $\Delta\mathbf{W} = \mathbf{BA}$ is non-identifiable: for any invertible $\mathbf{R} \in GL(r)$, the reparameterization $(\mathbf{B}, \mathbf{A}) \mapsto (\mathbf{BR}, \mathbf{R}^{-1}\mathbf{A})$ yields an identical weight update. Consequently, the optimizer has no mechanism to enforce a canonical factorization, and two adapters minimizing their respective task losses may arrive at solutions expressed in entirely different latent coordinate systems. Under such representational misalignment,

merging $\mathbf{A}^{(t)}$ with $\mathbf{A}^{(t-1)}$, or $\mathbf{B}^{(t)}$ with $\mathbf{B}^{(t-1)}$, conflates directions that are geometrically incomparable, introducing interference that stems not from task conflict, but from coordinate mismatch. This motivates the need for a principled approach that resolves the latent ambiguity before any consolidation is attempted. In subsequent sections, we lay the foundations for a principled approach to merge LoRA factors that respects the underlying coordinate systems.

3.2 Stiefel Constraint as a Precondition for Alignment

We begin by addressing the ambiguity in the input factor $\mathbf{A}$, since it defines the latent coordinate system in which task-specific adaptations are represented. In unconstrained settings, this coordinate system is not identifiable: for any invertible matrix $\mathbf{R} \in GL(r)$, the factorization satisfies

$$\mathbf{BA} = (\mathbf{BR})(\mathbf{R}^{-1}\mathbf{A}),$$

implying that the same update can be represented in arbitrarily different coordinate systems. As a result, adapters trained across tasks may adopt incompatible latent bases, making subsequent comparison and merging ill-posed.

To address this, we constrain $\mathbf{A}$ to lie on the row-Stiefel manifold,

$$\mathbf{A}\mathbf{A}^\top = \mathbf{I}_r. \tag{3}$$

This constraint reduces the symmetry of the factorization, as formalized below.

Lemma 3.1. *[Key Lemma: Reduction of invariance] Let $\mathbf{A}_t, \mathbf{A}_{t+1} \in \mathbb{R}^{r\times n}$ be input factors learned for two consecutive tasks where $t \geq 1$, both satisfying the Stiefel constraint $\mathbf{A}_t\mathbf{A}_t^\top = \mathbf{A}_{t+1}\mathbf{A}_{t+1}^\top = \mathbf{I}_r$. Suppose there exists an invertible matrix $\mathbf{R} \in GL(r)$ such that*

$$\mathbf{A}_{t+1} = \mathbf{R}^{-1}\mathbf{A}_t.$$

Then $\mathbf{R}$ must satisfy $\mathbf{R}\mathbf{R}^\top = \mathbf{I}$, i.e., $\mathbf{R} \in \mathcal{O}(r)$.

Proof. Using $\mathbf{A}_{t+1}\mathbf{A}_{t+1}^\top = \mathbf{I}$, we obtain $\mathbf{R}^{-1}\mathbf{A}_t\mathbf{A}_t^\top(\mathbf{R}^{-1})^\top = \mathbf{I}$. Since $\mathbf{A}_t\mathbf{A}_t^\top = \mathbf{I}$, this reduces to $\mathbf{R}^{-1}(\mathbf{R}^{-1})^\top = \mathbf{I}$, which implies $\mathbf{R}\mathbf{R}^\top = \mathbf{I}$. □

Lemma 3.1 shows that the Stiefel constraint reduces the ambiguity of the LoRA factorization from arbitrary invertible transformations to orthogonal rotations. This is critical for continual merging: without it, alignment would require solving for a general linear transformation that may introduce scaling and shearing distortions. Restricting the ambiguity to $\mathcal{O}(r)$ instead yields a well-posed alignment problem that can be solved efficiently via Orthogonal Procrustes analysis (Section 3.3).

We optimize $\mathbf{A}$ on the Stiefel manifold using standard Riemannian optimization methods (Appendix B). Importantly, the role of the constraint is not to align adapters directly, but to impose a consistent geometric structure that makes subsequent alignment meaningful.

3.3 Rotational Ambiguity and Alignment of B

Constraining $\mathbf{A}$ to the Stiefel manifold removes general linear invariance, but the LoRA factorization remains ambiguous up to orthogonal transformations. For any $\mathbf{R} \in \mathcal{O}(r)$, $\mathbf{BA} = (\mathbf{BR})(\mathbf{R}^\top\mathbf{A})$, so adapters may still differ by a rotation of their latent rank coordinates. While this ambiguity is irrelevant in single-task fine-tuning, it becomes problematic in continual learning, where factor matrices are merged across tasks. In particular, interpolation of output factors, $\mathbf{B}_{\text{merge}}^{(t)} = (1-\alpha)\mathbf{B}_{\text{ft}}^{(t)} + \alpha\mathbf{B}_{\text{merge}}^{(t-1)}$, is only meaningful when both factors are expressed in a common coordinate frame.

To resolve this mismatch, we align the previously merged factor $\mathbf{B}_{\text{old}} = \mathbf{B}_{\text{merge}}^{(t-1)}$ to the coordinate system of the current task factor $\mathbf{B}_{\text{new}} = \mathbf{B}_{\text{ft}}^{(t)}$ using Orthogonal Procrustes alignment [16]:

$$\mathbf{R}^* = \arg\min_{\mathbf{R}\in\mathcal{O}(r)} \left\|\mathbf{B}_{\text{new}} - \mathbf{B}_{\text{old}}\mathbf{R}^\top\right\|_F^2. \tag{4}$$

Let $\mathbf{B}_{\text{new}}^\top\mathbf{B}_{\text{old}} = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^\top$ be the singular value decomposition. The optimal rotation is given by [16]: $\mathbf{R}^* = \mathbf{U}\mathbf{V}^\top$, and the aligned factor is $\tilde{\mathbf{B}}_{\text{old}} = \mathbf{B}_{\text{old}}(\mathbf{R}^*)^\top$.

This alignment step removes residual rotational mismatch while preserving the geometry of the learned subspace, enabling meaningful comparison and consolidation of rank directions across tasks. We adopt Orthogonal Procrustes because it provides a minimal, closed-form, and hyperparameter-free correction suitable for sequential continual learning.

Remark. Several prior works [14, 33, 20] have observed an asymmetry between the LoRA factors, where the input factor $\mathbf{A}$ tends to remain relatively stable across tasks while the output factor $\mathbf{B}$ captures a larger portion of task-specific adaptation. Our framework builds on this observation. Rather than explicitly merging $\mathbf{A}$ across tasks, we use the Stiefel constraint to stabilize its geometric structure and reduce factorization ambiguity. The primary consolidation mechanism is therefore applied to $\mathbf{B}$, whose representations are first aligned and then selectively merged across tasks. To further verify this, we conduct an additional ablation detailed in Appendix D.5

3.4 From Global Averaging to per-rank retention gating

While Procrustes alignment resolves the global coordinate mismatch between adapters, it does not determine how information should be consolidated across tasks. A natural approach is to interpolate between the aligned factors using a single global coefficient. However, such a strategy is inherently limited: different rank directions often evolve differently during fine-tuning. Some directions remain stable and encode information that is reusable across tasks, while others are substantially modified or repurposed for the current task.

This heterogeneity cannot be captured by a single interpolation weight. Instead, consolidation requires a mechanism to distinguish between directions that should retain prior knowledge and those that should follow the current update. Importantly, this decision depends on two complementary factors: whether the old and new directions are geometrically aligned, and how much the current task has altered the magnitude of the corresponding component.

To capture these effects, we introduce a rank-wise, forgetting-aware merging strategy that evaluates each direction independently. By combining directional agreement with a measure of magnitude drift, the method selectively retains prior information only where it remains compatible with the current task, while avoiding interference in directions that have been significantly rewritten.

4 Methodology

We now present SOAR, a continual LoRA learning framework consisting of two components: (i) Stiefel-constrained optimization of the input factor $\mathbf{A}$ to stabilize the latent coordinate system, and (ii) Procrustes-aligned, rank-wise consolidation of the output factor $\mathbf{B}$. Algorithm 1 summarizes the overall procedure.

STEP 1: Stiefel optimization. We constrain $\mathbf{A} \in \mathbb{R}^{r\times n}$ to lie on the Stiefel manifold $\mathrm{St}(r, n) = \{\mathbf{A} \in \mathbb{R}^{r\times n} \mid \mathbf{A}\mathbf{A}^\top = \mathbf{I}_r\}$. Rather than enforcing orthogonality through penalty terms, we optimize directly on the manifold using Riemannian Adam (Appendix B).

Although one could project $\mathbf{A}$ onto the manifold after fine-tuning, post-hoc projection breaks the coupling between $\mathbf{A}_{\mathrm{ft}}^{(t)}$ and $\mathbf{B}_{\mathrm{ft}}^{(t)}$, which are jointly optimized during LoRA adaptation (Appendix C).

STEP 2: Geometrically aligned merging. Following the asymmetric roles of the LoRA factors, we directly update $\mathbf{A}_{\mathrm{merge}}^{(t)} \leftarrow \mathbf{A}_{\mathrm{ft}}^{(t)}$ and perform continual consolidation only on $\mathbf{B}$. Given the newly fine-tuned factor $\mathbf{B}_{\mathrm{ft}}^{(t)}$ and the previously merged factor $\mathbf{B}_{\mathrm{merge}}^{(t-1)}$, our goal is to construct a consolidated output factor that preserves prior knowledge while adapting to the current task. To this end, we first align the factors via Orthogonal Procrustes analysis and then apply rank-wise retention gating.

