MoRe: Modular Representations for Continual Representation Learning on Sequential Data

Jiaqi Sun[1,2] **Boyang Sun**[2] **Rasmy M. H.**[2] **Xiangchen Song**[1] **Kun Zhang**[1,2]
[1]Carnegie Mellon University
[2]Mohamed bin Zayed University of Artificial Intelligence
{jiaqisun, kunz1}@cmu.edu

Abstract

Continual learning requires models to adapt to new data while preserving previously acquired knowledge. At its core, this challenge can be viewed as principled one-step adaptation: incorporating new information with minimal interference to existing representations. Most existing approaches address this challenge by modifying model parameters or architectures in a supervised, task-specific manner. However, the underlying issue is representational: tasks require distinct yet structured representations that can be selectively updated without disrupting representations, while structure should reflect intrinsic organization in the data rather than task boundaries. In sequential data, time-delayed dependencies provide a natural signal for uncovering this structural organization, revealing how fundamental representations give rise to more specific ones. Inspired by the modular organization of the human brain, we propose MoRe, a framework that aims to derive a specific type of modular representation from the given sequential data and leverage it for continual learning with selective updates. MoRe decomposes knowledge into a hierarchy of fundamental and specific modules with identifiability guarantees, enabling principled module reuse, alignment, and expansion during adaptation while preserving old modules by construction. Experiments on synthetic benchmarks and real-world LLM activations demonstrate interpretable hierarchical structure and improved plasticity-stability trade-offs produced by MoRe, suggesting that MoRe may serve as a principled foundation for continual learning.

1 Introduction

With the growing attention to foundation models, training such models in a sequential manner, which is driven by limited learning capacity and the inherently sequential flow of data, has brought the long-standing problem of Continual Learning (CL), or lifelong learning. Generally speaking, the research problem of CL is how to avoid catastrophic forgetting while acquiring new knowledge or abilities during the learning procedure. In essence, without assuming that some part of the model remains stable, avoiding catastrophic forgetting is impossible.

Existing CL methods can be divided into supervised and unsupervised CL. Supervised CL directly models the abilities as tasks and aims to enable the model to conduct incremental tasks upon seeing more and more tasks. In this category, representative methods regularize parameter or output changes [12, 1, 14], replay information from previous tasks [23], or isolate parameters across tasks [24, 5, 2, 25, 19]; the architectural variants of the last case recognize that different tasks require different parts of the model [4, 38, 16, 34, 36, 35, 27], but how those parts are related is rarely addressed. Unsupervised CL, on the other hand, focuses on proper representation learning [22, 9], yet most of its methods still operate via replay, distillation, regularization, or constraint modification [18, 6, 7, 29, 17]. Across both categories, decisions about what to preserve and what to update are largely heuristic.

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

Accordingly, the core of CL is representational: the specialization of model components determines what can be reused and what ought to be updated. We therefore tackle the problem of *continual representation learning*: how can a model form representations that grow cumulatively, across sequentially observed scenarios, without catastrophic forgetting? This mirrors how cognition operates: understanding is updated continuously throughout life, and tasks are addressed by selecting the relevant pieces of understanding rather than by reshaping the system into a task-specific tool. Once rich representations underlying this understanding are achieved, only a few samples are needed to assemble them for downstream tasks; the substantive challenge is the representation itself.

Inspired by the specialization and composition of modules in the human brain [28, 31], we propose Modular Representations (MoRe) for principled continual representation learning. The core idea is that representations should consist of modules organized by a hierarchy from *fundamental* to *specific*, where fundamental modules are reused as building blocks for more specific ones. Crucially, this hierarchy should reflect intrinsic organization in the data rather than be imposed by task boundaries or architectural design.

Interestingly, we find that sequential data offer a particularly clear setting to enable identifying such modular structure. Cross-layer time-delayed effects flow unidirectionally from fundamental to specific layers, providing the asymmetry needed to make the hierarchy identifiable up to within-layer indeterminacies and a compatible layer order. Identifiability is what turns modules from a useful inductive bias into well-defined objects: with identifiable modules, decisions about what to reuse and what to expand at each adaptation step become principled rather than heuristic, and the global representation can be recovered incrementally from a sequence of partial views.

