NoTA: Normalized Tensor Adaptation for Parameter-Efficient Continual Learning

Yunsong Deng[1] **Yuning Qiu**[2] **Qibin Zhao**[2] **Guoxu Zhou**[1*]
[1]Guangdong University of Technology [2]RIKEN AIP

Abstract

Rehearsal-free class-incremental learning (CIL) requires a model to learn new classes sequentially without storing previous data, making the balance between plasticity and stability challenging. Adapter-based methods have become a prevalent solution by learning task-specific updates, but their effectiveness depends both on how each task update is parameterized and on how these updates accumulate across tasks. In this work, we propose Normalized Tensor Adaptation (NoTA), which addresses these two aspects jointly. NoTA represents task-specific updates using matrix product operator (MPO) adapters, providing a compact structured parameterization that supports expressive high-rank updates under a limited parameter budget. It further applies global normalization (GlobalNorm) to the accumulated update, calibrating its overall scale and reducing excessive cross-stage representation drift as tasks increase. We provide theoretical analysis showing that GlobalNorm yields a forgetting bound independent of the task sequence length, in contrast to task-wise normalization whose bound can grow with subsequent tasks. Building on the tensor structure of MPO, we further introduce H-NoTA, which uses task-level coefficients and inter-core modulation matrices to adaptively reweight historical updates at coarse and fine granularities. Extensive experiments and analyses demonstrate that our methods provide complementary benefits for plasticity and cumulative stability, achieving strong rehearsal-free CIL performance with high parameter efficiency.

1 Introduction

A fundamental goal of artificial intelligence is to develop systems that can learn continuously from a stream of data. Traditional deep neural networks, however, often suffer from catastrophic forgetting [1], where learning new tasks leads to the degradation of knowledge acquired from previous ones. Continual learning (CL) has therefore become an important research direction, with applications in robotics [2], on-device personal assistants [3], and autonomous systems [4], where data naturally arrive over time and retraining the entire model from scratch is impractical.

Among CL settings, class-incremental learning (CIL) is particularly challenging. In CIL, a model learns a sequence of tasks, each introducing a disjoint set of new classes, and is expected to recognize all classes seen so far with a single classifier. Existing approaches mitigate forgetting through regularization [5, 6], model expansion [7], or rehearsal with stored samples from previous tasks [8–10]. Although rehearsal is effective, storing historical data may be infeasible due to privacy constraints, memory limits, or deployment requirements on edge devices. This motivates rehearsal-free CIL, where the model must preserve old knowledge using only the data from the current task. The central difficulty is the stability-plasticity dilemma [11, 12], which means that the model should remain stable enough to retain previous classes while being sufficiently plastic to learn new ones.

*Corresponding author.

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

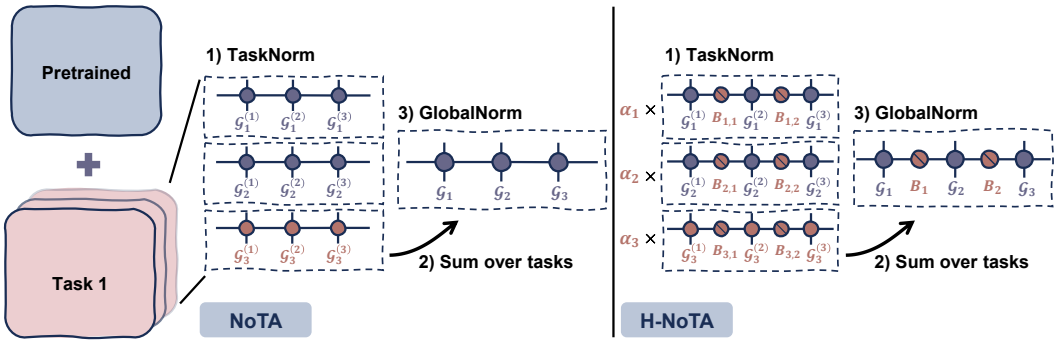


Figure 1: NoTA represents task-specific updates with MPO adapters and applies global normalization to the cumulative update. H-NoTA further introduces task-level coefficients α_t for coarse-grained control and inter-core modulation matrices $\mathbf{B}_{t,k}$ for fine-grained adjustment of historical feature directions. Red components denote the trainable parameters when learning the third task.

Recent work has explored parameter-efficient adaptation for rehearsal-free CIL, where a large pretrained backbone is frozen and only lightweight task-specific modules are learned. Adapter-based methods such as InfLoRA [13] and SD-LoRA [14], inspired by low-rank adaptation [15], achieve strong performance by learning and accumulating task-specific update matrices. However, this formulation introduces two coupled challenges. First, each task update should remain compact while retaining sufficient expressive capacity for learning new classes. Besides, as task-specific updates accumulate, their cumulative effect on the model should remain well calibrated to avoid excessive cross-stage drift and interference with historical knowledge. Existing rehearsal-free adapter methods provide limited investigation of these two aspects, particularly the structure of the task-update space and the behavior of accumulated updates over long task sequences.

In this work, we propose Normalized Tensor Adaptation (NoTA), a parameter-efficient framework that jointly addresses task-update representation and cumulative adaptation in rehearsal-free CIL. NoTA represents each task-specific update using a matrix product operator (MPO). Unlike matrix-based LoRA, which constrains the update through a low global matrix rank, MPO constrains cross-core tensor complexity and can reconstruct expressive high-rank updates under a compact parameter budget. This structured parameterization provides broader spectral support and more diverse task-update directions, improving plasticity while maintaining low storage cost. NoTA further applies global normalization (GlobalNorm) to the accumulated update carried by the model across tasks. GlobalNorm calibrates its overall scale and limits excessive cross-stage drift. Our empirical analysis shows that GlobalNorm reduces representation drift while consistently improving both matrix-based and MPO-based adapters, and our theoretical analysis establishes a forgetting bound governed by a fixed global budget and independent of the task sequence length, in contrast to task-wise normalization whose bound can grow with subsequent tasks. Taking advantage of the structure of MPO, we further introduce H-NoTA, which uses task-level coefficients and inter-core modulation matrices to adaptively refine historical task contributions at coarse and fine granularities. Extensive experiments across different task sequences, datasets, and pretrained backbones demonstrate the complementary benefits of tensorized task adaptation and cumulative normalization. [†] The main contributions of this work are summarized as follows:

- **Tensorized task adaptation.** We introduce MPO-based task updates that replace the low global matrix rank constraint with low cross-core tensor complexity, enabling compact yet expressive high-rank adaptation.
- **Cumulative update calibration.** We propose GlobalNorm to calibrate accumulated task updates and reduce cross-stage representation drift, with a forgetting bound independent of task sequence length.
- **Hierarchical adaptation.** We design H-NoTA with task-level coefficients and inter-core modulation for coarse-to-fine reweighting of historical updates.

[†]Our code is available here.

2 Related works

Class-incremental learning. Class-incremental learning (CIL) requires a model to learn a sequence of tasks with disjoint class sets and classify all seen classes using a single classifier. Existing CIL methods can be broadly grouped into rehearsal-based, architecture-based, and regularization-based approaches. Rehearsal-based methods store and replay a small subset of previous samples to preserve old knowledge [8, 9]. Architecture-based methods allocate additional components or subnetworks for new tasks [7, 16–18], while regularization-based methods constrain the update of parameters that are important for previous tasks [5, 6]. These methods provide different ways to balance stability and plasticity, but rehearsal is often constrained by privacy or memory budgets, and expanding the model may become costly as the task sequence grows. This has motivated recent rehearsal-free and parameter-efficient CIL methods, especially adapter-based approaches [13, 14] that freeze the pretrained backbone and accumulate task-specific updates. However, how task-specific updates accumulate across tasks and how this accumulation affects cross-stage model drift remain insufficiently understood. Our work studies this cumulative adaptation behavior and introduces explicit calibration of the accumulated update for rehearsal-free CIL.

Parameter-efficient adaptation. Parameter-efficient fine-tuning (PEFT) adapts large pretrained models by freezing the backbone and updating only lightweight modules, including adapters [19], prompts [20], and low-rank updates [15]. These ideas have been extended to rehearsal-free CIL through prompt-based methods [21, 22] and adapter-based methods [13, 14, 23, 24]. Prompt-based methods introduce learnable prompts to steer the frozen model, whereas adapter-based methods learn task-specific parameter updates to adapt the pretrained model. While effective, existing adapter-based methods primarily operate in flattened matrix spaces and provide limited analysis of how task-update structure and cross-task accumulation jointly affect continual adaptation. NoTA addresses both aspects through MPO-based tensorized updates and GlobalNorm within a unified framework.

Tensor networks. Tensor networks provide a powerful framework originating from quantum physics [25] for the efficient parameterization of high-order tensors. Representative forms include CANDECOMP/PARAFAC (CP) [26, 27], Tucker [28], Tensor Train (TT) [29], and Tensor Ring (TR) [30]. Their compact structured parameterization has been widely explored in machine learning for data completion [31], representation learning [32], and model compression [33, 34]. Matrix Product Operator (MPO) [35, 36] represents a matrix by tensorizing its input and output dimensions and decomposing the resulting high-order tensor into a sequence of interconnected cores. Unlike matrix low-rank adaptation, which directly constrains the global matrix rank, MPO constrains cross-core tensor complexity while permitting expressive high-rank reconstructed updates under a compact parameter budget. A recent CIL method [10] applies tensor decomposition to compress stored exemplar memory in a rehearsal-based setting. In contrast, NoTA is rehearsal-free and applies tensorization directly to task-specific model updates, using MPO to construct a compact and expressive continual adaptation space. Building on this structure, H-NoTA further introduces task-level coefficients and inter-core modulation matrices for hierarchical adjustment of historical updates.

3 Methods

In this section, the problem tackled in this work is outlined first. We then introduce the MPO-based tensor parameterization for task-specific updates. Based on this formulation, we examine the cumulative behavior of task-specific updates, which motivates explicit calibration of the accumulated adaptation through NoTA. Finally, we describe the proposed method in detail.

3.1 Preliminaries

Rehearsal-free CIL trains a model $\mathcal{M}$ with parameters θ on a sequence of tasks $\{\mathcal{T}_t\}_{t=1}^{T}$, where each task introduces a disjoint class set C_t. At task $\mathcal{T}_t$, the model minimizes $\mathcal{L}(\mathcal{D}_t, \theta)$ using only the current dataset $\mathcal{D}_t$. The goal is to obtain final parameters θ_T that recognize all seen classes $\bigcup_{t=1}^{T} \mathsf{C}_t$ without retaining previous task data.

The dominant approach to addressing this challenge is adapting the pretrained weight matrix $\mathbf{W}_0 \in \mathbb{R}^{d_{\text{out}} \times d_{\text{in}}}$ by learning a series of parameter-efficient updates $\Delta\mathbf{W}_t$ for each task $\mathcal{T}_t$. Recent matrix-

based CIL methods, such as SD-LoRA [14], represent the total weight update as a weighted linear combination of task-specific updates:

$$\mathbf{W}_T = \mathbf{W}_0 + \sum_{t=1}^{T} \alpha_t \Delta \mathbf{W}_t, \tag{1}$$

where $\alpha_t \in \mathbb{R}$ is a trainable scalar magnitude, and $\Delta \mathbf{W}_t \in \mathbb{R}^{d_{\text{out}} \times d_{\text{in}}}$ is the normalized incremental matrix from task $\mathcal{T}_t$. Additionally, $\Delta \mathbf{W}_t$ can be parameterized by low-rank matrices $\mathbf{A}_t \in \mathbb{R}^{d_{\text{out}} \times r}$ and $\mathbf{B}_t \in \mathbb{R}^{r \times d_{\text{in}}}$, with $r \ll \min(d_{\text{out}}, d_{\text{in}})$, such that $\Delta \mathbf{W}_t = \overline{\mathbf{A}_t \mathbf{B}_t}$ [13, 14]. When learning a new task $\mathcal{T}_T$, the set of trainable parameters includes the newly introduced incremental update $\Delta \mathbf{W}_T$ and the set of magnitudes $\{\alpha_k\}_{k=1}^{T}$, while past directions $\{\Delta \mathbf{W}_k\}_{k=1}^{T-1}$ are kept frozen.