Procrustes alignment and per-rank retention gating Let $\mathbf{B}_{\mathrm{new}} = \mathbf{B}_{\mathrm{ft}}^{(t)}$ and $\mathbf{B}_{\mathrm{old}} = \mathbf{B}_{\mathrm{merge}}^{(t-1)}$. As discussed in Sec. 3.3, the Stiefel constraint reduces the LoRA ambiguity to rotations within the rank space. Before comparing or merging columns, we therefore align $\mathbf{B}_{\mathrm{old}}$ with the coordinate frame of $\mathbf{B}_{\mathrm{new}}$ using an orthogonal Procrustes transformation.

Procrustes alignment. As mentioned in Sec. 3.3, we solve the orthogonal Procrustes problem in Eq. (4), which is efficiently done by computing the thin SVD $\mathbf{B}_{\text{new}}^{\top}\mathbf{B}_{\text{old}} = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^{\top}$. This gives the closed-form solution $\mathbf{R}^{*} = \mathbf{U}\mathbf{V}^{\top}$. The aligned historical factor is then $\widetilde{\mathbf{B}}_{\text{old}} = \mathbf{B}_{\text{old}}(\mathbf{R}^{*})^{\top}$, expressed in the coordinate frame of $\mathbf{B}_{\text{new}}$.

Per-rank retention gating. After aligning the old output factor with the new one, the remaining challenge is how to consolidate them. A single global interpolation weight is too coarse, since different rank directions evolve differently across tasks. Some directions remain stable and transferable, while others are substantially rewritten during adaptation. To capture this heterogeneity, we assign a separate retention score to each rank direction.

For each column $i \in \{1, \ldots, r\}$, let $\mathbf{b}_{\text{new},i}$ denote the i-th column of $\mathbf{B}_{\text{new}}$ and $\widetilde{\mathbf{b}}_{\text{old},i}$ denote the i-th column of $\widetilde{\mathbf{B}}_{\text{old}}$. We first measure their directional agreement:

$$c_i = \frac{\mathbf{b}_{\text{new},i}^{\top}\widetilde{\mathbf{b}}_{\text{old},i}}{\|\mathbf{b}_{\text{new},i}\|_2\|\widetilde{\mathbf{b}}_{\text{old},i}\|_2 + \epsilon} \in (-1, 1), \tag{5}$$

which captures whether the two adapters utilize a compatible output direction for rank component i.

Directional agreement alone is insufficient, since two columns may point in similar directions while differing substantially in magnitude. We therefore also compute the normalized residual

$$\delta_i = \frac{\|\mathbf{b}_{\text{new},i} - \widetilde{\mathbf{b}}_{\text{old},i}\|_2^2}{\|\widetilde{\mathbf{b}}_{\text{old},i}\|_2^2 + \epsilon}, \tag{6}$$

which quantifies how far the current task has moved the aligned historical direction relative to its original scale. These two signals are combined into a rank-wise retention gate:

$$\gamma_i = \max(c_i, 0)\exp\left(-(\delta_i/\tau)\right), \qquad \gamma_i \in [0, 1], \tag{7}$$

where $\tau > 0$ controls sensitivity to magnitude drift. The gate assigns high retention only to directions that remain both positively aligned and minimally altered.

Letting $\mathbf{\Gamma} = \text{diag}(\gamma_1, \ldots, \gamma_r)$, we update the output factor as

$$\mathbf{B}_{\text{merge}}^{(t)} = \mathbf{B}_{\text{new}} + \left(\widetilde{\mathbf{B}}_{\text{old}} - \mathbf{B}_{\text{new}}\right)\mathbf{\Gamma}, \tag{8}$$

or equivalently, for each rank direction, $\mathbf{b}_{\text{merge},i}^{(t)} = (1 - \gamma_i)\mathbf{b}_{\text{new},i} + \gamma_i\widetilde{\mathbf{b}}_{\text{old},i}$.

This anchors consolidation at the current task update: when evidence for recovery is weak ($\gamma_i \approx 0$), the new direction is preserved unchanged; when compatibility is strong ($\gamma_i \approx 1$), prior information is selectively restored.

Catastrophic forgetting mitigation. The retention gate provides SOAR with an explicit mechanism for catastrophic forgetting control within the low-rank subspace. The cosine term c_i measures directional compatibility between past and current adaptations, encouraging retention only when the corresponding rank directions remain aligned across tasks. The residual term δ_i measures the magnitude of task-induced change, suppressing retention for directions that have been substantially rewritten during adaptation. Together, these signals allow SOAR to selectively preserve reusable knowledge while permitting controlled forgetting in directions that have become task-specific, yielding a balance between stability and plasticity.

5 Experiments

Models and Datasets. We evaluate our approach across two model families and 3 models in total: Qwen3-8B, Llama-3.1-8B-Instruct, Llama-3.2-3B-Instruct. All experiments are conducted on one H100 GPU of 80GB. For evaluation similar to [14], we consider 3 continual learning benchmarks: (1) **Standard CL benchmark:** DBpedia, AG News, Amazon Polarity, Yahoo Answers. [30](2) **Large number of tasks:** It contains 4 standard CL tasks, along with GLUE tasks [23] (MNLI, QQP, SST-2, RTE) and 4 SuperGLUE tasks [22] (WiC, ReCoRD, MultiRC, BoolQ) and IMDB movie reviews. Each of these tasks uses 2500 randomly sampled training samples and 500 test samples per class

Table 1: (%) Performance comparison on three CL benchmark datasets using Qwen3-8B, across different task order, where O_i represents i^{th} task order.

	Standard CL Benchmark				Large Number of Tasks				SuperNI Benchmark		
Method	O1 ↑	O2 ↑	O3 ↑	avg ↑	O4 ↑	O5 ↑	O6 ↑	avg ↑	O7 ↑	O8 ↑	avg ↑
SeqLoRA	77.92	78.05	78.10	78.02	61.10	61.33	61.15	61.19	29.44	31.43	30.43
IncLoRA	82.20	82.40	82.92	82.50	65.20	65.50	65.40	65.36	33.44	35.39	34.42
O-LoRA	83.30	83.20	83.35	83.28	67.18	67.08	67.35	67.20	34.85	34.66	34.76
Gain-LoRA	83.27	83.36	83.80	83.48	67.39	67.39	67.41	67.39	36.42	37.38	36.90
CorDA	83.30	83.30	83.15	83.25	67.06	67.12	67.13	67.10	30.01	30.66	30.33
SLAO	85.25	85.75	86.98	85.99	68.19	68.78	68.51	68.49	47.72	47.96	47.84
KnOTS	76.60	77.10	76.70	76.80	59.23	59.54	59.46	59.41	26.99	27.07	27.03
LoRA-LEGO	82.63	82.60	82.70	82.64	67.72	67.91	66.64	67.42	40.95	41.08	41.02
SOAR (ours)	**89.35**	**89.95**	**89.15**	**89.48**	**71.23**	**70.93**	**70.72**	**70.96**	**51.28**	**51.90**	**51.59**

(3) **SuperNI Benchmark** [26]. It includes an extensive collection of NLP tasks with expert-written instructions, covering dialogue generation, summarization, sentiment analysis, question answering, and information extraction. For the SuperNI Benchmark, we use 1,000 training instances and 100 for validation/testing per task. To analyze the efficacy of our approach, as in previous work, we train models on eight different orders, which are task permutations. More details on task orders are detailed in Appendix (Table 15).

Baselines. We compare SOAR against different types of baselines: (1) Continual learning baselines: SLAO [14], current benchmark for single LoRA-based continual learning method; SeqLoRA: Sequentially fine-tunes a single LoRA on multiple tasks without any modification; IncLoRA: Incrementally adds a new LoRA per task while freezing previous LoRAs; O-LoRA [25]; GainLoRA [10]; CorDA [29]; (2) LoRA merging baselines: LoRA-LEGO [32]; KnOTS [19].

Evaluation. To evaluate the learning performance, we consider the following two metrics, i.e., accuracy (ACC), which measures the final accuracy averaged over all tasks, and backward transfer (BWT), which measures the average accuracy change of each task after learning new tasks:

$\mathrm{ACC} = \frac{1}{T}\sum_{i=1}^{T} A_{T,i}, \mathrm{BWT} = \frac{1}{T-1}\sum_{i=1}^{T-1}(A_{T,i} - A_{i,i})$, where $A_{i,j}$ represents the testing accuracy of task j after learning task i.

Experimental Setup. All the experiments in the Table 1 are performed on 1 H100, 80GB GPU with LoRA rank $\rightarrow$ 32, alpha $\rightarrow$ 64, learning rate $\rightarrow$ 5 e^{-5}. The τ to control the sensitivity of the gate in SOAR is set to 0.5 for the experiments. The targeted modules for LoRA training are q, k, v, and o. All the experiments are run 3 times on seeds $\{42, 43, 44\}$.

5.1 Discussion

LoRA-based continual learning. As shown in Table 1, SOAR consistently outperforms all baselines across three benchmarks on Qwen3-8B. SeqLoRA performs the worst because unconstrained sequential fine-tuning of a single shared adapter leads to severe catastrophic forgetting. IncLoRA improves by freezing previous adapters and introducing basic subspace isolation, but its memory cost grows linearly with tasks, and its separation strategy remains naive. O-LoRA provides modest gains through orthogonality constraints on new updates, yet applies them globally and does not exploit the asymmetric roles of LoRA factors. Gain-LoRA further improves via task-specific gating over shared components, but still treats $\mathbf{A}$ and $\mathbf{B}$ symmetrically, making it susceptible to coordinate drift during sequential merging. CorDA is competitive on the Standard CL and Large Number of Tasks benchmarks, but drops sharply on SuperNI, likely because nullspace selection from pretrained activations is less reliable under diverse instruction-following tasks. SLAO is the strongest LoRA-based baseline and benefits from single-adapter merging, but its factors are not aligned in a geometrically consistent coordinate system, limiting faithfulness of its consolidation step.