Our contributions are as follows. (1) We propose Modular Representations (MoRe) for sequential data, organizing latent mechanisms into a hierarchy grounded in the data-generating process and supporting selective, minimal updates across stages. (2) We establish identifiability guarantees under cross-layer time-delayed dependencies and develop estimation and reuse procedures, with preservation-first expansion and selective adaptation under capacity constraints. (3) On controlled synthetic benchmarks, we demonstrate factor recovery, accurate reuse, compact library growth, and plasticity–stability trade-offs. On real-world LLM activation data, we evaluate representation retention, category selectivity, and downstream informativeness in one-step and continual learning settings.

2 Problem Setting

General Continual Representation Learning We consider observations $\mathbf{X}_t \in \mathcal{X}$ from an underlying global time-series process, where t indexes time. At learning stage s, the learner receives a batch of data $\mathcal{D}_s$ from a stage-specific distribution P_s, with $\text{supp}(P_s) \subseteq \text{supp}(P)$, where P denotes the corresponding global observation distribution. Observations within each batch retain their temporal dependence. Different stages may expose different regions of the latent space or different combinations of latent factors, while sharing the generative mechanisms defined below. Each batch therefore provides only a partial view of the global process.

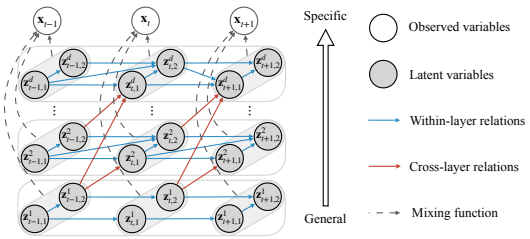

Figure 1: A hierarchical generative model for sequential data. Cross-layer influences flow from general to specific layers at positive time lags; within-layer effects may be instantaneous or time-delayed.

The goal is to learn an encoder $f : \mathcal{X} \to \tilde{\mathcal{Z}}$ incrementally from $\mathcal{D}_1, \ldots, \mathcal{D}_s$, without requiring joint access to all batches. The encoder is expected to learn more information of the latent structure with newly observed data while retaining its ability to represent previously observed data. We refer to this setting as *General Continual Representation Learning*. To support selective adaptation, we seek a representation structure that specifies which components can be reused and which require updating or expansion. We distinguish the underlying generative variables $\mathbf{Z}_t$ from the learned representations $\tilde{\mathbf{Z}}_t = f(\mathbf{X}_t)$.

Data Generating Process with Modular Representations We view real-world observations as arising from structured processes where general knowledge influences more specific knowledge, as characterized later. This motivates organizing the latent representations into a hierarchy: lower layers capture general, reusable factors, and higher layers capture more specific factors whose computation depends on them.

Specifically, the observation $\mathbf{X}_t$ is generated from d latent layers $\mathbf{Z}_t^1, \ldots, \mathbf{Z}_t^d$, with $\mathbf{Z}_t^i \in \mathbb{R}^m$, via

$$\mathbf{X}_t = g(\mathbf{Z}_t) \in \mathcal{X} \subseteq \mathbb{R}^n, \qquad \mathbf{Z}_t = \left[(\mathbf{Z}_t^1)^\top, \ldots, (\mathbf{Z}_t^d)^\top\right]^\top \in \mathcal{Z} \subseteq \mathbb{R}^{dm}. \tag{1}$$

Here, $g : \mathcal{Z} \to \mathcal{X}$ is a nonlinear mixing function that is invertible. We assume $dm \leq n$ and the regularity conditions specified in Section 3. Among layers $\mathbf{Z}_t^i$, $i = 1, 2, \cdots, d$, we distinguish fundamental ones from specific ones in the sense that the former can generate the latter, with a time delay, but not vice versa. Layers are indexed from general to specific, i.e., if $i < j$, $\mathbf{Z}_t^i$ is more fundamental relative to $\mathbf{Z}_t^j$.

Within a layer, the latents operate at a comparable level of abstraction, so we allow both instantaneous and time-delayed interactions. Thus, for the k-th dimension of layer i, we have

$$Z_{t,k}^i = h_{i,k}\Big(\mathbf{Z}_{t,\mathrm{pa}_i(k)}^i, \{\mathbf{Z}_{t-\tau}^i\}_{\tau=1}^L, \{\mathbf{Z}_{t-\tau}^j : 1 \leq j < i,\ 1 \leq \tau \leq L\}, \epsilon_{t,k}^i\Big), \tag{2}$$

where $\mathrm{pa}_i(k)$ denotes the instantaneous parents of coordinate k within layer i, and L is the maximum lag. We assume that instantaneous dependencies are acyclic and that the exogenous noise variables are mutually independent across coordinates, layers, and time. Note that the index t denotes timesteps within the sequential data, whereas later we use stages to refer to the steps of continual learning.