While matrix-based parameterizations are widely used in parameter-efficient continual learning, they operate directly in the flattened weight space and could be suboptimal in terms of expressive efficiency under a limited parameter budget. To obtain a more structured and compact representation of task-specific updates, we further reparameterize the incremental update in a tensorized form. Specifically, we adopt an MPO-based formulation to model each incremental update, which not only improves parameterization efficiency but also provides a basis for the normalization strategy introduced later.

3.2 MPO-based tensor adaptation

Following the formulation given in Eq. (1), we reparameterize each task-specific incremental update in a tensorized form. For a pretrained weight matrix $\mathbf{W}_0 \in \mathbb{R}^{d_{\text{out}} \times d_{\text{in}}}$, we design a tensor adapter in the form of an MPO decomposition with L tensor cores such that $d_{\text{out}} = \prod_{k=1}^{L} d_{\text{out},k}$ and $d_{\text{in}} = \prod_{k=1}^{L} d_{\text{in},k}$. For task $\mathcal{T}_t$, the trainable tensor parameter is denoted by $\mathcal{G}_t = \{\mathcal{G}_t^{(k)}\}_{k=1}^{L}$, where each core satisfies $\mathcal{G}_t^{(k)} \in \mathbb{R}^{r_{k-1} \times d_{\text{out},k} \times d_{\text{in},k} \times r_k}$ with $r_0 = r_L = 1$. The task-specific incremental matrix is then reconstructed from these tensor cores through an MPO mapping.

Formally, we define the incremental update of task $\mathcal{T}_t$ as $\Delta \mathbf{W}_t := \overline{\text{MPO}}(\mathcal{G}_t)$, where $\overline{\text{MPO}}(\mathcal{G}_t)$ denotes the task-wise normalized MPO reconstruction of the tensor parameter $\mathcal{G}_t$. The reconstructed update is normalized before being incorporated into the model, while the precise normalization rule will be introduced in Sec. 3.4. Using this notation, the cumulative incremental update up to task t is written as $\mathbf{S}_t := \sum_{s=1}^{t} \Delta \mathbf{W}_s$, and the adapted weight becomes

$$\mathbf{W}_t = \mathbf{W}_0 + \mathbf{S}_t = \mathbf{W}_0 + \sum_{s=1}^{t} \overline{\text{MPO}}(\mathcal{G}_s). \tag{2}$$

Compared with matrix-based adaptation, Eq. (2) parameterizes each task-specific update through MPO cores and reconstructs it into the original weight space. It replaces the global low-rank constraint of matrix adaptation with a structured cross-core parameterization, providing a compact yet expressive update space under a limited parameter budget. Each task-specific update $\Delta \mathbf{W}_t$ is normalized before being added to the model. Nevertheless, this normalization is imposed only on the current update, rather than on their accumulation across tasks. This motivates the following empirical study of how the accumulated update evolves over the task sequence.

3.3 Empirical observation: magnitude of the incremental update

To examine how the accumulated update evolves during CIL, we track the cumulative magnitude of the adapter at task t by aggregating the Frobenius norms of all adapted modules. We define $\Phi_t = (\sum_{\ell \in \mathcal{M}} \|\mathbf{S}_t^{(\ell)}\|_F^2)^{1/2}$ for TaskNorm and $\Phi_t = (\sum_{\ell \in \mathcal{M}} \|\widehat{\mathbf{S}}_t^{(\ell)}\|_F^2)^{1/2}$ for GlobalNorm, corresponding to the cumulative updates applied to the model under each setting. We refer to independently normalizing each task-specific update as task-wise normalization (TaskNorm), whereas global normalization (GlobalNorm) acts directly on their cumulative update. Their precise formulations are introduced in Sec. 3.4. As shown in Fig. 2, TaskNorm leads to steady growth of the cumulative magnitude for both the matrix-based and MPO adapters, showing that locally normalizing each task-specific update does not calibrate the scale accumulated across tasks. In contrast, GlobalNorm keeps the cumulative magnitude close to a prescribed global scale throughout training.

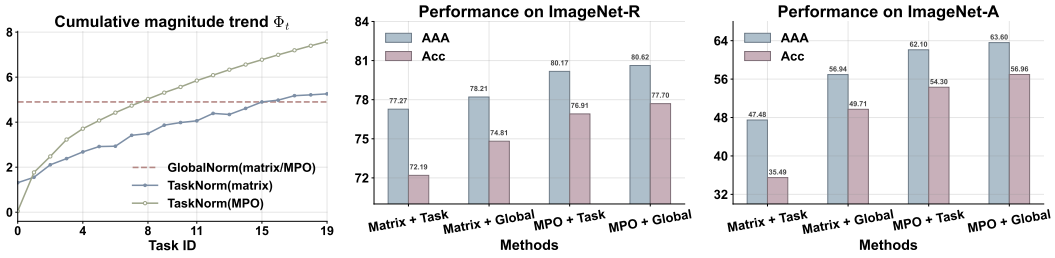


Figure 2: Results on ViT-Base over 20-task ImageNet-R and ImageNet-A sequences. The matrix-based adapter is the matrix low-rank counterpart of the MPO adapter under the same normalization framework. **Left:** cumulative magnitude trend Φ_t on ImageNet-A. **Middle:** Acc and AAA on ImageNet-R. **Right:** Acc and AAA on ImageNet-A.

The performance results on ImageNet-A and ImageNet-R show a consistent pattern. GlobalNorm improves Acc and AAA over TaskNorm, while the MPO adapter remains stronger than its matrix low-rank counterpart under the same normalization framework. These observations suggest that local normalization alone is insufficient to control the long-term accumulation of updates, and motivate us to introduce a normalization scheme that acts directly on the cumulative update.

3.4 Normalized tensor adaptation

The empirical results show that constraining each task-specific update does not calibrate the cumulative update carried by the model across tasks. This motivates us to normalize the adaptation at two levels within a unified framework. We first normalize each task-specific MPO update, and then normalize their cumulative sum. Based on this idea, we introduce NoTA and its variant H-NoTA with hierarchical coefficients.

Task-wise and global normalization We first define a unified normalization operator as:

$$\mathcal{N}_{\rho,\varepsilon}(\mathbf{X}) := \frac{\rho}{\max(\|\mathbf{X}\|_F, \varepsilon)}\mathbf{X}, \tag{3}$$

where $\rho > 0$ is the target magnitude and $\varepsilon > 0$ is a small constant for numerical stability. The task-wise normalized update is defined as

$$\Delta\mathbf{W}_t = \overline{\mathrm{MPO}}(\mathcal{G}_t) = \mathcal{N}_{1,\varepsilon}\big(\mathrm{MPO}(\mathcal{G}_t)\big). \tag{4}$$

To further control the module-wise cumulative update across tasks, we apply the same normalization form directly to the cumulative update $\mathbf{S}_t := \sum_{s=1}^{t} \Delta\mathbf{W}_s$, and define

$$\widehat{\mathbf{S}}_t := \mathcal{N}_{\tau,\varepsilon}(\mathbf{S}_t), \qquad \mathbf{W}_t^{\mathrm{g}} = \mathbf{W}_0 + \widehat{\mathbf{S}}_t. \tag{5}$$

The design maintains task-wise control over each incremental update while providing explicit control over the cumulative adaptation scale across tasks.

H-NoTA Matrix-based formulations such as Eq. (1) use task-level coefficients to reweight historical updates, which provide only coarse task-level control. In the MPO form, each update is represented by a sequence of tensor cores linked through bond dimensions, which makes it possible to introduce finer control inside the update itself. Based on this observation, we extend NoTA with two levels of lightweight coefficients: a task coefficient $\alpha_t \in \mathbb{R}$ for the overall contribution of task t, and diagonal bond modulation matrices $\mathbf{B}_{t,k} = \mathrm{diag}(\boldsymbol{\beta}_{t,k}) \in \mathbb{R}^{r_k \times r_k}$ with $\boldsymbol{\beta}_{t,k} \in \mathbb{R}^{r_k}$ for $k = 1, \ldots, L-1$. Here α_t provides coarse task-level reweighting, while $\mathbf{B}_{t,k}$ modulates the latent channels between adjacent MPO cores.

Concretely, for the k-th core $\mathcal{G}_t^{(k)}$, let $i_k \in \{1, \ldots, d_{\mathrm{out},k}\}$ and $j_k \in \{1, \ldots, d_{\mathrm{in},k}\}$ denote the output and input indices, and let $\mathbf{G}_{t,k}^{i_k,j_k} \in \mathbb{R}^{r_{k-1} \times r_k}$ be the corresponding matrix slice. The modulated MPO reconstruction is written as [3]

$$\mathrm{MPO}\big(\mathcal{G}_t, \{\mathbf{B}_{t,k}\}\big)_{i_1 \ldots i_L,\, j_1 \ldots j_L} = \mathbf{G}_{t,1}^{i_1,j_1}\mathbf{B}_{t,1}\mathbf{G}_{t,2}^{i_2,j_2}\cdots\mathbf{B}_{t,L-1}\mathbf{G}_{t,L}^{i_L,j_L}. \tag{6}$$

[3] An intuitive illustration of the contraction is provided in Appendix B.1.

Using the same normalization framework, H-NoTA defines

$$\mathbf{W}_t^{\mathrm{g}} = \mathbf{W}_0 + \mathcal{N}_{\tau,\varepsilon}\left(\sum_{s=1}^{t} \alpha_s \, \overline{\mathrm{MPO}}\big(\mathcal{G}_s, \{\mathbf{B}_{s,k}\}\big)\right), \tag{7}$$

where

$$\overline{\mathrm{MPO}}\big(\mathcal{G}_s, \{\mathbf{B}_{s,k}\}\big) := \mathcal{N}_{1,\varepsilon}\big(\mathrm{MPO}\big(\mathcal{G}_s, \{\mathbf{B}_{s,k}\}\big)\big). \tag{8}$$

When a new task $\mathcal{T}_t$ arrives, we initialize $\mathcal{G}_t$, α_t, and $\{\mathbf{B}_{t,k}\}_{k=1}^{L-1}$. The structural parameters of previous tasks, namely $\{\mathcal{G}_s\}_{s=1}^{t-1}$, remain frozen, while all coefficients $\{\alpha_s\}_{s=1}^{t}$ and $\{\mathbf{B}_{s,k}\}_{s=1,k=1}^{t,\,L-1}$ remain trainable. This keeps previously learned structural directions fixed, while allowing their contributions to be adjusted at both the task-level and the bond-level. The hierarchical coefficients determine the relative composition of historical updates, while GlobalNorm independently calibrates their overall scale.

4 Analysis

In this section, we analyze the complementary roles of MPO-based tensor adaptation and GlobalNorm in rehearsal-free CIL. We first examine the structural expressivity and task-update diversity of MPO, then investigate how GlobalNorm calibrates cross-stage representation drift. Finally, we establish forgetting bounds under task-wise and global normalization.