LoRA merging baselines. From Table 1 we infer, KnOTS underperforms surprisingly relative to continual learning methods across all three benchmarks, since it is designed for offline model merging where all task adapters are available simultaneously and does not account for the sequential, data-free nature of the continual setting. LoRA-LEGO performs more competitively on the Standard CL

Table 2: (%) Performance comparison of different initialization and merging on Qwen3-8B.

	Standard CL Benchmark				Large Number of Tasks				SuperNI Benchmark		
Method	O1 ↑	O2 ↑	O3 ↑	avg ↑	O4 ↑	O5 ↑	O6 ↑	avg ↑	O7 ↑	O8 ↑	avg ↑
QR+lambda	85.25	85.75	86.98	85.99	68.19	68.78	68.507	68.49	47.72	47.96	47.84
Stiefel+lambda	86.10	85.80	86.75	86.21	69.05	69.38	68.96	69.13	48.35	48.43	48.40
QR+SOAR Merging	87.80	87.50	87.70	87.67	70.17	70.01	69.88	70.02	48.91	49.68	49.29
SOAR	**89.35**	**89.95**	**89.15**	**89.48**	**71.23**	**70.93**	**70.72**	**70.96**	**51.28**	**51.90**	**51.59**

and Large Number of Tasks benchmarks by treating LoRA modules as composable blocks, but treats all rank directions equally and lacks any forgetting-aware consolidation, which limits its performance on the SuperNI benchmark where task diversity is highest. SOAR surpasses all baselines by significant margins across all settings, reaching 89.48, 70.96, and 51.59 average accuracy on the three benchmarks, respectively, validating that jointly enforcing Stiefel-constrained input subspaces and Procrustes-aligned rank-wise merging provides a principled and effective foundation for single-adapter continual LoRA learning. We report the mean and std for experiments in Table 1 in Appendix Figure 3.

Ablation: Initialization and Merging Strategy. Table 2 gives an ablation study over the two main components in SOAR: Stiefel initialization of $\mathbf{A}$, and Procrustes-aligned adaptive merging of $\mathbf{B}$. We evaluate four settings below: λ based merging (same as SLAO [14]) is $\mathbf{B}_{\text{merge}}$ = $\mathbf{B}_{\text{old}}$ + $(1/\lambda)$ * ($\mathbf{B}_{\text{new}}$ - $\mathbf{B}_{\text{old}}$), where λ is the square root of the task index. Here we show results of SOAR across three continual learning benchmarks on Qwen3-8B. First, replacing QR-based initialization with Stiefel optimization (*Stiefel+*λ) provides a slight improvement over *QR+*λ (e.g., 86.21 vs. 85.99 on the Standard CL Benchmark). This suggests that enforcing $\mathbf{A}^\top \in \mathrm{St}(n, r)$, equivalently $\mathbf{A}\mathbf{A}^\top = \mathbf{I}_r$, during training helps reduce the factorization ambiguity from $GL(r)$ to $\mathcal{O}(r)$, as shown in Lemma 3.1. However, without an alignment-aware merging rule for $\mathbf{B}$, the gain remains limited. A larger benefit is observed for *QR+SOAR Merging*, showing that rotating $\mathbf{B}_{\text{merge}}^{(t-1)}$ into the frame of $\mathbf{B}_{\text{ft}}^{(t)}$ with $\mathbf{R}^* = \mathbf{U}\mathbf{V}^\top$, then gatedly preserving only those rank directions via γ_i is indeed much more effective than uniform linear blending. Crucially, the quality of $\mathbf{R}^*$ depends on the geometric conditioning of the input subspace: post-hoc QR projection onto Stiefel breaks the $\mathbf{A}$–$\mathbf{B}$ coupling established during training, causing alignment errors to compound across tasks (Appendix C). By enforcing the Stiefel constraint *during* training, SOAR ensures a well-conditioned $\mathbf{R}^*$ and geometrically meaningful scores c_i and δ_i, explaining the consistent margin over all ablated variants and confirming that Stiefel initialization is a *precondition* for well-posed Procrustes merging, not merely a complementary component.

Ablation: Effect of Procrustes and Per-rank Gate-based merging. Figure 4 reports the Backward Transfer (BWT) for three $\mathbf{B}$ merging strategies with $\mathbf{A}$ fixed on the Stiefel manifold, across task order, across all three settings, the trend is consistent and mirrors the theoretical predictions: λ-merge exhibits the worst BWT (most negative), Procrustes+λ improves over it, and SOAR achieves the least forgetting throughout. The benefit of Procrustes alignment over plain λ-merge is visible even with a uniform interpolation coefficient: on (O7–O8), alignment alone recovers 0.52 BWT points ($-5.10 \rightarrow -4.58$), confirming that coordinate misalignment between $\mathbf{B}_{\text{merge}}^{(t-1)}$ and $\mathbf{B}_{\text{ft}}^{(t)}$ introduces spurious interference that is independent of task conflict; Procrustes alignment removes this confound and allows the uniform merge to operate on geometrically comparable factors. However, the gap between Procrustes+λ and SOAR is substantially larger and grows with task order (0.35 on O1–O3, 1.60 on O4–O6, 1.67 on O7–O8), revealing the fundamental limitation of uniform merging: as task sequences grow longer and more diverse, applying a single global coefficient to all rank directions increasingly over-retains directions that have been genuinely repurposed by the new task while under-protecting directions that are stable and compatible.

Performance on different models of varying sizes. Table 3 shows SOAR consistently outperforms SLAO across all model families and sizes, with gains on SuperNI most pronounced for larger models ($+3.07$ and $+3.75$ for Llama-3.1-8B and Qwen3-8B respectively), where task diversity makes faithful consolidation critical. The consistency across architectures suggests the benefits of

Table 3: Performance comparison of different model sizes and families.

		Standard CL Benchmark				Large Number of Tasks				SuperNI Benchmark		
Model	Method	O1 ↑	O2 ↑	O3 ↑	avg ↑	O4 ↑	O5 ↑	O6 ↑	avg ↑	O7 ↑	O8 ↑	avg ↑
Llama-3.2-3B	SLAO	82.12	81.48	83.93	82.51	57.31	62.13	59.42	59.92	46.07	45.52	45.79
	SOAR	**85.88**	**86.45**	**87.13**	**86.48**	**59.71**	**63.31**	**65.93**	**62.98**	**47.04**	**48.87**	**47.96**
Llama-3.1-8B	SLAO	87.08	87.53	87.35	87.32	65.25	68.76	65.80	66.60	47.50	48.18	47.84
	SOAR	**89.35**	**90.10**	**89.70**	**89.72**	**68.42**	**69.88**	**68.49**	**68.90**	**49.98**	**51.83**	**50.91**
Qwen3-8B	SLAO	85.25	85.75	86.98	85.99	68.19	68.78	68.51	68.49	47.72	47.96	47.84
	SOAR	**89.35**	**89.95**	**89.15**	**89.48**	**71.23**	**70.93**	**70.72**	**70.96**	**51.28**	**51.90**	**51.59**

Stiefel-constrained optimization and Procrustes-aligned merging scale with model expressivity rather than being architecture-specific. More Detailed comparisons are in the Appendix Tables 9 and 8.

Table 4: Additional Ablation on Gating

Variant	γ_i	**O1**	**O4**	**O7**	**Avg** (Δ)
Random gate	$\sim \mathcal{U}[0,1]$	80.10	62.55	41.59	61.41 (-9.21)
Cosine-only	$\max(c_i, 0)$	86.05	65.33	44.41	65.26 (-5.36)
Residual-only	$\exp(-\delta_i/\tau)$	86.65	68.41	49.22	68.09 (-2.53)
SOAR (full)	$\max(c_i, 0)\exp(-\delta_i/\tau)$	**89.35**	**71.23**	**51.28**	**70.62**

Additional ablations on gating: All variants share the identical merge skeleton and Procrustes alignment, differing only in γ_i, so the comparison isolates the retention rule itself.

The ablation in Table 4 confirms that the retention gate is a key component of SOAR. A random rank-wise gate performs substantially worse, demonstrating that continual learning performance depends on preserving the appropriate rank directions rather than simply retaining an equivalent amount of prior information. Furthermore, neither of the two signals used in the gate is sufficient on its own. Both the cosine-only and residual-only variants consistently underperform the full formulation across all benchmark suites, supporting the multiplicative design introduced in Eq. (7).

The cosine similarity captures directional agreement between old and new factors, while the residual term measures how much a rank direction has actually changed during adaptation. Relying solely on directional similarity may preserve directions whose magnitudes have already been significantly rewritten by the new task, whereas relying solely on residual magnitude ignores directional consistency. By combining both signals multiplicatively, the gate suppresses retention whenever either criterion indicates that a rank direction is no longer reliable, resulting in consistently stronger continual learning performance.

Table 5: BWT scores across task orders on different models.