From Modular Generative Process to Continual Learning This generative structure motivates an encoder composed of layer-wise modules. We will show in Section 3 that, under proper assumptions, the latent layers are identifiable up to the within-layer ambiguities and admissible layer orderings specified there. This result provides a correspondence between learned modules and underlying latent layers, giving a structural basis for selective adaptation. Furthermore, Section 4 introduces the estimation and adaptation criteria used to decide which modules to reuse, refine, or expand.

3 Theoretical Guarantees

We establish the identification properties of the proposed hierarchical generative model in two steps. First, Theorem 1 excludes cross-layer mixing while permitting transformations within a layer. Second, Theorem 2 shows that the hierarchical mask produces a compatible organization and gives a sufficient condition for exact recovery. For simplicity, we consider the single-lag case, $L = 1$; this can immediately extended to multiple lags, as seen from [15], which deals with a related problem. Formal assumptions and proofs are provided in the appendix.

Throughout, the true and estimated models satisfy the generative restrictions introduced above, with d layers and m latents per layer known during estimation. Observational equivalence of two models means that they induce the same distribution over observation sequences.

Theorem 1 (Identifiability of Latent Variables)**.** *Suppose the observed sequential data $\mathbf{X}_t$ follow the data-generating process with modular representations, and that the true and estimated models are observationally equivalent. We formulate the following conditions informally below; their mathematical treatment is given in Appendix A.2:*

- *C1 (Smooth and Positive Density) The latent transition density varies smoothly and remains positive throughout the latent domain.*
- *C2 (Sufficient Variability) Different past latent values produce sufficiently distinct changes in the current conditional distribution to rule out cross-layer mixing.*
- *C3 (Sparse Latent Process for Each Layer) Within-layer dependency patterns distinguish individual latents from their neighbors, either directly or through a child.*
- *C2-ind (Reduced Sufficient Variability) Without instantaneous dependencies, variation with the past need only distinguish individual latents.*

Under C1–C2, any recovered latent is a mixture of only the latents within the same layer in the true model. Further assuming C3 implies component-wise identifiability, i.e., each recovered latent is a function of only one true latent. Additionally, in the special case without instantaneous dependencies, C1 and C2-ind suffice for component-wise identifiability, which excludes cross-layer mixing, even though C2 may not hold.

Proof Sketch Since instantaneous dependencies occur only within a layer, conditional on the past, current latent layers are mutually independent. Combining this factorization with sufficient variability rules out cross-layer mixing, leaving only within-layer transformations. Under the additional conditions for the C3 refinement, existing identification results resolve this remaining ambiguity. Without instantaneous dependencies, the argument reduces to the conditionally independent setting of [37]. Full proof is provided in Appendix A.2.

Having constrained latent mixing, we next characterize the hierarchical organization of the recovered latents. An assignment of their order is compatible if no true time-delayed influences go from a later estimated layer to an earlier one, as shown below.

Theorem 2 (Identification of a Compatible Hierarchy)**.** *Suppose the conditions of Theorem 1 hold, or there are no instantaneous relations. Assume:*

C4 (Hierarchical mask) Each estimated layer contains m latents. Cross-layer delayed effects respect the estimated order, while instantaneous effects remain within layers.

Then any observationally equivalent estimated model produces a compatible hierarchical assignment.

Further assume the following condition holds.

C5 (Complete cross-layer dependencies) Every latent in an earlier true layer has a time-delayed dependency to every latent in each later true layer.

Then the ordered layer partition is uniquely identified. Please find formal conditions in Appendix A.3.

Proof Sketch The identification result preserves dependencies between recovered blocks, or between individual latents in the conditionally independent case. An assignment that reverses a required dependency would exclude it through the hierarchical mask, contradicting observational equivalence. Therefore, the estimated assignment must be compatible. Under C5, an incorrect first layer necessarily displaces a true first-layer latent and reverses a required dependency. Applying the same argument successively identifies all layers. Full proof is provided in Appendix A.3.