4.1 MPO expressivity and task-update diversity

Matrix-based LoRA constrains each task update $\Delta\mathbf{W}_t = \mathbf{A}_t\mathbf{B}_t$ to a global matrix rank of at most r_m, whereas MPO constrains the bond ranks between tensor cores. In particular, when all internal bond ranks equal one, the reconstructed matrix satisfies

$$\mathrm{rank}\big(\mathrm{MPO}(\mathcal{G}_t)\big) = \prod_{k=1}^{L} \mathrm{rank}\big(\mathcal{G}_t^{(k)}(1,:,:,1)\big). \tag{9}$$

This illustrates that low MPO bond ranks do not impose the same global matrix-rank constraint as matrix LoRA, allowing compact yet expressive high-rank updates. The compositional structure also allows different tasks to form distinct update directions, which we examine empirically.

Empirical validation. We analyze the learned task-specific updates on ImageNet-A ($T = 10$), as shown in Tab. 1. Numerical rank measures the rank of the reconstructed update, while stable rank $\mathrm{sr}(\Delta\mathbf{W}_t) = \|\Delta\mathbf{W}_t\|_F^2/\|\Delta\mathbf{W}_t\|_2^2$, effective rank $\exp(-\sum_i p_i \log p_i)$, and top-1 energy p_1, where $p_i = \sigma_i^2/\|\Delta\mathbf{W}_t\|_F^2$ and σ_i denotes the i-th largest singular value of $\Delta\mathbf{W}_t$, characterize spectral distribution and concentration. We measure task-update similarity using mean absolute cosine similarity between vectorized updates and historical interference using $I_{q\leftarrow s} = \mathrm{tr}(\overline{\Delta\mathbf{W}}_s\mathbf{\Sigma}_q\overline{\Delta\mathbf{W}}_s^{\top})$ for $s > q$, where $\overline{\Delta\mathbf{W}}_s = \Delta\mathbf{W}_s/\|\Delta\mathbf{W}_s\|_F$ and $\mathbf{\Sigma}_q = \mathbb{E}[\mathbf{h}_q\mathbf{h}_q^{\top}]$ is the historical activation second moment. Compared with Matrix $r = 10$, MPO $r = 4$ reconstructs nearly full-rank updates with broader spectral support, reducing mean task-update similarity by 76.8% and historical interference by 10.1%. Similar trends hold against the setting of Matrix $r = 2$. These results support the structured expressivity and task-update diversity of MPO, together with lower average interference on historical tasks.

Table 1: Spectral properties and cross-task interference on ImageNet-A ($T = 10$) under GlobalNorm. Results are averaged over 24 modules and the relevant tasks or task pairs. Cosine and interference are scaled by 10^{-3}.

Adapter	Num. rank ↑	Stable rank ↑	Eff. rank ↑	Top-1 (%) ↓	Abs. cosine ↓	Interference ↓
Matrix ($r = 2$)	2	1.46	1.81	69.91	3.81	1.99
Matrix ($r = 10$)	10	2.58	5.97	41.22	4.29	1.68
MPO ($r = 4$)	**767–768**	**72.70**	**326.32**	**1.41**	**0.99**	**1.51**

4.2 GlobalNorm calibration and representation drift

The cumulative magnitude Φ_t characterizes the overall adaptation scale but does not capture cross-stage changes in historical representations. For an adapted module, let $\mathbf{D}_{q,t} = \widehat{\mathbf{S}}_t - \widehat{\mathbf{S}}_q$ under GlobalNorm and $\mathbf{D}_{q,t} = \mathbf{S}_t - \mathbf{S}_q$ under TaskNorm. We decompose $\mathbf{D}_{q,t} = \rho_{q,t}\mathbf{V}_{q,t}$, where $\rho_{q,t} = \|\mathbf{D}_{q,t}\|_F$ and $\|\mathbf{V}_{q,t}\|_F = 1$ for $\rho_{q,t} > 0$. Given fixed historical input activations $\mathbf{h}_q$ with second moment $\mathbf{\Sigma}_q$, the module-level representation drift satisfies

$$\mathcal{D}_{q,t} = \mathbb{E}\left[\|\mathbf{D}_{q,t}\mathbf{h}_q\|_2^2\right] = \underbrace{\rho_{q,t}^2}_{\text{magnitude}} \underbrace{\operatorname{tr}\left(\mathbf{V}_{q,t}\mathbf{\Sigma}_q\mathbf{V}_{q,t}^\top\right)}_{A_{q,t}\text{: alignment}}. \tag{10}$$

GlobalNorm bounds the cross-stage magnitude by $\rho_{q,t} \le 2\tau$, while the activation alignment $A_{q,t}$ depends on update directions and historical activations.

Empirical validation. Table 2 compares these factors on ImageNet-A ($T = 10$). For MPO, GlobalNorm reduces both squared cross-stage magnitude $\overline{\rho^2}$ and representation drift $\overline{\mathcal{D}}$ by approximately 31%, with nearly unchanged activation alignment $\overline{A}$ and a corresponding Acc improvement of 1.64 points. At comparable cross-stage magnitude, MPO under GlobalNorm also exhibits lower alignment and representation drift than Matrix $r = 10$. These results support the complementary effects of MPO-based adaptation and GlobalNorm-based scale calibration.

Table 2: Cross-stage representation drift on ImageNet-A ($T = 10$), averaged over 45 stage pairs and 24 modules. TN/GN denote TaskNorm/GlobalNorm.

Adapter	$\overline{\rho^2}$	$\overline{A}$	$\overline{\mathcal{D}}$	Acc
Matrix $r = 10$ (GN)	2.46	1.44	3.63	48.37
MPO $r = 4$ (TN)	3.67	1.25	4.57	58.74
MPO $r = 4$ (GN)	2.53	1.25	3.16	**60.38**

4.3 Forgetting bounds

We further establish forgetting bounds under task-wise and global normalization. Let $\mathcal{M}$ denote the set of adapted modules, and let $\|\cdot\|_F$ denote the aggregate Frobenius norm over these modules.

For an old task indexed by q and a later stage $t > q$, define the forgetting under task-wise normalization and global normalization as

$$\Delta\mathcal{L}_q^{\text{task}}(q \to t) := \mathcal{L}_q(\mathbf{W}_t) - \mathcal{L}_q(\mathbf{W}_q), \qquad \Delta\mathcal{L}_q^{\text{global}}(q \to t) := \mathcal{L}_q(\mathbf{W}_t^{\text{g}}) - \mathcal{L}_q(\mathbf{W}_q^{\text{g}}), \tag{11}$$

where $\mathcal{L}_q(\mathbf{W})$ denotes the loss of task q evaluated as a function of all adapted modules, $\mathbf{W}_t$ is the model obtained under task-wise normalization, and $\mathbf{W}_t^{\text{g}}$ is the model obtained under global normalization.

Theorem 1 (Forgetting bounds)**.** *Assume that, for each old task q, the loss $\mathcal{L}_q(\mathbf{W})$ is locally L_q-smooth with respect to the adapted modules. Then, for any $t > q$,*

$$\Delta\mathcal{L}_q^{\text{task}}(q \to t) \le (t-q)\sqrt{|\mathcal{M}|}\left\|\nabla\mathcal{L}_q(\mathbf{W}_q)\right\|_F + \frac{L_q}{2}|\mathcal{M}|(t-q)^2, \tag{12}$$

$$\Delta\mathcal{L}_q^{\text{global}}(q \to t) \le 2\tau\sqrt{|\mathcal{M}|}\left\|\nabla\mathcal{L}_q(\mathbf{W}_q^{\text{g}})\right\|_F + 2L_q|\mathcal{M}|\tau^2. \tag{13}$$

Moreover, since H-NoTA uses the same module-wise global normalization, the bound in Eq. (13) *also applies to H-NoTA. If $\mathbf{W}_q$ and $\mathbf{W}_q^{\text{g}}$ are stationary points of $\mathcal{L}_q$ with respect to the adapted module weights, the bounds reduce to*

$$\Delta\mathcal{L}_q^{\text{task}}(q \to t) \le \frac{L_q}{2}|\mathcal{M}|(t-q)^2, \qquad \Delta\mathcal{L}_q^{\text{global}}(q \to t) \le 2L_q|\mathcal{M}|\tau^2. \tag{14}$$

Theorem 1 shows that task-wise normalization leads to a forgetting bound that grows with the number of subsequent tasks, while global normalization yields a bound controlled by the global budget τ and the number of adapted modules, independent of the task sequence length. The required local smoothness region is also bounded for GlobalNorm, since the compared parameter states remain within distance $2\tau\sqrt{|\mathcal{M}|}$, whereas the TaskNorm distance can grow with $t - q$. Note that even if the adapted module weights do not reach an exact stationary point, the general bounds in Eqs. (12) and (13) remain valid. Detailed proofs are deferred to Appendix A.2.

Table 3: Performance and parameter comparison of different methods on ImageNet-A (IN-A) and ImageNet-R (IN-R) with the pretrained ViT-B/16 backbone. We report the final accuracy (Acc), average anytime accuracy (AAA), and the number of additional trainable parameters per task. Values in parentheses denote the standard deviation (SD) across multiple runs. The best result for each metric is highlighted in bold and the second best is underlined.

Method	#Para.	IN-A, $T=10$		IN-R, $T=5$		IN-R, $T=10$		IN-R, $T=20$	
		Acc	AAA	Acc	AAA	Acc	AAA	Acc	AAA
Full Fine-Tuning	86M	$16.31_{(7.89)}$	$30.04_{(13.2)}$	$64.92_{(0.87)}$	$75.57_{(0.50)}$	$60.57_{(1.06)}$	$72.31_{(1.09)}$	$49.95_{(1.31)}$	$65.32_{(0.84)}$
L2P [40]	480K	$51.37_{(0.84)}$	$57.14_{(0.66)}$	$75.62_{(0.45)}$	$78.02_{(0.44)}$	$73.85_{(0.68)}$	$76.63_{(0.68)}$	$71.86_{(0.48)}$	$74.71_{(0.70)}$
DualPrompt [41]	60K	$55.92_{(0.56)}$	$61.73_{(1.02)}$	$74.53_{(0.33)}$	$77.20_{(0.35)}$	$73.17_{(0.35)}$	$76.26_{(0.47)}$	$71.64_{(0.47)}$	$74.58_{(0.88)}$
CODA-Prompt [21]	380K	$46.64_{(0.51)}$	$55.74_{(0.49)}$	$75.82_{(0.26)}$	$78.81_{(0.16)}$	$74.30_{(0.36)}$	$77.78_{(0.17)}$	$72.28_{(0.57)}$	$75.33_{(0.38)}$
HiDe-Prompt [22]	80K	$38.14_{(0.68)}$	$52.65_{(0.26)}$	$68.16_{(0.23)}$	$73.94_{(0.29)}$	$64.82_{(0.28)}$	$71.91_{(0.14)}$	$60.07_{(0.47)}$	$67.84_{(0.48)}$
InfLoRA [13]	370K	$49.68_{(0.89)}$	$63.15_{(0.52)}$	$78.04_{(0.26)}$	$\underline{82.57}_{(0.28)}$	$75.22_{(0.41)}$	$80.64_{(0.24)}$	$71.21_{(0.75)}$	$77.13_{(0.55)}$
SD-LoRA [14]	369K	$55.77_{(3.46)}$	$64.27_{(0.48)}$	$79.02_{(0.48)}$	$81.81_{(0.39)}$	$\underline{78.04}_{(0.46)}$	$\underline{81.10}_{(0.41)}$	$\underline{76.30}_{(1.05)}$	$\underline{79.06}_{(1.34)}$
NoTA (r=4)	68K	$\mathbf{59.48}_{(0.96)}$	$\underline{64.98}_{(0.53)}$	$\underline{79.21}_{(0.15)}$	$81.87_{(0.25)}$	$77.19_{(0.51)}$	$80.15_{(0.58)}$	$72.83_{(0.29)}$	$76.30_{(0.41)}$
H-NoTA (r=4)	68K	$\underline{59.33}_{(1.47)}$	$\mathbf{65.47}_{(1.12)}$	$\mathbf{80.32}_{(0.38)}$	$\mathbf{82.87}_{(0.48)}$	$\mathbf{79.10}_{(0.25)}$	$\mathbf{81.88}_{(0.39)}$	$\mathbf{76.88}_{(0.51)}$	$\mathbf{80.00}_{(0.40)}$

5 Experiments

5.1 Experimental setups

Datasets. We conduct experiments on three widely used datasets for rehearsal-free CIL: ImageNet-R [37], ImageNet-A [38], and DomainNet [39]. Following prior class incremental protocols [13, 14], we split the classes of each dataset evenly into disjoint tasks and train the model sequentially without storing or replaying previous task data. For ImageNet-R and ImageNet-A, we evaluate 5-task, 10-task, and 20-task sequences. For DomainNet, we use a 5-task sequence to evaluate performance under a larger and more diverse data distribution.