Method	**A (SCL)**	**B (SCL)**	**C (SCL)**	**A (LNT)**	**B (LNT)**	**C (LNT)**	**A (SNI)**	**B (SNI)**	**C (SNI)**
SeqLoRA	-12.40	-12.85	-13.62	-14.41	-14.78	-15.81	-17.57	-18.04	-19.42
IncLoRA	-8.02	-8.74	-9.38	-8.41	-9.12	-10.21	-14.98	-15.46	-16.84
O-LoRA	-7.10	-7.58	-8.16	-5.66	-6.18	-6.91	-13.78	-14.02	-15.18
Gain-LoRA	-7.35	-7.96	-8.42	-4.54	-5.08	-5.84	-11.39	-11.87	-12.73
CoRDA	-7.02	-7.46	-8.37	-7.61	-8.04	-8.92	-15.56	-16.03	-17.44
SLAO	-4.12	-4.38	-5.18	-3.53	-3.91	-4.67	-6.04	-6.28	-6.21
KnoTS	-12.65	-13.08	-13.94	-11.55	-11.97	-12.81	-22.66	-23.21	-26.85
LoRA-LEGO	-7.98	-8.41	-9.05	-7.51	-7.96	-8.73	-9.89	-10.54	-11.62
SOAR (ours)	**-1.25**	**-1.41**	**-1.86**	**-1.68**	**-1.92**	**-2.31**	**-3.43**	**-3.71**	**-4.26**

Backward transfer analysis Table 5 reports Backward Transfer (BWT) analysis for all three LLM families on all task orders. Here, **A** is Qwen3-8B, **B** is Llama-3.1-8B, and **C** is Llama-3.2-3B, SCL is the standard continual learning task; LNT is large number of tasks; and SNI is the SuperNI benchmark dataset, where less negative values indicate better retention of prior knowledge. Three patterns hold consistently. First, the ordering of methods is stable across all three backbones and all nine orders: SeqLoRA suffers the most severe forgetting, confirming that unconstrained sequential

fine-tuning of a single adapter catastrophically overwrites prior tasks; memory-isolation methods such as IncLoRA and CorDA improve on it but still forget substantially, as they do not explicitly consolidate knowledge across tasks; O-LoRA and Gain-LoRA reduce forgetting further through orthogonality constraints and gating; and SLAO is the strongest baseline throughout, benefiting from continual merging within a single adapter. The offline merging methods behave as expected in this sequential, data-free setting: KnOTS degrades sharply, while LoRA-LEGO remains competitive with the continual-learning baselines. Second, forgetting grows with benchmark difficulty for every method, from the Standard CL benchmark through the Large Number of Tasks setting to SuperNI, where longer and more heterogeneous task sequences leave the most room for interference. Third, SOAR attains the lowest forgetting in every column, reducing BWT relative to SLAO by 52–70% on the Standard CL and Large Number of Tasks benchmarks and by 31–43% on SuperNI, and by nearly an order of magnitude relative to SeqLoRA. Crucially, SOAR's margin is preserved across model families and scales, indicating that Procrustes-aligned, rank-wise gated merging provides more faithful knowledge consolidation than either orthogonality-based or uniform merging, independently of the backbone.

Table 6: Total compute time taken

Method	**Time Taken (seconds)**
KnoTS	1300.0
LoRA-LEGO	1472.0
GainLoRA	1299.5
IncLoRA	1219.0
SLAO	1254.0
OLoRA	1452.0
SOAR (ours)	1582.0

Compute Analysis Table 11 reports total continual learning runtime on the O1 task order (4 tasks, batch size 2, and 1,250 steps per task) using Qwen3-8B on a single H100 under identical configurations. All methods fall within 1,200–1,600s, indicating that consolidation strategy has a secondary impact compared with the dominant $O(bnm)$ forward and backward passes. SOAR takes 1,582s, a 15.6% overhead over vanilla sequential fine-tuning (1,368s). This overhead comes primarily from per-step Riemannian gradient projection and polar retraction on $\mathbf{A}$, with a negligible per-task Procrustes SVD on an $(r \times r)$ matrix. Both scale as $O(r^3)$ and are independent of sequence length and batch size; thus, their relative cost diminishes with increasing model size or token budget. Other methods with additional per-step operations, O-LoRA (1,452s) and LoRA-LEGO (1,472s), have comparable runtimes, while lighter consolidation methods such as IncLoRA (1,219s), SLAO (1,254s), GainLoRA (1,299.5s), and KnOTS (1,300s) are marginally cheaper. Overall, SOAR introduces a modest, rank-dependent overhead that does not grow with sequence length.

Additional Experiments. To further validate SOAR, we conduct additional experiments reported in the Appendix. superior knowledge retention throughout the task sequence. (1) **Ablation on Merging $\mathbf{A}$** D.5: Here we conduct experiments where we also merge $\mathbf{A}$ and we report the average change in $\mathbf{A}$ on varying tasks. (2) **Gating Analysis** D.3: We also detail how the per-rank retention gating value varies across different attention.

6 Conclusion

We introduced SOAR, a geometrically based framework for continual LoRA learning that combines task updates into one adapter. By placing $\mathbf{A}$ on the Stiefel manifold and aligning $\mathbf{B}$ using Orthogonal Procrustes, SOAR minimizes coordinate confusion and allows for better rank-wise merging. In various continual learning tests, SOAR consistently enhances stability and flexibility while keeping adapter memory constant. One limitation is that SOAR retains information at the level of individual rank directions, which may not fully reflect knowledge spread across larger subspaces. Additionally, since SOAR does not use replay or saved data, information that gets completely overwritten during adaptation cannot be retrieved. Future efforts can build on this framework to achieve subspace-level consolidation and simpler memory solutions.

References

[1] Zhan Fa, Yue Duan, Jian Zhang, Lei Qi, Wanqi Yang, and Yinghuan Shi. Decomposing and composing: Towards efficient vision-language continual learning via rank-1 expert pool in a single lora. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 40, pages 20985–20993, 2026.

[2] Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, et al. The Llama 3 Herd of Models, 2024.

[3] Xiaolong Han, Ferrante Neri, Zijian Jiang, Fang Wu, Yanfang Ye, Lu Yin, and Zehong Wang. W2t: Lora weights already know what they can do, 2026.

[4] Yongchang Hao, Yanshuai Cao, and Lili Mou. Flora: Low-rank adapters are secretly gradient compressors. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pages 17554–17571. PMLR, 21–27 Jul 2024.

[5] Gabriel Ilharco, Marco Túlio Ribeiro, Mitchell Wortsman, Ludwig Schmidt, Hannaneh Hajishirzi, and Ali Farhadi. Editing models with task arithmetic. In *The Eleventh International Conference on Learning Representations, ICLR 2023*, 2023.

[6] 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. Overcoming catastrophic forgetting in neural networks. *Proceedings of the National Academy of Sciences*, 114(13):3521–3526, 2017.

[7] Shashi Kumar, Yacouba Kaloga, John Mitros, Petr Motlicek, and Ina Kodrasi. Latent space factorization in lora. In *Advances in Neural Information Processing Systems*, volume 38, Main Conference, pages 88111–88138. Curran Associates, Inc., 2025.

[8] Xingyu Li, Bo Tang, and Haifeng Li. AdaER: An adaptive experience replay approach for continual lifelong learning. *Neurocomputing*, 572:127204, 2024.

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

[10] Yan-Shuo Liang, Jia-Rui Chen, and Wu-Jun Li. Gated integration of low-rank adaptation for continual learning of large language models. In *Advances in Neural Information Processing Systems*, volume 38, Main Conference, pages 68659–68689. Curran Associates, Inc., 2025.

[11] Yan-Shuo Liang and Wu-Jun Li. InfLoRA: Interference-free low-rank adaptation for continual learning. In *2024 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 23638–23647, 2024.

[12] David Lopez-Paz and Marc' Aurelio Ranzato. Gradient episodic memory for continual learning. In *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc., 2017.

[13] Arun Mallya and Svetlana Lazebnik. PackNet: Adding multiple tasks to a single network by iterative pruning. In *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 7765–7773, 2018.

[14] Fuli Qiao and Mehrdad Mahdavi. Merge before forget: A single lora continual learning via continual merging. In *International Conference on Learning Representations*, volume 2026, pages 8133–8159, 2026.

[15] David Rolnick, Arun Ahuja, Jonathan Schwarz, Timothy Lillicrap, and Gregory Wayne. Experience replay for continual learning. In *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 2019.

[16] Peter H. Schönemann. A generalized solution of the orthogonal procrustes problem. *Psychometrika*, 31(1):1–10, 1966.

[17] Joan Serra, Didac Suris, Marius Miron, and Alexandros Karatzoglou. Overcoming catastrophic forgetting with hard attention to the task. In *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pages 4548–4557. PMLR, 10–15 Jul 2018.

[18] Haizhou Shi, Zihao Xu, Hengyi Wang, Weiyi Qin, Wenyuan Wang, Yibin Wang, Zifeng Wang, Sayna Ebrahimi, and Hao Wang. Continual learning of large language models: A comprehensive survey. *ACM Comput. Surv.*, 58(5), November 2025.

[19] George Stoica, Pratik Ramesh, Boglarka Ecsedi, Leshem Choshen, and Judy Hoffman. Model merging with svd to tie the knots. In *International Conference on Learning Representations*, volume 2025, pages 4501–4519, 2025.

[20] Youbang Sun, Zitao Li, Yaliang Li, and Bolin Ding. Improving LoRA in privacy-preserving federated learning. In B. Kim, Y. Yue, S. Chaudhuri, K. Fragkiadaki, M. Khan, and Y. Sun, editors, *International Conference on Learning Representations*, volume 2024, pages 17978–17994, 2024.

[21] Anke Tang, Enneng Yang, Li Shen, Yong Luo, Han Hu, Bo Du, and Dacheng Tao. Merging models on the fly without retraining: A sequential approach to scalable continual model merging, 2025.

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

[23] Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R. Bowman. GLUE: A multi-task benchmark and analysis platform for natural language understanding. In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*, 2019.

[24] Liyuan Wang, Xingxing Zhang, Hang Su, and Jun Zhu. A comprehensive survey of continual learning: Theory, method and application. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 46(8):5362–5383, 2024.

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

[26] Yizhong Wang, Swaroop Mishra, Pegah Alipoormolabashi, Yeganeh Kordi, Amirreza Mirzaei, Atharva Naik, Arjun Ashok, Arut Selvan Dhanasekaran, Anjana Arunkumar, David Stap, Eshaan Pathak, Giannis Karamanolakis, Haizhi Lai, Ishan Purohit, Ishani Mondal, Jacob Anderson, Kirby Kuznia, Krima Doshi, Kuntal Kumar Pal, Maitreya Patel, Mehrad Moradshahi, Mihir Parmar, Mirali Purohit, Neeraj Varshney, Phani Rohitha Kaza, Pulkit Verma, Ravsehaj Singh Puri, Rushang Karia, Savan Doshi, Shailaja Keyur Sampat, Siddhartha Mishra, Sujan Reddy A, Sumanta Patro, Tanay Dixit, and Xudong Shen. Super-NaturalInstructions: Generalization via declarative instructions on 1600+ NLP tasks. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 5085–5109, Abu Dhabi, United Arab Emirates, December 2022. Association for Computational Linguistics.