These results provide the structural basis for the estimation and adaptation procedures developed in Section 4, which determine how modules are learned, reused, refined, or expanded.

4 Method

The identifiability results state that the learned modules are related to the structure of the true data generating process in a clean way. For estimation, we propose the framework MoRe for continual representation learning on sequential data. MoRe organizes the knowledge as a library of hierarchical latent modules, retaining both the latent mechanisms and the coordinate mappings that connect them to observations. At each stage, MoRe estimates the structure required by the incoming data, identifies compatible existing modules, and learns the remaining structure. We consider two operating regimes: *preservation-first learning*, which retains existing modules and expands when necessary, and *capacity-constrained learning*, which permits selective adaptation within explicit storage and update limits.

4.1 Initial Estimation and the Module Library

We first estimate the modular representations from the initial dataset $\mathcal{D}_0$ at stage 0, as the starting point of the persistent library. Guided by Theorems 1 and 2, the estimator captures the information needed to explain the data and enforces the assumed temporal structure, which supports the identification of the underlying mechanisms. Specifically, let Enc_θ map observations to estimated latents $\widehat{\mathbf{Z}}_t = \mathrm{Enc}_\theta(\mathbf{X}_t)$,

partitioned into d modules of m coordinates, and let $p_\psi\big(\widehat{\mathbf{Z}}_t \mid \{\widehat{\mathbf{Z}}_{t-\tau}\}_{\tau=1}^{L}\big)$ denote the temporal dependencies. When $n = dm$ and Enc_θ is invertible, the estimation objective is

$$\min_{\theta,\psi} \mathbb{E}_t\Big[-\log\big|\det \mathbf{J}_{\mathrm{Enc}_\theta}(\mathbf{X}_t)\big| - \log p_\psi\big(\widehat{\mathbf{Z}}_t \mid \{\widehat{\mathbf{Z}}_{t-\tau}\}_{\tau=1}^{L}\big)\Big] + \lambda_{\mathrm{h}}\,\mathcal{R}_{\mathrm{h}}(\theta,\psi), \tag{3}$$

where the first two terms constitute the conditional negative log-likelihood under the invertible observation-to-latent transformation, and the regularizer $\mathcal{R}_{\mathrm{h}}$ promotes sparse dependencies within the permitted hierarchical structure (C4). If Enc_θ is a normalizing flow, invertibility holds by construction and (3) is exact maximum likelihood. See more details of the implementation at Appendix B.1 for constructing $\mathcal{R}_h$ and the case when $dm < n$.

Constructing the module library. The fitted model provides both the latent mechanisms and the coordinate mappings needed to reuse them. We therefore initialize the library by retaining: the encoder, decoder, and the structural temporal model. The decoder is particularly important, because the central question to answer for reuse assessment in Section 4.2: what previous modules should be reused to help explain the current stage observations. In the following, we give a general example of linear case when $dm < n$ to clearly explain each step in our algorithm, with clear explanation of how the nonlinear case is handled differently if any. Appendix B.2 provides the full details of nonlinear case for the method section.

Since $dm < n$, we fit a coordinate basis $\mathbf{V}_0$ from the principal directions of $\mathcal{D}_0$, with mean μ_0. And then we project the data and the encoder and decoder are specified by

$$\widehat{\mathbf{Z}}_t = \mathrm{Enc}_0(\mathbf{X}_t) = \mathbf{F}_0^\top \mathbf{V}_0^\top (\mathbf{X}_t - \boldsymbol{\mu}_0), \qquad \widehat{\mathbf{X}}_t = \mathrm{Dec}_0(\widehat{\mathbf{Z}}_t) = \boldsymbol{\mu}_0 + \mathbf{V}_0 \mathbf{F}_0^{-\top} \widehat{\mathbf{Z}}_t, \tag{4}$$

where $\mathbf{F}_0$ is the learned invertible square transformation by Eq. 3, and we denote $\mathbf{A}_0 = \mathbf{V}_0 \mathbf{F}_0^{-\top}$. We write $\mathbf{a}_{0,i} = (\mathbf{A}_0)_{:,i}$ for the i-th column and $\mathbf{A}_0^{(i)} \in \mathbb{R}^{n\times m}$ for the block of module i.