Compared methods and evaluation metrics. We compare our methods with full fine-tuning and representative rehearsal-free CIL methods, including prompt-based methods L2P [40], DualPrompt [41], CODA-Prompt [21], and HiDe-Prompt [22], as well as parameter-efficient adaptation methods InfLoRA [13] and SD-LoRA [14]. Our proposed NoTA and H-NoTA follow the definitions in Eqs. (5) and (7). The MPO rank is specified in the corresponding experimental tables. Let $a_{t,i}$ denote the accuracy on task i after finishing task t. We report the final average accuracy (Acc) and average anytime accuracy (AAA), which are defined as $\mathrm{Acc} = \frac{1}{T}\sum_{i=1}^{T} a_{T,i}$ and $\mathrm{AAA} = \frac{2}{T(T+1)}\sum_{t=1}^{T}\sum_{i=1}^{t} a_{t,i}$.

Implementation details. We follow the basic experimental settings of recent parameter-efficient CIL works [42, 14]. For the main experiments, we use ViT-B/16 [43] pretrained on ImageNet-21K and fine-tuned on ImageNet-1K as the foundation model. To evaluate the robustness to different pretrained backbones, we additionally conduct experiments with a pretrained ViT-B/16 backbone from DINO [44]. The proposed MPO adapters are inserted into the query and value projections of the self-attention layers, and each MPO chain contains $L = 3$ tensor cores. The main results are averaged over five independent runs with different random seeds. Additional implementation details are provided in Appendix C.1.

5.2 Main results

Comparison on various datasets and backbones. A comprehensive evaluation of the proposed NoTA and H-NoTA is conducted across different datasets and pretrained backbones, with detailed results reported in Tabs. 3 and 4. As shown in Tab. 3, H-NoTA achieves the best performance in terms of both Acc and AAA

Table 4: Generalization results on DomainNet and ViT-B/16 from DINO.

Dataset	Method	Backbone	Tasks	Rank	#Para.	Acc	AAA
DomainNet	SD-LoRA	ViT-B/16	$T=5$	10	369K	$68.22_{(0.25)}$	$72.27_{(0.50)}$
	H-NoTA	ViT-B/16	$T=5$	8	246K	$\mathbf{68.69}_{(0.13)}$	$\mathbf{72.60}_{(0.26)}$
IN-A	SD-LoRA	ViT-DINO	$T=10$	10	369K	$22.50_{(4.09)}$	$32.31_{(1.33)}$
	H-NoTA	ViT-DINO	$T=10$	10	376K	$\mathbf{30.19}_{(1.26)}$	$\mathbf{36.45}_{(1.25)}$
	H-NoTA	ViT-DINO	$T=10$	4	68K	$28.33_{(1.07)}$	$35.07_{(1.10)}$

Table 5: Ablation of MPO structure and hierarchical coefficients on ImageNet-A (IN-A) with the pretrained ViT-B/16 backbone. We compare SD-LoRA (matrix adapter), NoTA, and H-NoTA across 5, 10, and 20 task sequences. NoTA and H-NoTA are evaluated under the same MPO rank to isolate the effect of hierarchical coefficients.

Method	Rank	#Para.	IN-A, $T = 5$		IN-A, $T = 10$		IN-A, $T = 20$	
			Acc	AAA	Acc	AAA	Acc	AAA
SD-LoRA	10	369K	$59.93_{(0.87)}$	$65.47_{(0.91)}$	$55.77_{(3.46)}$	$64.27_{(0.48)}$	$51.13_{(0.91)}$	$58.89_{(1.32)}$
NoTA	4	68K	$59.76_{(0.88)}$	$65.02_{(0.96)}$	$\mathbf{59.48}_{(0.96)}$	$64.98_{(0.53)}$	$56.66_{(1.35)}$	$62.90_{(1.12)}$
H-NoTA	4	68K	$\mathbf{60.12}_{(0.67)}$	$\mathbf{65.68}_{(0.71)}$	$59.33_{(1.47)}$	$\mathbf{65.47}_{(1.12)}$	$\mathbf{56.68}_{(0.99)}$	$\mathbf{63.07}_{(0.81)}$

across most ImageNet-A (IN-A) and ImageNet-R (IN-R) settings. With only 68K parameters per task at rank 4, H-NoTA is substantially more parameter-efficient than the strong adapter-based baseline SD-LoRA (369K). The DomainNet and DINO results further demonstrate its effectiveness across different data distributions and pretrained backbones. Additional comparisons with CL-LoRA [23] and LoRA-Sub [24] are provided in Appendix C.2, where H-NoTA achieves competitive performance with higher efficiency.

5.3 Ablation studies

TaskNorm versus GlobalNorm. The ablation of the proposed normalization strategy is provided in Fig. 2, with discussion in Section 3.3. The results show that TaskNorm cannot prevent the cumulative update from growing over the task sequence, whereas GlobalNorm directly constrains the cumulative adapter and improves both Acc and AAA for matrix-based and MPO-based adapters. This demonstrates that global normalization provides a consistent benefit beyond local task-wise control. We then analyze how the MPO structure and hierarchical coefficients further affect performance. Detailed quantitative analysis is provided in Appendix C.3.

Effect of MPO structure and hierarchical coefficients. Tab. 5 studies the effect of the MPO structure and the hierarchical coefficients on IN-A. The dominant gain comes from the proposed normalized tensor adaptation. Compared with SD-LoRA (matrix adapter), NoTA achieves stronger performance in longer task sequences, demonstrating the benefit of using an MPO-based update space together with global normalization. On top of this strong baseline, H-NoTA further improves NoTA under the same MPO rank in most settings. The results show that the hierarchical coefficients provide useful additional flexibility for reweighting task directions and modulating the MPO reconstruction, while the main stability benefit is already provided by NoTA.

5.4 Performance analysis

Efficiency analysis. We compare the storage and computational efficiency of H-NoTA with representative prompt-based and adapter-based methods on IN-R in Tab. 6 and Fig. 3. H-NoTA significantly reduces storage cost while maintaining strong performance, at the price of only marginal computational overhead. In terms of GFLOPs, H-NoTA incurs a slight increase over SD-LoRA, with about $1.2\times$ online inference GFLOPs and $1.3\times$ training GFLOPs relative to SD-LoRA, but it remains substantially more efficient than the prompt-based baseline CODA-Prompt. In terms of storage, although the amount of additional parameters grows linearly with the number of tasks for both H-NoTA and SD-LoRA, the growth rate of H-NoTA is smaller. In this setting, H-NoTA with rank 4 achieves better AAA than SD-LoRA with rank 10 (81.88% vs. 81.10%) while reducing the per-task stored parameters from 369K to

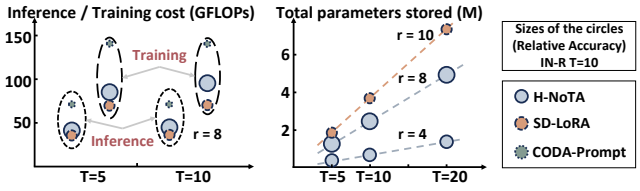
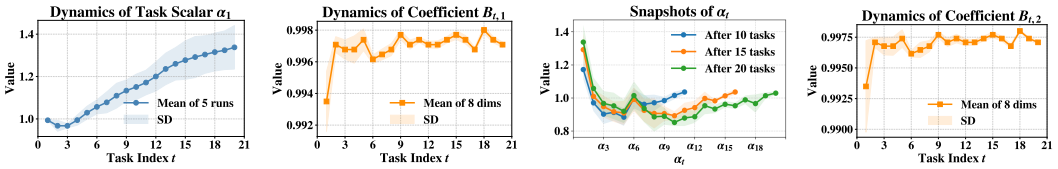


Figure 3: Efficiency comparison on ImageNet-R. **Left:** inference GFLOPs with online computation of the cumulative update and training GFLOPs. **Right:** stored additional parameters as the task sequence grows.

Table 6: Efficiency comparison with representative prompt-based and adapter-based methods on IN-R. We report the number of additional parameters to be stored after learning a sequence of tasks, together with training and online inference GFLOPs under different numbers of tasks.

Method	$T = 5$			$T = 10$		
	Stored Param.	Train GFLOPs	Infer. GFLOPs	Stored Param.	Train GFLOPs	Infer. GFLOPs
CODA-Prompt [21]	3.84M	140.11	70.29	3.84M	140.11	70.29
SD-LoRA [14]	1.85M	68.36	34.59	3.69M	69.51	35.19
H-NoTA (r=8)	1.23M	88.67	41.90	2.46M	94.13	44.52
H-NoTA (r=4)	0.34M	85.00	40.55	0.68M	86.88	41.83



(a) (b) (c) (d)

Figure 4: Visualization of hierarchical coefficient dynamics in H-NoTA on ImageNet-R with $T = 20$ tasks. **(a)** Evolution of the first task-level coefficient α_1 as the task index increases. **(b)** Evolution of the first inter-core modulation matrix $\mathbf{B}_{t,1}$, summarized by the mean and standard deviation of its diagonal entries $\boldsymbol{\beta}_{t,1}$ across bond dimensions. **(c)** Snapshots of the task-level coefficient vector $\{\alpha_i\}_{i=1}^{T}$ after training on 10, 15, and 20 tasks. **(d)** Evolution of the second inter-core modulation matrix $\mathbf{B}_{t,2}$, summarized in the same way as in **(b)**.

68K, corresponding to an approximately $5.4\times$ compression ratio. These results show that H-NoTA provides a favorable storage-performance trade-off for rehearsal-free CIL.

Behavior of hierarchical coefficients. To understand how H-NoTA takes advantage of the expanded adaptation space in Eq. (7), we visualize the learned hierarchical coefficients in Fig. 4. For the inter-core modulation matrices $\mathbf{B}_{t,k} = \mathrm{diag}(\boldsymbol{\beta}_{t,k})$, Figs. 4b and 4d show that the diagonal coefficients remain close to 1 with small variations across bond dimensions throughout the task sequence. This suggests that H-NoTA maintains mild inter-core modulation while largely preserving the structural directions of the MPO reconstruction. The task-level coefficients α_t show a complementary pattern. Fig. 4a shows that α_1 increases as training proceeds, while Fig. 4c reveals a task-dependent pattern in which early task coefficients remain large, intermediate ones are suppressed, and recent ones are mildly elevated. We speculate that this reflects a temporary boost for learning, which is subsequently suppressed as these tasks age and become intermediate. Overall, these dynamics suggest that H-NoTA uses $\mathbf{B}_{t,k}$ to regulate feature-level plasticity and α_t to reweight task-level contributions, providing a more flexible way to balance stability and plasticity in rehearsal-free CIL.