[27] Prateek Yadav, Derek Tam, Leshem Choshen, Colin Raffel, and Mohit Bansal. TIES-Merging: Resolving interference when merging models. In *Advances in Neural Information Processing Systems*, volume 36, pages 7093–7115. Curran Associates, Inc., 2023.

[28] An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, Chujie Zheng, Dayiheng Liu, Fan Zhou, Fei Huang, Feng Hu, Hao Ge, Haoran Wei, Huan Lin, Jialong Tang, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Yang, Jiaxi Yang, Jing Zhou, Jingren Zhou, Junyang Lin, Kai Dang, Keqin Bao, Kexin Yang, Le Yu, Lianghao Deng, Mei Li, Mingfeng Xue, Mingze Li, Pei Zhang, Peng Wang, Qin Zhu, Rui Men, Ruize Gao, Shixuan Liu, Shuang Luo, Tianhao Li, Tianyi Tang, Wenbiao Yin,

Xingzhang Ren, Xinyu Wang, Xinyu Zhang, Xuancheng Ren, Yang Fan, Yang Su, Yichang Zhang, Yinger Zhang, Yu Wan, Yuqiong Liu, Zekun Wang, Zeyu Cui, Zhenru Zhang, Zhipeng Zhou, and Zihan Qiu. Qwen3 technical report, 2025.

[29] Yibo Yang, Xiaojie Li, Zhongzhu Zhou, Shuaiwen Leon Song, Jianlong Wu, Liqiang Nie, and Bernard Ghanem. Corda: Context-oriented decomposition adaptation of large language models for task-aware parameter-efficient fine-tuning. In *Advances in Neural Information Processing Systems*, volume 37, pages 71768–71791. Curran Associates, Inc., 2024.

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

[31] Weixiang Zhao, Shilong Wang, Yulin Hu, Yanyan Zhao, Bing Qin, Xuanyu Zhang, Qing Yang, Dongliang Xu, and Wanxiang Che. SAPT: A shared attention framework for parameter-efficient continual learning of large language models. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 11641–11661, Bangkok, Thailand, August 2024. Association for Computational Linguistics.

[32] Ziyu Zhao, tao shen, Didi Zhu, Zexi Li, Jing Su, Xuwu Wang, and Fei Wu. Merging loras like playing lego: Pushing the modularity of lora to extremes through rank-wise clustering. In *International Conference on Learning Representations*, volume 2025, pages 72896–72913, 2025.

[33] Jiacheng Zhu, Kristjan Greenewald, Kimia Nadjahi, Haitz Sáez De Ocáriz Borde, Rickard Brüel Gabrielsson, Leshem Choshen, Marzyeh Ghassemi, Mikhail Yurochkin, and Justin Solomon. Asymmetry in low-rank adapters of foundation models. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pages 62369–62385. PMLR, 21–27 Jul 2024.

A Related Works on General Continual Learning

Continual learning has been a major interest in the deep learning community, aiming for lifelong learning and model evolution. Previous approaches have explored various methods (see [24] and references therein) to tackle catastrophic forgetting and improve knowledge sharing: To tackle two major problems in continual learning tasks: (a) catastrophic forgetting, where model performance degrades on previous tasks when learning new tasks, and (b) knowledge sharing, where knowledge acquired from previous tasks is leveraged to improve learning on subsequent tasks. Some earlier methods explored are 1) Memory replay-based methods, such as: CLEAR [15] using a replay buffer with mixed on/off-policy learning and behavioral cloning, which enables reinforcement learning agents to learn new tasks while significantly reducing catastrophic forgetting. AdaER [8] improves replay through selective memory recall (conflict-based sampling) and entropy-based buffer management. GEM [12] stores task-specific gradients and projects new gradients to avoid interference with previous knowledge. 2) Regularization-based methods: EWC [6] penalizes changes to weights important for past tasks using a Fisher information–based quadratic constraint. LwF [9] uses distillation loss to preserve old task predictions while learning new tasks. 3) Architecture-based methods: Packnet [13] iteratively prunes and freezes weights to allocate capacity for new tasks. HAT [17] learns task-specific masks to gate which neurons are used.

B Optimization on the Stiefel Manifold

To enforce the orthogonality constraint $\mathbf{A}\mathbf{A}^\top = \mathbf{I}_r$, we perform optimization on the row-Stiefel manifold

$$\mathrm{St}(r, n) = \left\{ \mathbf{A} \in \mathbb{R}^{r \times n} : \mathbf{A}\mathbf{A}^\top = \mathbf{I}_r \right\}.$$

Given the Euclidean gradient $\mathbf{G}_\mathbf{A} = \nabla_\mathbf{A} \mathcal{L}$, the tangent space at $\mathbf{A}$ is

$$T_\mathbf{A}\mathrm{St}(r, n) = \left\{ \mathbf{\Xi} \in \mathbb{R}^{r \times n} : \mathbf{\Xi}\mathbf{A}^\top + \mathbf{A}\mathbf{\Xi}^\top = \mathbf{0} \right\}.$$

Table 7: Effect of different ranks on SOAR using Qwen3-8B.

	Standard CL Benchmark				Large Number of Tasks				SuperNI Benchmark		
Method	O1 ↑	O2 ↑	O3 ↑	avg ↑	O4 ↑	O5 ↑	O6 ↑	avg ↑	O7 ↑	O8 ↑	avg ↑
$r \to 8$	88.82	88.88	88.38	88.69	69.34	69.07	69.70	69.37	49.50	50.09	49.79
$r \to 16$	88.78	88.87	88.60	88.72	70.20	70.13	70.19	70.17	49.58	50.83	50.21
$r \to 32$	**89.35**	**89.95**	**89.15**	**89.48**	71.23	**70.93**	70.72	**70.96**	51.28	**51.90**	**51.59**
$r \to 64$	88.60	88.78	89.13	88.83	**71.40**	70.44	**70.80**	70.88	**51.60**	51.42	51.51

The Riemannian gradient is obtained by projecting the Euclidean gradient onto the tangent space:

$$\mathrm{grad}_{\mathbf{A}}\mathcal{L} = \mathbf{G}_{\mathbf{A}} - \mathrm{sym}(\mathbf{G}_{\mathbf{A}}\mathbf{A}^\top)\mathbf{A}, \quad \mathrm{sym}(\mathbf{M}) = \tfrac{1}{2}(\mathbf{M} + \mathbf{M}^\top).$$

After taking a step along a tangent direction $\boldsymbol{\Xi}$, we retract back to the manifold. We use the polar retraction:

$$\mathrm{Retr}_{\mathbf{A}}(\boldsymbol{\Xi}) = \left(\mathbf{I}_r + \boldsymbol{\Xi}\boldsymbol{\Xi}^\top\right)^{-1/2}(\mathbf{A} + \boldsymbol{\Xi}),$$

which ensures that the updated $\mathbf{A}$ remains orthonormal.

In practice, this procedure maintains $\mathbf{A}\mathbf{A}^\top = \mathbf{I}_r$ throughout training, rather than encouraging orthogonality via soft regularization. This is critical for our method, as it guarantees that the latent coordinate system remains well-conditioned and comparable across tasks, enabling the alignment procedure described in the main text.

C Impact of constraining A on Stiefel Manifold

Effect of post-hoc projection on coupled factors. In LoRA, the factors $\mathbf{A}$ and $\mathbf{B}$ are jointly optimized, and the learned pair $(\mathbf{B}, \mathbf{A})$ represents a particular element within an equivalence class under transformations of the form $(\mathbf{B}, \mathbf{A}) \mapsto (\mathbf{B}\mathbf{R}^\top, \mathbf{R}^{-1}\mathbf{A})$ for $\mathbf{R} \in GL(r)$. As a result, the optimizer implicitly selects a coordinate system in which $\mathbf{A}$ and $\mathbf{B}$ are coupled.

If $\mathbf{A}$ is subsequently projected onto the Stiefel manifold to enforce $\mathbf{A}\mathbf{A}^\top = \mathbf{I}_r$ (by, for instance, QR-decomposition), this operation modifies the coordinate system of $\mathbf{A}$. However, unless a corresponding transformation is applied to $\mathbf{B}$, the original coupling between the two factors is broken, and the resulting pair no longer represents the same low-rank update.

Even if one attempts to compensate by transforming $\mathbf{B}$ after projecting $\mathbf{A}$, the required transformation generally lies in $GL(r)$, reflecting the unconstrained symmetry of the LoRA factorization during training. However, the subsequent alignment step is restricted to orthogonal transformations, as Procrustes solves an optimization over $\mathcal{O}(r)$. As a result, it can only recover the closest orthogonal approximation to the true transformation, but cannot capture general scaling or shearing components. This mismatch leads to residual misalignment between factors, as the coordinate discrepancy introduced during training is only partially corrected. In a continual setting, such approximation errors can accumulate across tasks, degrading the consistency of rank directions and affecting subsequent merging.

By enforcing the Stiefel constraint throughout training, we ensure that $\mathbf{A}$ and $\mathbf{B}$ are learned within a restricted equivalence class (up to orthogonal transformations), preserving their coupling and maintaining a consistent coordinate system across tasks.