Additionally, if in some scenarios the joint estimation is not computationally feasible, in Appendix B.4 we provide details of how to estimate the initial model in a layer-wise manner by making use of the conditional independence relations in the assymetric temporal causal process.

4.2 Reuse Assessment

After the library is initialized from $\mathcal{D}_0$, we consider the general question: given incoming data $\mathcal{D}_s$ and the saved library, we estimate the required latent width and identify reusable existing modules.

Still, in linear case, let $\mathbf{V}_s$ be an orthonormal basis for the incoming observation span, with width $\widehat{r}_s$ estimated by thresholding its singular values. For each saved decoder direction $\mathbf{a}_i$, define

$$q_i = \frac{\|\mathbf{V}_s^\top \mathbf{a}_{s-1,i}\|_2}{\|\mathbf{a}_{s-1,i}\|_2}, \qquad \mathbf{a}_{s-1,i} = (\mathbf{A}_{s-1})_{:,i}, \tag{5}$$

which is well defined under component-wise identifiability (Theorem 1), since q_i is invariant to rescaling of coordinate i. We retain directions with $q_i \geq \tau$, selecting the highest-scoring $\widehat{r}_s$ directions if necessary. The comparison takes place in the common observation space, rather than between stage-specific PCA coordinates.

For nonlinear representations, we compare coordinate-specific decoder responses instead of fixed directions. A current-data representation is fitted and matched to the saved model, with width and matching selected using validation data. Details are provided in Appendix B.2.

In either implementation, let R_s denote the selected old coordinates. Accordingly, the uncovered demand is $d_s = \max(0, \widehat{r}_s - |R_s|)$.

4.3 Preservation-First Learning with Expansion

After deciding the reused modules what the number of uncovered demanded modules, we first introduce the preservation-first regime, which learns incoming structure while preserving the existing library. MoRe reuses the coordinates in R_s, introduces d_s coordinates for uncovered demand, and retains inactive old coordinates for future reuse.

In the linear case, let $\mathbf{A}_{s-1,R_s}$ contain the retained decoder directions. The incoming decoder is

$$\widehat{\mathbf{X}}_t = \boldsymbol{\mu}_s + \mathbf{A}_{s-1,R_s}\widehat{\mathbf{Z}}_{t,R_s} + \mathbf{A}_{s,\mathrm{new}}\widehat{\mathbf{Z}}_{t,\mathrm{new}}, \tag{6}$$

where $\widehat{\mathbf{Z}}_{t,\mathrm{new}}$ has d_s coordinates. An identification fit on $\mathcal{D}_s$ yields candidate coordinates; those matched to retained directions are discarded, and the rest initialize the new ones. Retained directions stay fixed while the new coordinates and their permitted temporal interactions are learned, and the library appends the additions and this stage's preprocessing and encoding configuration without overwriting earlier ones. Notably, new directions need not lie in the initial PCA subspace, and a new coordinate may be upstream of an existing one. When both are observed in $\mathcal{D}_s$, such new-to-old dependencies are learned while the saved old configuration is kept.

For nonlinear representations, a shared decoder replaces the linear direction matrix. The construction and training objective are given in Appendix B.2.

4.4 Selective Adaptation under Capacity Constraints

When storage is limited, MoRe accommodates incoming structure by first modifying selected existing coordinates. Let C_s be the maximum number of coordinates the library may hold and Q_s the maximum number of coordinates added or modified at stage s. MoRe adds

$$d_s^{\mathrm{add}} = \min\{d_s,\ C_s - |\mathcal{L}_{s-1}|,\ Q_s\} \tag{7}$$

new coordinates and adapts at most $Q_s - d_s^{\mathrm{add}}$ existing ones; demand beyond these limits remains unmet.

In the linear case, adapting coordinate i updates its encoder row, its decoder direction $\mathbf{a}_{s-1,i}$, and its temporal equation. Coordinates are ranked by

$$\pi_i = (1 - q_i) + \beta\, \Delta_i^{\mathrm{tem}}, \tag{8}$$

where q_i is the observation compatibility from Section 4.2 and Δ_i^{tem} is the gain in temporal prediction obtained by refitting the transition of coordinate i on $\mathcal{D}_s$; the second term is needed because dynamics can change without changing the observation span. Among highly ranked coordinates, MoRe prefers those with fewer descendants in the estimated transition graph.