6 Conclusion

In this paper, we proposed NoTA, a normalized tensor adaptation framework for rehearsal-free class-incremental learning. The central idea is to represent task-specific updates with MPO-based tensor adapters and to apply global normalization to the cumulative update carried by the model across tasks. It constructs expressive task-specific updates and calibrates their accumulation across tasks. We further introduced H-NoTA, which augments NoTA with task-level coefficients and inter-core modulation matrices to provide more flexible control over historical task contributions. Our analysis demonstrates the structured high-rank expressivity of MPO and establishes a forgetting bound independent of the task sequence length. Extensive experiments show that NoTA and H-NoTA achieve strong CIL performance with high parameter efficiency, and support the complementary roles of MPO and GlobalNorm. While this work focuses on visual classification benchmarks, the formulation is not tied to image models. Its extension to language, multimodal learning, and other more complex continual learning scenarios is a promising direction for future work.

Acknowledgments and Disclosure of Funding

This work was supported in part by the National Natural Science Foundation of China under Grant 62606102; in part by the Guangdong Natural Science Foundation under Grant 2024A1515010114; in part by JSPS Bilateral Program under Grant JPJSBP120257420; in part by JSPS KAKENHI under Grant JP24K20849 and JP23K28109.

References

[1] Xilai Li, Yingbo Zhou, Tianfu Wu, Richard Socher, and Caiming Xiong. Learn to grow: A continual structure learning framework for overcoming catastrophic forgetting. In *International Conference on Machine Learning*, volume 97, pages 3925–3934, 2019.

[2] Maciej Wołczyk, Michał Zając, Razvan Pascanu, Łukasz Kuciński, and Piotr Miłoś. Continual world: A robotic benchmark for continual reinforcement learning. In *Advances in Neural Information Processing Systems*, volume 34, pages 28496–28510, 2021.

[3] Zifeng Wang, Zheng Zhan, Yifan Gong, Geng Yuan, Wei Niu, Tong Jian, Bin Ren, Stratis Ioannidis, Yanzhi Wang, and Jennifer Dy. SparCL: Sparse continual learning on the edge. In *Advances in Neural Information Processing Systems*, volume 35, pages 20366–20380, 2022.

[4] Huiping Zhuang, Di Fang, Kai Tong, Yuchen Liu, Ziqian Zeng, Xu Zhou, and Cen Chen. Online analytic exemplar-free continual learning with large models for imbalanced autonomous driving task. *IEEE Transactions on Vehicular Technology*, 74(2):1949–1958, 2025.

[5] James Kirkpatrick, Razvan Pascanu, Neil Rabinowitz, Joel Veness, Guillaume Desjardins, Andrei A Rusu, Kieran Milan, John Quan, Tiago Ramalho, Agnieszka Grabska-Barwinska, et al. Overcoming catastrophic forgetting in neural networks. *Proceedings of the National Academy of Sciences*, 114(13):3521–3526, 2017.

[6] Zhizhong Li and Derek Hoiem. Learning without forgetting. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 40(12):2935–2947, 2017.

[7] Andrei A Rusu, Neil C Rabinowitz, Guillaume Desjardins, Hubert Soyer, James Kirkpatrick, Koray Kavukcuoglu, Razvan Pascanu, and Raia Hadsell. Progressive neural networks. *arXiv preprint arXiv:1606.04671*, 2016.

[8] Sylvestre-Alvise Rebuffi, Alexander Kolesnikov, Georg Sperl, and Christoph H. Lampert. iCaRL: Incremental classifier and representation learning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, July 2017.

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

[10] Yuhang Li, Guoxu Zhou, Zhenhao Huang, Xinqi Chen, Yuning Qiu, and Qibin Zhao. Tensor decomposition based memory-efficient incremental learning. In *International Conference on Machine Learning*, volume 267, pages 36906–36920, 2025.

[11] Shibhansh Dohare, J Fernando Hernandez-Garcia, Qingfeng Lan, Parash Rahman, A Rupam Mahmood, and Richard S Sutton. Loss of plasticity in deep continual learning. *Nature*, 632(8026):768–774, 2024.

[12] Dongwan Kim and Bohyung Han. On the stability-plasticity dilemma of class-incremental learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 20196–20204, June 2023.

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

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

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

[16] Rahul Ramesh and Pratik Chaudhari. Model zoo: A growing brain that learns continually. In *International Conference on Learning Representations*, 2022.

[17] Marcin Pietron, Kamil Faber, Dominik Żurek, and Roberto Corizzo. TinySubNets: An efficient and low capacity continual learning strategy. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 39, pages 19913–19920, 2025.

[18] Arthur Douillard, Alexandre Ramé, Guillaume Couairon, and Matthieu Cord. DyTox: Transformers for continual learning with dynamic token expansion. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9285–9295, June 2022.

[19] Neil Houlsby, Andrei Giurgiu, Stanislaw Jastrzebski, Bruna Morrone, Quentin De Laroussilhe, Andrea Gesmundo, Mona Attariyan, and Sylvain Gelly. Parameter-efficient transfer learning for NLP. In *International Conference on Machine Learning*, volume 97, pages 2790–2799, 2019.

[20] Brian Lester, Rami Al-Rfou, and Noah Constant. The power of scale for parameter-efficient prompt tuning. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, 2021.

[21] James Seale Smith, Leonid Karlinsky, Vyshnavi Gutta, Paola Cascante-Bonilla, Donghyun Kim, Assaf Arbelle, Rameswar Panda, Rogerio Feris, and Zsolt Kira. CODA-Prompt: Continual decomposed attention-based prompting for rehearsal-free continual learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 11909–11919, June 2023.

[22] Liyuan Wang, Jingyi Xie, Xingxing Zhang, Mingyi Huang, Hang Su, and Jun Zhu. Hierarchical decomposition of prompt-based continual learning: Rethinking obscured sub-optimality. In *Advances in Neural Information Processing Systems*, volume 36, pages 69054–69076, 2023.

[23] Jiangpeng He, Zhihao Duan, and Fengqing Zhu. CL-LoRA: Continual low-rank adaptation for rehearsal-free class-incremental learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 30534–30544, June 2025.

[24] Xuan Liu and Xiaobin Chang. LoRA subtraction for drift-resistant space in exemplar-free continual learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 15308–15318, June 2025.

[25] Román Orús. Tensor networks for complex quantum systems. *Nature Reviews Physics*, 1(9):538–550, 2019.

[26] J Douglas Carroll and Jih-Jie Chang. Analysis of individual differences in multidimensional scaling via an n-way generalization of "Eckart-Young" decomposition. *Psychometrika*, 35(3):283–319, 1970.

[27] Richard A Harshman. Foundations of the PARAFAC procedure: Models and conditions for an "explanatory" multi-modal factor analysis. *UCLA working papers in phonetics*, 16(1):84, 1970.

[28] Ledyard R Tucker. Some mathematical notes on three-mode factor analysis. *Psychometrika*, 31(3):279–311, 1966.

[29] Ivan V Oseledets. Tensor-train decomposition. *SIAM Journal on Scientific Computing*, 33(5):2295–2317, 2011.

[30] Qibin Zhao, Guoxu Zhou, Shengli Xie, Liqing Zhang, and Andrzej Cichocki. Tensor ring decomposition. *arXiv preprint arXiv:1606.05535*, 2016.

[31] Ji Liu, Przemyslaw Musialski, Peter Wonka, and Jieping Ye. Tensor completion for estimating missing values in visual data. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 35(1):208–220, 2012.

[32] Yongxin Yang and Timothy M Hospedales. Deep multi-task representation learning: A tensor factorisation approach. In *International Conference on Learning Representations*, 2017.

[33] Miao Yin, Yang Sui, Siyu Liao, and Bo Yuan. Towards efficient tensor decomposition-based DNN model compression with optimization framework. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10674–10683, June 2021.

[34] Wei Dai, Jicong Fan, Yiming Miao, and Kai Hwang. Deep learning model compression with rank reduction in tensor decomposition. *IEEE Transactions on Neural Networks and Learning Systems*, 36(1):1315–1328, 2025.

[35] F. Verstraete, J. J. García-Ripoll, and J. I. Cirac. Matrix product density operators: Simulation of finite-temperature and dissipative systems. *Physical Review Letters*, 93:207204, Nov 2004.

[36] Michael Zwolak and Guifré Vidal. Mixed-state dynamics in one-dimensional quantum lattice systems: A time-dependent superoperator renormalization algorithm. *Physical Review Letters*, 93:207205, Nov 2004.

[37] Dan Hendrycks, Steven Basart, Norman Mu, Saurav Kadavath, Frank Wang, Evan Dorundo, Rahul Desai, Tyler Zhu, Samyak Parajuli, Mike Guo, Dawn Song, Jacob Steinhardt, and Justin Gilmer. The many faces of robustness: A critical analysis of out-of-distribution generalization. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 8340–8349, October 2021.

[38] Dan Hendrycks, Kevin Zhao, Steven Basart, Jacob Steinhardt, and Dawn Song. Natural adversarial examples. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 15262–15271, June 2021.

[39] Xingchao Peng, Qinxun Bai, Xide Xia, Zijun Huang, Kate Saenko, and Bo Wang. Moment matching for multi-source domain adaptation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, October 2019.

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

[41] Zifeng Wang, Zizhao Zhang, Sayna Ebrahimi, Ruoxi Sun, Han Zhang, Chen-Yu Lee, Xiaoqi Ren, Guolong Su, Vincent Perot, Jennifer Dy, et al. DualPrompt: Complementary prompting for rehearsal-free continual learning. In *European Conference on Computer Vision*, pages 631–648. Springer, 2022.

[42] Wei-Cheng Huang, Chun-Fu Chen, and Hsiang Hsu. OVOR: OnePrompt with virtual outlier regularization for rehearsal-free class-incremental learning. In *International Conference on Learning Representations*, 2024.

[43] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. An image is worth 16x16 words: Transformers for image recognition at scale. In *International Conference on Learning Representations*, 2021.

[44] Mathilde Caron, Hugo Touvron, Ishan Misra, Hervé Jégou, Julien Mairal, Piotr Bojanowski, and Armand Joulin. Emerging properties in self-supervised vision transformers. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 9650–9660, October 2021.

[45] Dimitri Panteli Bertsekas. *Nonlinear programming*. Athena Scientific, 1999. ISBN 9781886529007.

[46] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 1026–1034, 2015.

A Detailed proofs for analysis

This section provides the detailed proofs for Section 4.3. Let $\mathcal{M}$ denote the set of adapted modules. For a collection of adapted modules $\mathbf{W} = \{\mathbf{W}^{(\ell)}\}_{\ell \in \mathcal{M}}$, we define the aggregate Frobenius norm as

$$\|\mathbf{W}\|_F := \left(\sum_{\ell \in \mathcal{M}} \|\mathbf{W}^{(\ell)}\|_F^2 \right)^{1/2}. \tag{15}$$

All norms below are Frobenius norms unless otherwise specified.

A.1 Auxiliary lemmas

We first recall the standard smoothness inequality used in the proof.