D Further Experimental Analysis

D.1 Effect of LoRA ranks:

Table 7 looks at how LoRA rank affects SOAR's performance on Qwen3-8B. Performance improves steadily from $r = 8$ to $r = 32$ across all three benchmarks. The Standard CL and SuperNI benchmarks show the most noticeable trends ($88.69 - 89.48$ and $49.79 - 51.59$, respectively). This indicates that a larger rank gives more expressive low-rank directions for the Procrustes alignment and per-rank gating to work with. After $r = 32$, the gains level off or even drop slightly. For example,

Table 8: (%) Performance comparison on three CL benchmark datasets using LLaMA-3.2-3B-Instruct, across different task order, where O_i represents i^{th} task order.

	Standard CL Benchmark				Large Number of Tasks				SuperNI Benchmark		
Method	O1 ↑	O2 ↑	O3 ↑	avg ↑	O4 ↑	O5 ↑	O6 ↑	avg ↑	O7 ↑	O8 ↑	avg ↑
SeqLoRA	61.21	61.31	61.50	61.34	49.30	50.10	49.10	49.50	31.00	31.85	31.43
IncLoRA	66.23	66.40	66.44	66.35	53.41	53.10	53.15	53.22	32.64	32.75	32.70
O-LoRA	68.08	68.01	68.16	68.03	56.14	56.25	56.67	56.35	33.07	33.51	33.29
Gain-LoRA	68.45	68.80	68.70	68.65	53.70	53.20	54.10	53.67	33.54	33.70	33.62
CorDA	68.08	68.05	68.10	68.07	54.53	55.20	55.16	55.3	28.00	29.02	28.51
SLAO	82.12	81.48	83.93	82.51	57.31	62.13	59.42	59.62	46.07	45.52	45.80
KnOTS	59.50	61.50	59.38	60.13	47.07	46.86	47.21	47.04	18.93	19.28	19.11
LoRA-LEGO	80.48	80.40	80.35	80.41	54.37	54.63	54.17	54.39	39.54	39.20	39.37
SOAR (ours)	**85.88**	**86.45**	**87.13**	**86.48**	**59.71**	**63.31**	**65.93**	**62.98**	**47.04**	**48.87**	**47.95**

$r = 64$ achieves similar performance on the Large Number of Tasks benchmark (70.88 vs. 70.96) but shows a small decrease on the Standard CL Benchmark (88.83 vs. 89.48). This may indicate mild overfitting or redundancy in the extra rank directions. Overall, it seems that $r = 32$ provides the best balance between subspace expressiveness and the geometric consistency required by the Stiefel constraint as the rank increases. Maintaining a well-conditioned orthonormal basis becomes more difficult, and the additional rank directions offer less coherent structure to the alignment process.

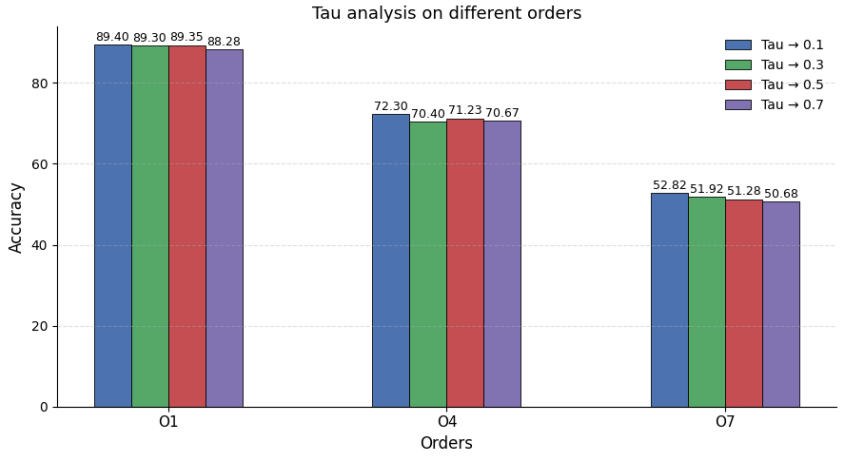


Figure 1: (%) Performance analysis across different τ values for three task orders.

D.2 Ablation on the impact of τ:

Figure 1 examines the sensitivity of SOAR to the temperature parameter τ, which controls the sensitivity of the retention gate $\gamma_i = \max(c_i, 0) \cdot \exp(-\delta_i/\tau)$ to magnitude drift δ_i. Across all three benchmarks, performance is remarkably stable for $\tau \in \{0.1, 0.3, 0.5\}$, with differences of at most 1.07 points, indicating that the gate is not critically sensitive to this hyperparameter in this range. At $\tau = 0.7$, a modest but consistent drop is observed across all orders, suggesting that too large a τ reduces the gate's sensitivity to drift; the exponential factor $\exp(-\delta_i/\tau)$ flattens toward 1 for all rank directions, causing the gate to over-retain directions that the new task has substantially overwritten. We fix $\tau = 0.5$ as the default, as it achieves the best balance between drift sensitivity and stability, though the method remains competitive across the full range tested.

D.3 Detailed per-rank gating analysis:

Figure 2 reports the rank-wise retention gate $\gamma_i = \max(c_i, 0) \cdot \exp(-\delta_i/\tau)$, averaged across all transformer layers, for all four projection matrices (`Q_proj`, `K_proj`, `V_proj`, `O_proj`) under same-task and different-task transition regimes. Two consistent patterns emerge across all projection

Table 9: (%) Performance comparison on three CL benchmark datasets using LLaMA-3.1-8B-Instruct, across different task order, where O_i represents i^{th} task order.

	Standard CL Benchmark				Large Number of Tasks				SuperNI Benchmark		
Method	O1 ↑	O2 ↑	O3 ↑	avg ↑	O4 ↑	O5 ↑	O6 ↑	avg ↑	O7 ↑	O8 ↑	avg ↑
SeqLoRA	77.3	76.75	77.9	77.32	61.38	60.4	61.76	61.18	29.43	30.3	29.87
IncLoRA	78.34	78.11	77.8	78.08	63.45	64.11	63.76	63.77	31.34	32.44	31.89
O-LoRA	79.82	80.11	79.21	79.71	66.28	65.16	65.67	65.70	34.83	36.11	35.47
Gain-LoRA	80.10	80.55	80.40	80.35	65.47	66.24	65.79	65.83	35.15	36.23	35.69
CorDA	79.87	79.98	79.82	79.89	65.08	64.65	65.67	65.13	31.98	31.05	31.52
SLAO	87.08	87.53	87.35	87.32	65.25	68.76	65.8	66.60	47.50	48.18	47.84
KnOTS	77.62	77.50	77.75	77.62	60.42	60.49	60.69	60.54	28.64	28.61	28.63
LoRA-LEGO	85.52	85.32	85.4	85.41	63.37	63.28	63.46	63.37	41.90	42.24	42.07
SOAR (ours)	**89.35**	**90.10**	**89.70**	**89.72**	**68.42**	**69.88**	**68.49**	**68.93**	**49.98**	**51.83**	**50.90**

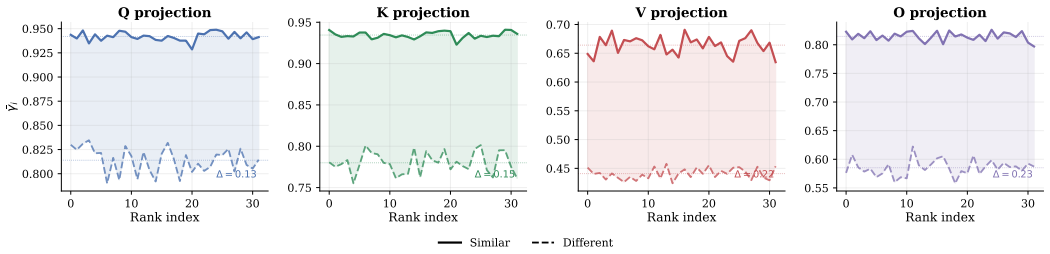


Figure 2: γ variance on similar vs dissimilar task, averaged across layers for rank 32

types. First, the same-task γ_i (blue) is uniformly and substantially higher than the different-task γ_i (orange) across all 32 rank directions, confirming that the gate correctly discriminates task-relatedness: when consecutive tasks are similar, the aligned old directions $\tilde{\mathbf{b}}_i^{\text{old}}$ exhibit high cosine agreement c_i with $\mathbf{b}_i^{\text{ft}}$ and small forgetting score δ_i, driving γ_i toward 1 and encouraging strong retention of prior knowledge. Conversely, under dissimilar-task transitions, γ_i is lower, which means even after Procrustes alignment, the old and new factors still differ significantly at the local scales, implying the need for local adjustment for proper merging. Second, the absolute level of γ_i varies meaningfully across projection types, revealing a structural hierarchy in how much prior information is retained. `Q_proj` and `K_proj` attain the highest same-task gates ($\gamma_i \approx 0.94$ and 0.93 respectively), suggesting that the query and key subspaces are the most geometrically stable across tasks and thus accumulate the most reusable structure. `O_proj` occupies an intermediate regime ($\gamma_i \approx 0.81$), while `V_proj` exhibits the lowest gates overall ($\gamma_i \approx 0.66$), indicating that value projections undergo the most task-specific rewriting and are therefore appropriately down-weighted during consolidation. Critically, this projection-wise differentiation is not manually specified but emerges automatically from the gate's construction, validating that the Procrustes-aligned coordinate frame provides a consistent basis in which c_i and δ_i reflect genuine task-specific geometry rather than coordinate artifacts. Here, two Similar tasks considered are (amazon polarity, imdb movie reviews); dissimilar tasks consist of (xsum summarization, emotion classification).

Algorithm 1 SOAR: Stiefel Optimization and Aligned Rotation for Continual LoRA Learning

Require: Pre-trained weights $\mathbf{W}_0$, tasks $\{\mathcal{D}_t\}_{t=1}^{T}$, rank r, lr η, temperature τ
Ensure: $\mathbf{A}_{\text{merge}}^{(T)} \in \text{St}(n, r)$, $\mathbf{B}_{\text{merge}}^{(T)} \in \mathbb{R}^{m \times r}$

Init $\mathbf{A} \sim \text{St}(n, r)$, $\mathbf{B} \leftarrow \mathbf{0}$, $\mathbf{A}_{\text{merge}} \leftarrow \varnothing$, $\mathbf{B}_{\text{merge}} \leftarrow \varnothing$
for $t = 1, \ldots, T$ **do**
  $(\mathbf{A}, \mathbf{B}) \leftarrow \left(\mathbf{A}_{\text{merge}}^{(t-1)}, \mathbf{B}_{\text{merge}}^{(t-1)}\right)$ **if** $t > 1$, **else** $\left(\mathbf{A}^{(0)}, \mathbf{0}\right)$
  for each mini-batch $\mathcal{B} \sim \mathcal{D}_t$ **do** ▷ Riemannian Adam
    Update $\mathbf{B}$ via Euclidean Adam on $\nabla_{\mathbf{B}} \mathcal{L}_t$
    Update $\mathbf{A}$ via Riemannian gradient + polar retraction onto $\text{St}(n, r)$
  end for
  $\mathbf{A}_{\text{merge}}^{(t)} \leftarrow \mathbf{A}_{\text{ft}}^{(t)}$ ▷ Direct replacement
  if $t = 1$ **then**
    $\mathbf{B}_{\text{merge}}^{(1)} \leftarrow \mathbf{B}_{\text{ft}}^{(1)}$
  else ▷ Procrustes-aligned adaptive merge
    SVD: $\left(\mathbf{B}_{\text{ft}}^{(t)}\right)^{\top} \mathbf{B}_{\text{merge}}^{(t-1)} = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^{\top}, \quad \mathbf{R}^{*} = \mathbf{U}\mathbf{V}^{\top}$ ▷ Procrustes on $\mathbf{B}$
    $\widetilde{\mathbf{B}}_{\text{merge}}^{(t-1)} \leftarrow \mathbf{B}_{\text{merge}}^{(t-1)} \left(\mathbf{R}^{*}\right)^{\top}$ ▷ Align old $\mathbf{B}$ to new frame
    $c_i \leftarrow \left(\mathbf{b}_i^{\text{ft}}\right)^{\top} \tilde{\mathbf{b}}_i^{\text{old}} \in [-1, 1]$ for each i ▷ Per-rank cosine similarity
    $\delta_i \leftarrow \dfrac{\|\mathbf{b}_i^{\text{ft}} - \tilde{\mathbf{b}}_i^{\text{old}}\|^2}{\|\tilde{\mathbf{b}}_i^{\text{old}}\|^2 + \epsilon}$ for each i ▷ Per-rank forgetting score
    $\gamma_i \leftarrow \max(c_i, 0) \cdot \exp(-\delta_i / \tau) \in [0, 1]$ for each i ▷ Retention gate
    $\mathbf{B}_{\text{merge}}^{(t)} \leftarrow \mathbf{B}_{\text{ft}}^{(t)} + \left(\widetilde{\mathbf{B}}_{\text{merge}}^{(t-1)} - \mathbf{B}_{\text{ft}}^{(t)}\right) \text{diag}(\boldsymbol{\gamma})$ ▷ Adaptive additive correction
  end if
end for
return $\mathbf{A}_{\text{merge}}^{(T)}$, $\mathbf{B}_{\text{merge}}^{(T)}$

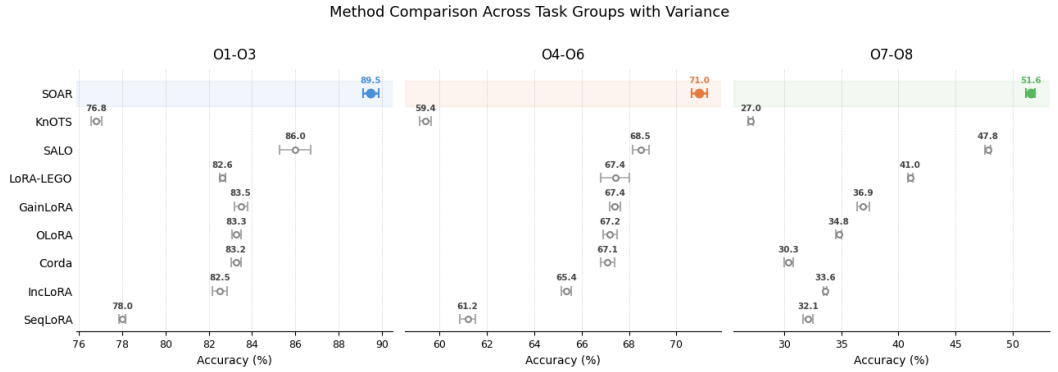


Figure 3: (%) Performance analysis averaged across the benchmark dataset.

D.4 Compute Overhead Analysis:

Table 11 reports total continual learning runtime for each method, measured on the O1 task order with 4 tasks, batch size 2, and 1,250 training steps per task on a single H100 GPU, using Qwen3-8B under an identical experimental configuration. All methods fall within a narrow band of roughly 1,200–1,600s, confirming that the choice of consolidation strategy has a second-order effect on cost relative to the dominant $O(bnm)$ forward and backward passes. SOAR is the most expensive at 1,582s, a 15.6% overhead over vanilla sequential fine-tuning (1,368s) and between 7.5% and 30% over the other baselines. This overhead is fully accounted for by two operations: the Riemannian gradient projection and polar retraction applied to $\mathbf{A}$ at each step, and a single per-task Procrustes SVD on an $(r \times r)$ matrix. Both scale as $O(r^3)$ in the rank and are independent of sequence length and batch size, so the relative cost decreases as model size or token budget grows. The per-step manifold operations account for the larger share, as they recur at every optimizer step, whereas the Procrustes

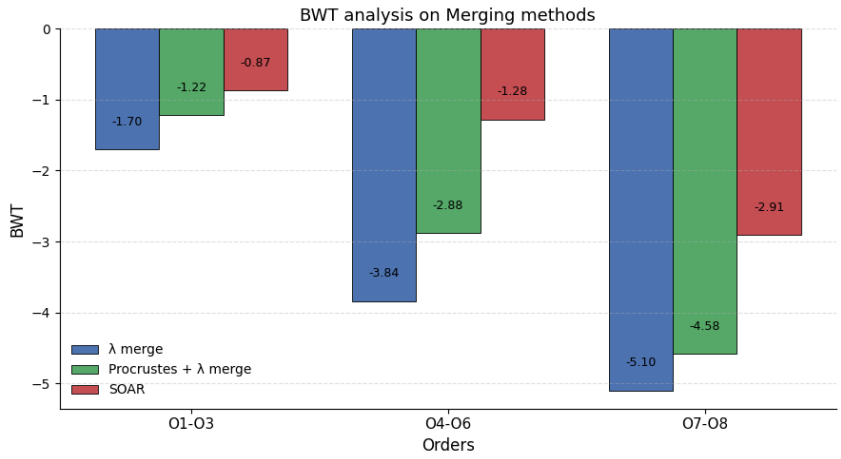


Figure 4: Effect of Procrustes and Per Rank Gating using Qwen3-8B across varying datasets.

Table 10: Descriptions of datasets used from standard continual learning benchmarks, GLUE, and SuperGLUE.

Dataset name	**Category**	**Task**	**Domain**	**Metric**
Amazon Polarity	CL Benchmark	Sentiment analysis	Amazon reviews	Accuracy
DBpedia	CL Benchmark	Topic classification	Wikipedia	Accuracy
Yahoo Answers	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	Paraphrase 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
BoolQ	SuperGLUE	Boolean question answering	Wikipedia	Accuracy
MultiRC	SuperGLUE	Question answering	Various	Accuracy
ReCoRD	SuperGLUE	Reading comprehension	News	Accuracy
IMDB	Sentiment	Sentiment analysis	Movie reviews	Accuracy

SVD is incurred once per task boundary and is negligible at $r{=}32$. Methods requiring additional per-step machinery are comparable: O-LoRA (1,452s) pays for its orthogonality penalty across all previous adapters, and LoRA-LEGO (1,472s) for rank-wise clustering at each merge. Methods with lighter-weight consolidation — IncLoRA (1,219s), SLAO (1,254s), GainLoRA (1,299.5s) and KnOTS (1,300s) — are marginally cheaper, with IncLoRA's savings offset by adapter memory that grows linearly in the number of tasks. SOAR's overhead is therefore modest, bounded, and predictable: it is fixed by the rank rather than the data, and does not accumulate with sequence length.

Table 11: Total compute time taken

Method	**Time Taken (seconds)**
KnoTS	1300.0
LoRA-LEGO	1472.0
GainLoRA	1299.5
IncLoRA	1219.0
SLAO	1254.0
OLoRA	1452.0
SOAR (ours)	1582.0

D.5 Ablation for update rule

To directly examine the assumption underlying our update rule, we first measure the principal-angle drift between the input subspaces spanned by consecutive $\mathbf{A}$ matrices. Principal angles isolate genuine subspace drift while remaining invariant to rotations within the subspace, making them an appropriate measure for this question. As shown in Table 13, the measured drift is small on O1 and O4, where tasks within each sequence are relatively similar, indicating that the learned input subspace is largely preserved across tasks. In contrast, O7 exhibits noticeably larger drift, consistent with its substantially greater task diversity.

To further validate this observation, in Table 12 we evaluate three alternative update strategies for $\mathbf{A}$. A fixed random orthonormal $\mathbf{A}$ performs the worst, demonstrating that a randomly chosen input subspace is far from the task-specific subspace learned during adaptation. Freezing $\mathbf{A}$ after the first task results in only a modest degradation O1 and O4, where the measured subspace drift is limited, but degrades much more noticeably on O7. This behavior supports the intuition that while previously learned input directions remain useful for related tasks, adapting $\mathbf{A}$ becomes increasingly important as the task distribution evolves and new input subspaces emerge.