Lemma 1 (Descent lemma [45])**.** *Let f be differentiable and locally L-smooth on a region containing the line segment between $\mathbf{U}$ and $\mathbf{V}$. Then*

$$f(\mathbf{V}) \le f(\mathbf{U}) + \langle \nabla f(\mathbf{U}), \mathbf{V} - \mathbf{U} \rangle + \frac{L}{2} \|\mathbf{V} - \mathbf{U}\|_F^2. \tag{16}$$

To bound the drift induced by task-wise normalization, the normalization operator is defined as follows. For any block $\mathbf{X}$,

$$\mathcal{N}_{\rho,\varepsilon}(\mathbf{X}) := \frac{\rho}{\max(\|\mathbf{X}\|_F, \varepsilon)} \mathbf{X}, \qquad \|\mathcal{N}_{\rho,\varepsilon}(\mathbf{X})\|_F \le \rho. \tag{17}$$

Lemma 2 (Drift under task-wise normalization)**.** *Under task-wise normalization, for any $t > q$,*

$$\|\mathbf{W}_t - \mathbf{W}_q\|_F \le (t-q)\sqrt{|\mathcal{M}|}. \tag{18}$$

Proof. Under task-wise normalization, the update of module ℓ for task s is

$$\Delta \mathbf{W}_s^{(\ell)} = \mathcal{N}_{1,\varepsilon}\big(\mathrm{MPO}(\mathcal{G}_s^{(\ell)})\big), \qquad \|\Delta \mathbf{W}_s^{(\ell)}\|_F \le 1, \tag{19}$$

where the inequality follows from Eq. (17). Since the task-wise model satisfies $\mathbf{W}_t^{(\ell)} = \mathbf{W}_0^{(\ell)} + \sum_{s=1}^{t} \Delta \mathbf{W}_s^{(\ell)}$, we have

$$\mathbf{W}_t^{(\ell)} - \mathbf{W}_q^{(\ell)} = \sum_{s=q+1}^{t} \Delta \mathbf{W}_s^{(\ell)}. \tag{20}$$

By the triangle inequality and Eq. (19),

$$\|\mathbf{W}_t^{(\ell)} - \mathbf{W}_q^{(\ell)}\|_F \le \sum_{s=q+1}^{t} \|\Delta \mathbf{W}_s^{(\ell)}\|_F \le t - q. \tag{21}$$

Aggregating Eq. (21) over all adapted modules gives

$$\|\mathbf{W}_t - \mathbf{W}_q\|_F = \left(\sum_{\ell \in \mathcal{M}} \|\mathbf{W}_t^{(\ell)} - \mathbf{W}_q^{(\ell)}\|_F^2 \right)^{1/2} \le (t-q)\sqrt{|\mathcal{M}|}. \tag{22}$$

□

Next, we bound the drift induced by module-wise global normalization.

Lemma 3 (Drift under global normalization)**.** *Under module-wise global normalization, for any $t > q$,*

$$\|\mathbf{W}_t^{\mathrm{g}} - \mathbf{W}_0\|_F \le \tau\sqrt{|\mathcal{M}|}, \qquad \|\mathbf{W}_t^{\mathrm{g}} - \mathbf{W}_q^{\mathrm{g}}\|_F \le 2\tau\sqrt{|\mathcal{M}|}. \tag{23}$$

Proof. For each adapted module ℓ, global normalization gives

$$\widehat{\mathbf{S}}_t^{(\ell)} = \mathcal{N}_{\tau,\varepsilon}(\mathbf{S}_t^{(\ell)}), \qquad \mathbf{W}_t^{\mathrm{g},(\ell)} = \mathbf{W}_0^{(\ell)} + \widehat{\mathbf{S}}_t^{(\ell)}. \tag{24}$$

By Eq. (17), we have $\|\widehat{\mathbf{S}}_t^{(\ell)}\|_F \le \tau$. Therefore,

$$\|\mathbf{W}_t^{\mathrm{g}} - \mathbf{W}_0\|_F = \left(\sum_{\ell\in\mathcal{M}} \|\widehat{\mathbf{S}}_t^{(\ell)}\|_F^2\right)^{1/2} \le \tau\sqrt{|\mathcal{M}|}. \tag{25}$$

For two stages $q < t$, the triangle inequality and Eq. (25) give

$$\|\mathbf{W}_t^{\mathrm{g}} - \mathbf{W}_q^{\mathrm{g}}\|_F \le \|\mathbf{W}_t^{\mathrm{g}} - \mathbf{W}_0\|_F + \|\mathbf{W}_q^{\mathrm{g}} - \mathbf{W}_0\|_F \le 2\tau\sqrt{|\mathcal{M}|}. \tag{26}$$

□

The smoothness assumption in Lemma 1 only needs to hold on the parameter region connecting the compared model states. By Lemma 3, all GlobalNorm parameter states lie within the compact convex set $\mathcal{B}_\tau = \{\mathbf{W} : \|\mathbf{W} - \mathbf{W}_0\|_F \le \tau\sqrt{|\mathcal{M}|}\}$. If the task loss is twice continuously differentiable with respect to the adapted module weights on a neighborhood of this set, its Hessian is bounded, ensuring a finite smoothness constant L_q for all stages. Thus, the smoothness region required for GlobalNorm is independent of the number of subsequent tasks, whereas the corresponding region under TaskNorm may expand with the task sequence.

A.2 Proof of Theorem 1

The task-wise forgetting bound is proved first. Applying Lemma 1 with $f = \mathcal{L}_q$, $\mathbf{U} = \mathbf{W}_q$, and $\mathbf{V} = \mathbf{W}_t$ gives

$$\mathcal{L}_q(\mathbf{W}_t) \le \mathcal{L}_q(\mathbf{W}_q) + \langle \nabla\mathcal{L}_q(\mathbf{W}_q), \mathbf{W}_t - \mathbf{W}_q\rangle + \frac{L_q}{2}\|\mathbf{W}_t - \mathbf{W}_q\|_F^2. \tag{27}$$

Subtracting $\mathcal{L}_q(\mathbf{W}_q)$ from both sides and applying Cauchy–Schwarz inequality yield

$$\Delta\mathcal{L}_q^{\mathrm{task}}(q \to t) \le \|\nabla\mathcal{L}_q(\mathbf{W}_q)\|_F\|\mathbf{W}_t - \mathbf{W}_q\|_F + \frac{L_q}{2}\|\mathbf{W}_t - \mathbf{W}_q\|_F^2. \tag{28}$$

Combining Eq. (28) with Lemma 2, we obtain

$$\Delta\mathcal{L}_q^{\mathrm{task}}(q \to t) \le (t-q)\sqrt{|\mathcal{M}|}\|\nabla\mathcal{L}_q(\mathbf{W}_q)\|_F + \frac{L_q}{2}|\mathcal{M}|(t-q)^2. \tag{29}$$

The global forgetting bound can be proved in a similar manner. According to Lemma 1,

$$\mathcal{L}_q(\mathbf{W}_t^{\mathrm{g}}) \le \mathcal{L}_q(\mathbf{W}_q^{\mathrm{g}}) + \left\langle \nabla\mathcal{L}_q(\mathbf{W}_q^{\mathrm{g}}), \mathbf{W}_t^{\mathrm{g}} - \mathbf{W}_q^{\mathrm{g}}\right\rangle + \frac{L_q}{2}\|\mathbf{W}_t^{\mathrm{g}} - \mathbf{W}_q^{\mathrm{g}}\|_F^2, \tag{30}$$

and the inequality can be written as

$$\Delta\mathcal{L}_q^{\mathrm{global}}(q \to t) \le \|\nabla\mathcal{L}_q(\mathbf{W}_q^{\mathrm{g}})\|_F\|\mathbf{W}_t^{\mathrm{g}} - \mathbf{W}_q^{\mathrm{g}}\|_F + \frac{L_q}{2}\|\mathbf{W}_t^{\mathrm{g}} - \mathbf{W}_q^{\mathrm{g}}\|_F^2. \tag{31}$$

Combining Eq. (31) with Lemma 3, we obtain

$$\Delta\mathcal{L}_q^{\mathrm{global}}(q \to t) \le 2\tau\sqrt{|\mathcal{M}|}\|\nabla\mathcal{L}_q(\mathbf{W}_q^{\mathrm{g}})\|_F + 2L_q|\mathcal{M}|\tau^2. \tag{32}$$

This proves the general bounds in Theorem 1.

If $\mathbf{W}_q$ and $\mathbf{W}_q^{\mathrm{g}}$ are stationary points of $\mathcal{L}_q$ in the full adapted-module weight space, then the gradient terms in Eqs. (29) and (32) vanish. Therefore,

$$\Delta\mathcal{L}_q^{\mathrm{task}}(q \to t) \le \frac{L_q}{2}|\mathcal{M}|(t-q)^2, \qquad \Delta\mathcal{L}_q^{\mathrm{global}}(q \to t) \le 2L_q|\mathcal{M}|\tau^2. \tag{33}$$

This proves the simplified bounds.

A.3 Extension to H-NoTA

We now show that H-NoTA inherits the same global boundedness guarantee. For each adapted module ℓ, Eq. (7) can be written as

$$\mathbf{W}_t^{\mathrm{g},(\ell)} = \mathbf{W}_0^{(\ell)} + \mathcal{N}_{\tau,\varepsilon}\left(\widetilde{\mathbf{S}}_t^{(\ell)}\right), \tag{34}$$

where $\widetilde{\mathbf{S}}_t^{(\ell)}$ denotes the pre-normalized cumulative update after incorporating the task-level coefficients and inter-core modulation matrices. Since Eq. (34) has the same outer global normalization form as Eq. (24), Lemma 3 applies directly after replacing $\mathbf{S}_t^{(\ell)}$ with $\widetilde{\mathbf{S}}_t^{(\ell)}$. Hence,

$$\|\mathbf{W}_t^{\mathrm{g}} - \mathbf{W}_0\|_F \le \tau\sqrt{|\mathcal{M}|}, \qquad \|\mathbf{W}_t^{\mathrm{g}} - \mathbf{W}_q^{\mathrm{g}}\|_F \le 2\tau\sqrt{|\mathcal{M}|}. \tag{35}$$

Applying Lemma 1 as in the proof of Theorem 1 gives

$$\Delta\mathcal{L}_q^{\text{H-NoTA}}(q \to t) \le 2\tau\sqrt{|\mathcal{M}|}\|\nabla\mathcal{L}_q(\mathbf{W}_q^{\mathrm{g}})\|_F + 2L_q|\mathcal{M}|\tau^2. \tag{36}$$

If the stage-q H-NoTA model is a stationary point of $\mathcal{L}_q$ in the full adapted-module weight space, then the first term vanishes, and we obtain

$$\Delta\mathcal{L}_q^{\text{H-NoTA}}(q \to t) \le 2L_q|\mathcal{M}|\tau^2. \tag{37}$$

Thus, H-NoTA retains the same sequence-length independent forgetting bound induced by the outer global normalization.

B Additional method details

B.1 Illustration of MPO contraction

Fig. 5 illustrates how the MPO reconstruction in our method maps a sequence of tensor cores back to an incremental matrix. A dense update in $\mathbb{R}^{d_{\text{out}} \times d_{\text{in}}}$ is first represented in a tensorized space by factorizing the input and output dimensions into multiple modes. In the example shown in the figure, $d_{\text{in}} = \prod_{k=1}^{3} d_{\text{in},k}$ and $d_{\text{out}} = \prod_{k=1}^{3} d_{\text{out},k}$. The MPO cores are then contracted along their bond dimensions according to Eq. (6), producing a tensorized update indexed by the decomposed input and output modes. Finally, this tensorized update is reshaped into the original matrix form and added to the pretrained weight as the task-specific adapter.

This construction also illustrates the parameter efficiency of the MPO adapter. A dense update for a square $n \times n$ weight requires $O(n^2)$ parameters, whereas an MPO update stores only the tensor cores, with parameter count $\sum_{k=1}^{L} r_{k-1} d_{\text{out},k} d_{\text{in},k} r_k$. By factorizing the input and output dimensions into multiple modes, MPO distributes the adaptation parameters across these modes and provides a compact structured update space for parameter-efficient CIL.

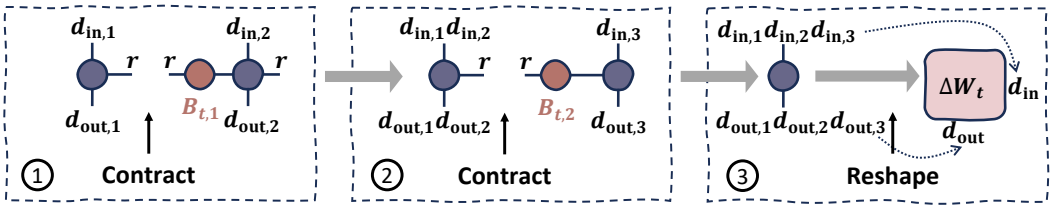


Figure 5: Illustration of MPO contraction and matricization. For clarity, we show the case with $L = 3$ tensor cores and omit the normalization procedure. The input and output dimensions are factorized as $d_{\text{in}} = \prod_{k=1}^{3} d_{\text{in},k}$ and $d_{\text{out}} = \prod_{k=1}^{3} d_{\text{out},k}$. The MPO cores are contracted along the bond dimensions following Eq. (6), and the resulting tensorized update is reshaped back into the original matrix space.

B.2 Algorithm of NoTA and H-NoTA

Algorithm 1 summarizes the training procedure of NoTA and H-NoTA. For each new task, we initialize a new set of MPO structural parameters while keeping all historical structural parameters

Algorithm 1 Training procedure of NoTA and H-NoTA

Require: Task sequence $\{\mathcal{D}_t\}_{t=1}^{T}$; frozen pretrained weights $\{\mathbf{W}_0^{(\ell)}\}_{\ell\in\mathcal{M}}$; adapted modules $\mathcal{M}$; normalization operator $\mathcal{N}_{\rho,\varepsilon}(\cdot)$; global budget τ; mode $\in \{\mathrm{NoTA}, \mathrm{H\text{-}NoTA}\}$.
Ensure: Adapted model after learning all tasks.

for $t = 1, \ldots, T$ **do**
  Initialize the new MPO structural parameters $\{\mathcal{G}_t^{(\ell)}\}_{\ell\in\mathcal{M}}$ for task $\mathcal{T}_t$.
  if mode is H-NoTA **then**
    Initialize the task-level coefficient α_t and inter-core modulation matrices $\{\mathbf{B}_{t,k}^{(\ell)}\}_{k=1}^{L-1}$ for each adapted module ℓ.
  end if
  Freeze all historical structural parameters $\{\mathcal{G}_s^{(\ell)}\}_{s=1}^{t-1}$.
  if mode is H-NoTA **then**
    Keep all task-level coefficients $\{\alpha_s\}_{s=1}^{t}$ and all modulation matrices $\{\mathbf{B}_{s,k}^{(\ell)}\}_{s=1,k=1}^{t,L-1}$ trainable.
  end if
  while training on current dataset $\mathcal{D}_t$ **do**
    for each adapted module $\ell \in \mathcal{M}$ **do**
      if mode is NoTA **then**
        for $s = 1, \ldots, t$ **do**
          $\Delta\mathbf{W}_s^{(\ell)} \leftarrow \mathcal{N}_{1,\varepsilon}\Big(\mathrm{MPO}(\mathcal{G}_s^{(\ell)})\Big)$
        end for
        $\mathbf{S}_t^{(\ell)} \leftarrow \sum_{s=1}^{t} \Delta\mathbf{W}_s^{(\ell)}$
        ▷ *For NoTA, historical updates with $s < t$ are fixed and can be pre-computed.*
      else
        for $s = 1, \ldots, t$ **do**
          $\Delta\mathbf{W}_s^{(\ell)} \leftarrow \mathcal{N}_{1,\varepsilon}\Big(\mathrm{MPO}(\mathcal{G}_s^{(\ell)}, \{\mathbf{B}_{s,k}^{(\ell)}\}_{k=1}^{L-1})\Big)$
        end for
        $\mathbf{S}_t^{(\ell)} \leftarrow \sum_{s=1}^{t} \alpha_s \Delta\mathbf{W}_s^{(\ell)}$
      end if
      $\widehat{\mathbf{S}}_t^{(\ell)} \leftarrow \mathcal{N}_{\tau,\varepsilon}\Big(\mathbf{S}_t^{(\ell)}\Big)$
      $\mathbf{W}_t^{\mathrm{g},(\ell)} \leftarrow \mathbf{W}_0^{(\ell)} + \widehat{\mathbf{S}}_t^{(\ell)}$
    end for
    Compute the training loss on $\mathcal{D}_t$ using $\{\mathbf{W}_t^{\mathrm{g},(\ell)}\}_{\ell\in\mathcal{M}}$.
    ▷ *Including the classification loss and the anchoring term described in Appendix C.1.*
    Update the trainable parameters by back-propagation.
  end while
end for

frozen. In NoTA, each task-specific MPO update is first normalized at the task-level, then all normalized updates are summed and passed through global normalization before being added to the pretrained weight. Since the historical normalized updates in NoTA are fixed after their corresponding tasks are learned, they can be pre-computed before training the next task. H-NoTA follows the same global normalization framework, but further introduces task-level scaling coefficients and inter-core modulation matrices, which remain trainable across tasks and allow historical adapters to be reweighted without modifying their frozen structural directions. This procedure is applied to all adapted modules, such as the query and value projections in ViT.

C Additional experimental results

C.1 Implementation details and hyperparameters

We follow the experimental settings of recent parameter-efficient CIL baselines, with dataset-specific training hyperparameters summarized in Tab. 7. The class-incremental task sequences follow the standard protocol described in Sec. 5.1, where the class order is randomly shuffled according to a seed

Table 7: Training hyperparameters for different experimental settings. The IN-21K backbone is pretrained on ImageNet-21K and fine-tuned on ImageNet-1K.

Dataset	Task splits	Backbone	Optimizer	Epochs	Base learning rate
ImageNet-R	5 / 10 / 20	ViT-B/16, IN-21K	SGD	30	1×10^{-2}
ImageNet-A	5 / 10 / 20	ViT-B/16, IN-21K	Adam	20	4×10^{-3}
DomainNet	5	ViT-B/16, IN-21K	SGD	10	2×10^{-2}
ImageNet-A	10	ViT-B/16, DINO	Adam	20	6×10^{-3}

and partitioned into disjoint tasks. The seed controls the class permutation, parameter initialization, and stochastic training operations. All experiments use a batch size of 128. The main results are reported as mean and standard deviation over five independent runs. The analyses of update spectra, historical interference, and representation drift use seed 1993. MPO adapters are inserted into the query and value projections of all ViT attention layers. Each MPO chain contains $L = 3$ cores, with the bond rank r specified in the corresponding experimental tables. Since the query and value projection matrices in ViT-B/16 have input and output dimensions of 768, we factorize both dimensions as $768 = 8 \times 12 \times 8$, so the input and output mode sizes of the three MPO cores are set to $(8, 12, 8)$. The initial learning rate of the middle MPO core is set to $20\times$ that of the two edge cores. For DomainNet, we use the official six-domain train/test splits and partition the classes into five disjoint tasks of 69 classes each. Further ablation studies on key hyperparameters are provided in Sec. C.3. All experiments are conducted on a single GPU (NVIDIA RTX 6000 Ada / A6000). The complete experimental configurations are available in our code repository.

Initialization. For each newly introduced task, all MPO structural cores except the last one are initialized with Kaiming uniform initialization [46], while the last core is initialized to zero so that the current task adapter starts from a zero reconstructed update. In H-NoTA, all task-level coefficients are initialized as $\alpha_t = 1$, and all inter-core modulation matrices $\mathbf{B}_{t,k}$ are initialized as identity matrices. Thus, H-NoTA starts from the same MPO reconstruction as NoTA before learning task-level reweighting and inter-core modulation. When learning a new task, historical MPO structural cores are kept frozen, while previous task-level coefficients and inter-core modulation parameters remain trainable. To avoid abrupt changes in historical contributions, we apply a lightweight L2 anchoring term to these coefficients with strengths 0.01 for task-level coefficients and 0.1 for inter-core modulation parameters. Under the outer global normalization, α_1 is scale-redundant during the first task because only one task direction is present in the cumulative update. After multiple tasks have been learned, however, α_t acts as a relative task-level coefficient for reweighting different task directions. For numerical stability, the normalization denominator uses $\max(\|\cdot\|_F, \varepsilon)$ as in Eq. (3), and ε is set to 1×10^{-4} by default.

Hierarchical coefficient optimization. In H-NoTA experiments, we refined the learning rates of the hierarchical coefficients while keeping the pretrained backbones, task splits, training schedules, MPO configurations, and evaluation procedures unchanged. Specifically, we conducted a coarse search for the learning rate multipliers of the current and historical inter-core modulation matrices. The learning rate multipliers for current-task and historical modulation matrices $\mathbf{B}_{s,k}$ were selected as 1.0 and 0.1 from $\{0.5, 1.0, 2.0\}$ and $\{0.05, 0.1, 0.2\}$, respectively. The task-level coefficients α_s use the same base learning rate for the current task ($s = t$) and historical tasks ($s < t$).

C.2 Additional comparisons with existing baselines

We further compare H-NoTA with CL-LoRA [23] and LoRA-Sub [24] under the same pretrained backbone and comparable per-task parameter budgets. As shown in Tab. 8, H-NoTA consistently outperforms LoRA-Sub on ImageNet-R and achieves competitive performance with CL-LoRA. H-NoTA also achieves higher Acc and AAA than CL-LoRA on ImageNet-A. More importantly, CL-LoRA evaluates each input separately with all T task-specific adapters during inference, resulting in 20 inference paths and 698.90 GFLOPs at $T = 20$. In contrast, H-NoTA aggregates historical task updates into a single cumulative inference path, requiring only 45.52 GFLOPs, approximately $15.4\times$ less than CL-LoRA. Although LoRA-Sub has slightly lower inference cost, it requires data scanning and 48 SVDs for subspace construction at task transitions, whereas H-NoTA requires no

Table 8: Performance and computational comparison with additional baselines on ImageNet-R (IN-R) and ImageNet-A (IN-A). Acc and AAA are reported as mean with standard deviation. GFLOPs and #Paths denote online inference cost and the number of inference paths at $T = 20$, respectively. Transition cost denotes additional preprocessing between tasks.

Method	IN-R, $T = 10$		IN-R, $T = 20$		IN-A, $T = 10$		GFLOPs	#Paths	Transition cost
	Acc	AAA	Acc	AAA	Acc	AAA			
LoRA-Sub	$76.43_{(0.27)}$	$80.01_{(0.21)}$	$74.71_{(0.48)}$	$78.48_{(0.32)}$	$59.97_{(0.44)}$	$65.99_{(0.89)}$	41.57	1	Scan + 48 SVDs
CL-LoRA	$79.56_{(0.16)}$	$82.85_{(0.24)}$	$76.86_{(0.23)}$	$80.36_{(0.42)}$	$55.80_{(1.06)}$	$62.70_{(0.55)}$	698.90	20	None
H-NoTA	$79.10_{(0.25)}$	$81.88_{(0.39)}$	$76.88_{(0.51)}$	$80.00_{(0.40)}$	$59.33_{(1.47)}$	$65.47_{(1.12)}$	45.52	1	None

such preprocessing. These results highlight the favorable balance of accuracy, inference efficiency, and task-transition overhead achieved by H-NoTA.

C.3 Additional ablation studies

In this section, we provide additional ablation studies to examine the effects of TaskNorm and GlobalNorm and the individual contributions of the hierarchical coefficients α_t and $\mathbf{B}_{t,k}$. We further analyze the sensitivity of H-NoTA to key hyperparameters, including the global budget τ, the MPO rank, the number of MPO cores, and the core-specific learning rates. Unless otherwise specified, the hyperparameter sensitivity studies are conducted using H-NoTA with a single random seed, while other training settings are kept fixed within each study.