We also investigate explicit merging of $\mathbf{A}$ using the same Procrustes alignment framework employed for $\mathbf{B}$. We observe improvements over direct replacement, with relatively modest gains on O1 and O4 and larger gains on O7, mirroring the measured increase in subspace drift. These results suggest that explicit $\mathbf{A}$ merging becomes more beneficial as tasks become more heterogeneous. However, this improvement comes at the cost of an additional interpolation hyperparameter and an extra Procrustes alignment and Stiefel retraction step, increasing the runtime by approximately 134.6 seconds (Qwen 3 8B for task order O1) per continual learning run. Our method is motivated by the asymmetry between the two LoRA factors [14], which suggests that the input factor $\mathbf{A}$ remains relatively stable across tasks. Our empirical analysis is consistent with this observation: as shown in Table 13, the measured principal-angle drift of $\mathbf{A}$ is small on O1 and O4, while a larger, though still moderate, drift is observed on the more heterogeneous O7 benchmark.

Table 12: Ablations on initialisation strategies for $\mathbf{A}$

A update rule	**O1**	**O4**	**O7**	**Average**
Random fixed $\mathbf{A}$	70.80	58.43	34.56	54.59
Freeze after first task	87.45	67.86	42.91	65.92
SOAR	89.35	71.23	51.28	70.62
SOAR + Procrustes $\mathbf{A}$-merge	**89.90**	**72.80**	**53.10**	**71.93**

Table 13: Measured principal-angle drift of $\mathbf{A}$

Metric	**O1**	**O4**	**O7**
Mean principal-angle drift of $\mathbf{A}$	3.50°	5.04°	12.59°

Table 14: Descriptions of datasets used from SNLI-derived SuperNI benchmark.

Dataset name	Task	Category	Metric
multi-woz-user-utterance-generation	Dialogue generation	SuperNI	Rouge-L
diplomacy-text-generation	Dialogue generation	SuperNI	Rouge-L
personachat-generate-next	Dialogue generation	SuperNI	Rouge-L
outcome-extraction	Information extraction	SuperNI	Rouge-L
glucose-reverse-cause-event-detection	Information extraction	SuperNI	Rouge-L
evaluation-relation-extraction	Information extraction	SuperNI	Rouge-L
quoref-answer-generation	Question answering	SuperNI	Rouge-L
commonsenseqa-answer-generation	Question answering	SuperNI	Rouge-L
sciq-answer-generation	Question answering	SuperNI	Rouge-L
reddit-tifu-long-text-summarization	Summarization	SuperNI	Rouge-L
xsum-summarization	Summarization	SuperNI	Rouge-L
samsum-summary	Summarization	SuperNI	Rouge-L
sst2-polarity-classification	Sentiment analysis	SuperNI	Accuracy
emotion-classification	Sentiment analysis	SuperNI	Accuracy

Table 15: Eight different task orders used in continual learning experiments.

Order	Task Sequence
1	dbpedia → ag → amazon → yahoo
2	dbpedia → amazon → ag → yahoo
3	ag → yahoo → dbpedia → amazon
4	amazon → qqp → mnli → imdb → sst-2 → wic → yahoo → boolq → ag → dbpedia → rte → multirc → record
5	mnli → imdb → rte → record → boolq → qqp → sst-2 → yahoo → multirc → ag → amazon → dbpedia → wic
6	multirc → wic → yahoo → qqp → rte → ag → dbpedia → amazon → mnli → record → imdb → boolq → sst-2
7 (SuperNI)	sciq → commonsenseqa → quoref → samsum → relation → reddit → xsum → diplomacy → multiwoz → emotion → personachat → outcome → glucose → sst2
8 (SuperNI)	quoref → samsum → multiwoz → xsum → outcome → diplomacy → commonsenseqa → relation → glucose → personachat → emotion → sciq → reddit → sst2

E Limitations

While SOAR provides an efficient and principled framework for continual LoRA consolidation, it has several limitations. First, the method operates at the level of individual rank directions, implicitly assuming that useful information can be localized within these components. In practice, however, task-relevant information may be distributed across multiple rank directions, forming shared subspaces rather than axis-aligned components. As a result, per-direction gating may not fully capture cross-directional dependencies, potentially leading to suboptimal retention of distributed knowledge.

Second, SOAR does not rely on replay or stored data, and therefore cannot recover information that has been fully overwritten during adaptation. Once a direction is reassigned to a new task and its prior contribution is lost, this information cannot be reconstructed in subsequent steps. Addressing these limitations, for example, through subspace-level consolidation or lightweight memory mechanisms, is an important direction for future work.

F Licenses for Existing Assets

All assets used in this work are properly credited with citations to their original papers. We list the primary assets and their licenses below.

Datasets and Evaluation dataset (1) DBpedia, AG News, Amazon Polarity, Yahoo Answers. [30] (2) **Large number of tasks:** It contains 4 standard CL tasks, along with GLUE tasks [23] (MNLI, QQP, SST-2, RTE) (CC BY-NC 4.0) and 4 SuperGLUE tasks [22] (WiC, ReCoRD, MultiRC, BoolQ) and IMDB movie reviews (CC BY-NC 4.0). Each of these tasks uses 2500 randomly sampled training samples and 500 test samples per class (3) **SuperNI Benchmark** [26] (Apache-2.0 license).

Models. Qwen3-8B [28] is released under Apache 2.0 (`https://huggingface.co/Qwen/Qwen3-4B`). LLaMA-3.2-3B [2] is released under the Llama 3.2 Community License (`https://huggingface.co/meta-llama/Llama-3.2-3B`). Llama-3.1-8B [2] is released under the Llama 3.1 Community License Agreement (`https://huggingface.co/meta-llama/Llama-3.1-8B-Instruct`)

Baselines. (1) Continual learning baselines: SLAO [14] (Since the original implementation is not publicly available, we implemented the method ourselves for our evaluation.), SeqLoRA; IncLoRA (are common and relatively simple baselines, so we implemented them ourselves for our experiments).

O-LoRA [25] (`https://github.com/cmnfriend/O-LoRA`) with MIT license; GainLoRA [10] (`https://github.com/liangyanshuo/gainlora`) with MIT license; CorDA [29] (`https://github.com/iboing/CorDA`) with Apache-2.0 license;

(2)LoRA merging baselines: LoRA-LEGO [32] (codes are not publicly available; hence it is coded by us); KnOTS [19] (`https://github.com/gstoica27/knots`) with MIT license.

All model and dataset versions correspond to those publicly available at the time of submission.

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: We have identified a problem of non-identifiably in LoRA based continual merging which leads to ill-posed merging as well as catastrophic forgetting. We have tried to theoretically explain via Lemma3.1 and via extensive experiments in Section 5.

   Guidelines:

   - The answer [N/A] means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A [No] or [N/A] answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: Limitations have briefly discussed in Conclusion and more detailes are provided in Appendix section tilted "Limitations".

   Guidelines:

   - The answer [N/A] means that the paper has no limitation while the answer [No] means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

Answer: [Yes]

Justification: We have provided proof of Lemma3.1 and justified other claims empirically also.

Guidelines:

- The answer [N/A] means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes]

   Justification: See section 5.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - If the paper includes experiments, a [No] answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
   - If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
   - Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
   - While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
     (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
     (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
     (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
     (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

Answer: [No]

Justification: We will subject to internal approval.

Guidelines:

- The answer [N/A] means that paper does not include experiments requiring code.
- Please see the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- While we encourage the release of code and data, we understand that this might not be possible, so [No] is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
- The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
- The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
- At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
- Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer) necessary to understand the results?

   Answer: [Yes]

   Justification: See Section 5 and Appendix.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [Yes]

   Justification: We have ran the experiments on multiple seeds and reported the mean and standard deviations for each method.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The authors should answer [Yes] if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
   - The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
   - The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)

- The assumptions made should be given (e.g., Normally distributed errors).
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g., negative error rates).
- If error bars are reported in tables or plots, the authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: All experiment were run on 1 H100 GPU of 80GB. Details are provided in Section 5.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [N/A]

   Justification: The paper does not involve human participants, all data used are open source popular datasets for Continual Learning tasks.

   Guidelines:

   - The answer [N/A] means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer [No], they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [N/A]

    Justification: This work does not deal with safety issues, discrimination, bias etc to the best of our knowledge.

    Guidelines:

    - The answer [N/A] means that there is no societal impact of the work performed.
    - If the authors answer [N/A] or [No], they should explain why their work has no societal impact or why the paper does not address societal impact.

- Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate Deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pre-trained language models, image generators, or scraped datasets)?

    Answer: [N/A]

    Justification: We use general models for our experiments and have cited all models and data appropriately.

    Guidelines:

    - The answer [N/A] means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
    - Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
    - We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [Yes]

    Justification: All models, datasets and codes for baselines used are taken from open source and cited.

    Guidelines:

    - The answer [N/A] means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.
    - The name of the license (e.g., CC-BY 4.0) should be included for each asset.
    - For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.

- If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
- For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [N/A]

    Justification: No new asset introduced in paper.

    Guidelines:

    - The answer [N/A] means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [N/A]

    Justification: The paper does not involve crowdsourcing nor research with human subjects.

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
    - According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [N/A]

    Justification: The paper does not involve crowdsourcing nor research with human subjects.

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
    - Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.

- We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
- For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

    Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does *not* impact the core methodology, scientific rigor, or originality of the research, declaration is not required.

    Answer: [N/A]

    Justification: The core method development in this research does not involve LLMs as any important, original, or non-standard components.

    Guidelines:

    - The answer [N/A] means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
    - Please refer to our LLM policy in the NeurIPS handbook for what should or should not be described.