Effect of hierarchical coefficients. We conduct factorized ablations of the task-level coefficients α_t and inter-core modulation matrices $\mathbf{B}_{t,k}$ to further examine the contributions of the hierarchical coefficients in H-NoTA. As shown in Tab. 9, H-NoTA further improves upon each variant across most settings, demonstrating the benefit of combining task-level reweighting with inter-core modulation.

Table 9: Ablation of hierarchical coefficients on ImageNet-R (IN-R) and ImageNet-A (IN-A) with MPO rank $r = 4$. Acc and AAA are reported as mean with standard deviation.

Method	IN-A, $T = 10$		IN-R, $T = 10$		IN-R, $T = 20$	
	Acc	AAA	Acc	AAA	Acc	AAA
NoTA	$\mathbf{59.48}_{(0.96)}$	$64.98_{(0.53)}$	$77.19_{(0.51)}$	$80.15_{(0.58)}$	$72.83_{(0.29)}$	$76.30_{(0.41)}$
α-only	$58.76_{(1.24)}$	$65.20_{(0.92)}$	$78.98_{(0.23)}$	$81.62_{(0.37)}$	$76.53_{(0.51)}$	$79.70_{(0.32)}$
B-only	$58.93_{(1.28)}$	$65.25_{(1.00)}$	$79.02_{(0.16)}$	$81.65_{(0.35)}$	$76.39_{(0.65)}$	$79.72_{(0.37)}$
H-NoTA	$59.33_{(1.47)}$	$\mathbf{65.47}_{(1.12)}$	$\mathbf{79.10}_{(0.25)}$	$\mathbf{81.88}_{(0.39)}$	$\mathbf{76.88}_{(0.51)}$	$\mathbf{80.00}_{(0.40)}$

Ablation on GlobalNorm and TaskNorm. As depicted in Fig. 2, TaskNorm and GlobalNorm lead to different cumulative magnitude trajectories. To conduct a more detailed analysis, we provide an additional quantitative comparison under different task sequence lengths. Tab. 10 compares matrix-based and MPO-based adapters on ImageNet-A with $T = 5$ and $T = 20$, where the matrix-based adapter uses rank 10 and the MPO-based adapter uses rank 4. Under $T = 5$, GlobalNorm provides moderate gains for both adapters. For the matrix-based adapter, Acc/AAA improve from $55.87/63.17$ to $58.35/64.00$; for the MPO-based adapter, they improve from $60.51/65.98$ to $61.03/66.21$. In this short sequence setting, the cumulative magnitude under TaskNorm is still below the global budget, and GlobalNorm calibrates the accumulated adapter to a more stable scale rather than simply shrinking it.

The advantage becomes more pronounced under the longer $T = 20$ sequence. For the MPO-based adapter, TaskNorm leads to a larger cumulative magnitude with $\Phi_T = 7.59$, while GlobalNorm calibrates it to $\Phi_T = 4.90$ and improves Acc/AAA from $54.30/62.10$ to $56.96/63.60$. The matrix-based adapter shows a similar trend, with Φ_T calibrated from 5.26 to 4.90 and Acc/AAA improved from $35.49/47.48$ to $49.71/56.94$. These results support the theoretical analysis in Section 4. Local task-wise control does not directly constrain the cumulative adapter, whose magnitude may either

be insufficient in short sequences or grow excessively in long sequences. In contrast, GlobalNorm imposes a stable global scale on the accumulated adapter, making its benefit increasingly evident as the number of tasks grows.

Table 10: Effect of TaskNorm and GlobalNorm under different task sequence lengths on ImageNet-A with the pretrained ViT-B/16 backbone. We compare a matrix-based adapter with rank 10 and an MPO-based adapter with rank 4. Note that the matrix-based adapter is the matrix low-rank counterpart of the MPO adapter under the same normalization framework. We report Acc, AAA, and the cumulative magnitude Φ_T at the final task.

Adapter	Norm.	Rank	#Para.	IN-A, $T = 5$			IN-A, $T = 20$		
				Acc	AAA	Φ_T	Acc	AAA	Φ_T
Matrix-based	TaskNorm	10	$369K$	55.87	63.17	2.51	35.49	47.48	5.26
Matrix-based	GlobalNorm	10	$369K$	58.35	64.00	4.90	49.71	56.94	4.90
MPO-based	TaskNorm	4	$68K$	60.51	65.98	2.97	54.30	62.10	7.59
MPO-based	GlobalNorm	4	$68K$	61.03	66.21	4.90	56.96	63.60	4.90

Sensitivity to the global budget τ. We further evaluate the sensitivity of H-NoTA to the global budget τ on ImageNet-A with $T = 20$ and MPO rank $r = 4$ under 5 runs. As shown in Tab. 11, the performance does not fluctuate drastically when τ varies from 0.5 to 2.0, indicating that H-NoTA is reasonably robust to this hyperparameter. Among the tested values, $\tau = 1$ achieves the best Acc and AAA, suggesting that it provides a better balance between retaining sufficient plasticity for new tasks and constraining the cumulative update. We therefore set $\tau = 1$ as the default value in our experiments.

Table 11: Sensitivity to the global budget τ.

Dataset	Tasks	τ	Acc	AAA
ImageNet-A	20	0.5	55.71	62.17
		1.0	56.68	63.07
		2.0	55.82	62.35

Sensitivity to the number of MPO cores. The number of MPO cores L controls the structural depth of the tensorized adapter. For the 768×768 query and value projections in ViT-B/16, we use the mode factorizations $(8, 12, 8)$, $(4, 6, 8, 4)$, and $(2, 4, 6, 4, 4)$ for $L = 3, 4, 5$, respectively. We evaluate H-NoTA with $L \in \{3, 4, 5\}$ on ImageNet-R with $T = 10$. As shown in Tab. 12, increasing L factorizes the update into finer tensor modes and reduces the number of adapter parameters. Meanwhile, the performance remains stable across different choices of L. In particular, $L = 3$ achieves the best Acc, while larger L values provide a more compact parameterization with comparable AAA. These results show that MPO-based adapters are robust to the choice of L, and justify our use of $L = 3$ as an effective default setting in the main experiments.

Table 12: Sensitivity to the number of MPO cores.

Dataset	Tasks	Number of cores L	#Para.	Acc	AAA
ImageNet-R	10	3	$246K$	79.54	82.04
		4	$160K$	79.02	81.83
		5	$108K$	79.08	81.97

Sensitivity to core-specific learning rates. For the default setting with $L = 3$, we further study whether different MPO cores benefit from different learning rates. We evaluate several learning rate multiplier schedules for the three cores, denoted as $(\mathcal{G}^{(1)}, \mathcal{G}^{(2)}, \mathcal{G}^{(3)})$. As shown in Tab. 13, assigning a larger learning rate to the middle core, i.e., $(1, 20, 1)$, achieves the best performance on ImageNet-R with $T = 20$. This suggests that the middle core plays a more active role in task-specific adaptation under our tensorized adapter, while uniformly increasing the learning rate of all cores is slightly less effective. Based on this observation, we use the $(1, 20, 1)$ multiplier as the default setting in the main experiments.

Table 13: Sensitivity to the core-specific learning rates.

Dataset	Tasks	LR multiplier	Acc	AAA
ImageNet-R	20	$(1, 1, 1)$	72.36	75.67
		$(1, 1, 20)$	73.44	76.68
		$(20, 1, 1)$	75.22	78.31
		$(20, 20, 20)$	77.46	80.35
		$(1, 20, 1)$	77.70	80.62

Sensitivity to the MPO rank. The MPO rank controls the trade-off between parameter efficiency and expressive capacity. We evaluate H-NoTA with $r \in \{4, 6, 8, 10\}$ on ImageNet-R with $T = 10$. As shown in Tab. 14, increasing the rank generally improves Acc and AAA, and $r = 10$ gives the best overall performance. At the same time, H-NoTA with $r = 4$ still achieves competitive results, indicating that the proposed tensorized adapter remains effective even under a compact parameter budget.

Table 14: Sensitivity to the MPO rank.

Dataset	Tasks	MPO rank r	#Para.	Acc	AAA
ImageNet-R	10	4	$68K$	78.80	81.66
		6	$143K$	79.14	81.90
		8	$246K$	79.54	82.04
		10	$376K$	79.98	82.54

D Supplementary materials

D.1 Broader impacts

This work contributes to rehearsal-free CIL by shifting attention from only designing task-specific adapters to understanding and controlling the cumulative update that the model carries across tasks. The proposed global normalization provides a simple and theoretically grounded way to limit accumulated drift, offering a reusable principle for future parameter-efficient continual learning methods beyond the particular MPO adapter used here. Meanwhile, NoTA and H-NoTA show that tensorized and hierarchically controlled adapters can improve the expressivity and storage efficiency of continual adaptation, suggesting a promising direction for building long-lived models that learn new tasks without retaining previous data. More broadly, this perspective may encourage future CIL research to jointly consider representation efficiency, cumulative stability, and plasticity, rather than treating adapter design and forgetting control as separate problems. As with other continual learning techniques, potential negative impacts may arise if the method is used to continually adapt models on biased, private, or harmful data without proper governance. Careful data curation, privacy protection, and deployment monitoring are therefore important when applying such methods in real-world settings.

D.2 Limitations

In this work, our theoretical analysis characterizes the structured expressivity of MPO-based adaptation and establishes forgetting bounds under GlobalNorm, while a more precise mathematical characterization of forgetting in continual learning remains open. A deeper understanding of how tensor rank, mode factorization, and hierarchical coefficients interact with different task distributions would further strengthen the theoretical foundation of this framework. In addition, although our experiments focus on visual classification benchmarks, extending normalized tensor adaptation to language, multimodal data, and other continual learning scenarios remains a promising direction for future research. Since retaining task-specific MPO adapters still leads to linear storage growth with the number of tasks, exploring the integration of tensorized parameterization with update merging could further improve the scalability of the framework.

D.3 Licenses for existing assets

We build our implementation on the publicly available codebase of [14], which is released under the MIT License. We acknowledge the original authors and contributors of SD-LoRA and follow its licensing terms. The ViT-B/16 backbone implementation and pretrained resources are used under their corresponding license terms, including Apache-2.0 for the public Google Research ViT repository. For baseline comparisons, we use publicly released implementations when available, and all third-party code used for evaluation is handled under its original license. The datasets used in this work, including ImageNet-A, ImageNet-R, and DomainNet, are employed under their original usage terms and are not redistributed. Our implementation of the proposed methods will be released under the MIT License.

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 outline the contributions of this work in the Abstract and Introduction.

   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: We discuss the limitations of our approach in Appendix D.2.

   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 present the main theoretical results in Sec. 4 and provide detailed analysis and proofs in Appendix A.

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: We report implementation details in Sec. 5.1 and Appendix C.1. We also provided detailed algorithmic pseudocode in Sec. B.2.

   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: [Yes]

Justification: We provide code implemented in PyTorch, and the datasets are publicly available.

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: We include the necessary training and testing details in Sec. 5.1 and Appendix C.1.

   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: The main results of this work are the averages obtained from five independent experiments. We also report the standard deviation of the results in Sec. 5.

   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: We discuss the GFLOPs for training and inference in Sec. 5.4 and report the computational resources used in Appendix C.1.

   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: [Yes]

   Justification: The research in this work adheres to the NeurIPS Code of Ethics in all respects.

   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: [Yes]

    Justification: We discuss the broader impacts of our work in Appendix D.1.

    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: Our paper poses no such risks.

    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: We report this information in Appendix D.3.

    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: [Yes]

    Justification: We will release the code with documentation.

    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: Our 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: Our 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: We only use LLMs for writing and editing purposes.

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.