The selected parameters are then trained on $\mathcal{D}_s$ with the objective of Eq. (3), with all other parameters frozen, and the library replaces the modified blocks and appends the new coordinates. For nonlinear representations, coordinate effects share network parameters, so we isolate permitted changes with a frozen backbone and coordinate-associated residual networks (Appendix B.2).

4.5 What Each Regime Preserves

Preservation-first learning never modifies a stored stage configuration (preprocessing, encoder, decoder, and temporal model), so every earlier representation is reproduced exactly: there is no forgetting of saved representations by construction. In addition, retained modules keep their observation contributions, through fixed decoder directions in the linear case and through a decoder that agrees with the saved one when new coordinates are zero in the nonlinear case. Selective adaptation guarantees less: protected coordinates keep their encoder outputs, but decoders and temporal predictions that depend on modified coordinates may change. The regime and resource limits are set externally; MoRe does not predict whether currently inactive coordinates will be useful later.

5 Experiments

Synthetic experiments with known ground truth test identifiability, reuse, stage order, long-horizon growth, and adaptation under limited capacity (S-RQ1–5); LLM activation experiments test category selectivity, label efficiency, and continual learning (RW-RQ1–3). Preservation-first learning is the primary pipeline; capacity-constrained adaptation is evaluated in S-RQ5.

5.1 Synthetic Experiments

Setup. We generate sequential data following Equations (1) and (2), with linear ($dm \in \{4, \ldots, 128\}$) and nonlinear ($dm \in \{4, \ldots, 16\}$) mixing across $d \in \{2, 3, 4\}$ layers. S-RQ1 sweeps

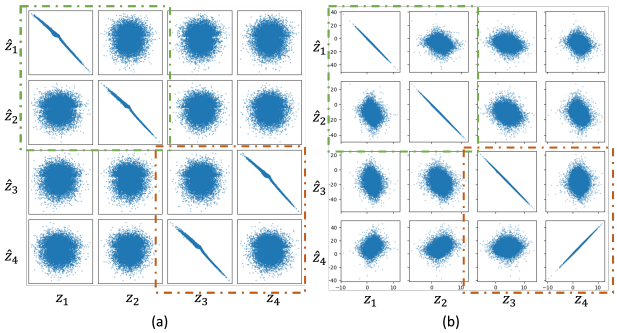

Figure 2: Ground-truth versus estimated latents for (a) nonlinear and (b) linear mixing in the two-layer, four-variable setting. Dashed blocks mark layers.

Type	Layer dimensions	MCC
Linear	$[2, 2]$	0.99
	$[4, 4]$	0.99
	$[8, 8, 8]$	0.99
	$[16, 16, 16, 16]$	0.93
	$[32, 32, 32, 32]$	0.85
Nonlinear	$[2, 2]$	0.98
	$[4, 4]$	0.98
	$[8, 8]$	0.98
	$[3, 3, 3]$	0.97
	$[4, 4, 4, 4]$	0.96

Table 1: Initial latent recovery.

Table 2: Nonlinear reuse (63 scenario–seed combinations per condition).

Condition	Gate acc. ↑	MCC ↑
Shared, clean	0.984	0.977
Shared, noise 0.02	0.984	0.969
Shared, noise 0.05	0.992	0.964
Rotated target	0.500	0.965

Table 3: Dormant recurrence at capacity nine (mean $\pm$ SD over data seeds).

Policy	New MCC ↑	Forget. ↓
Preserve	$.559 \pm .005$	$.000 \pm .000$
Hierarchy-guided	$.901 \pm .018$	$.019 \pm .004$
Random	$.740 \pm .018$	$.041 \pm .014$
Full fine-tuning	$.780 \pm .026$	$.112 \pm .005$

dimensionality and depth; S-RQ2–5 use controlled transition scenarios covering shared-factor shifts, selective reuse with expansion, missing and newly introduced factors, late-arriving fundamental factors, temporal drift, and recurrence (Appendix E.3). We report matched Mean Correlation Coefficient (MCC) for latent recovery, gate accuracy for reuse decisions, normalized mean squared error (NMSE) for reconstruction or prediction, and module counts for capacity. Forgetting is the decrease in old-factor MCC relative to its pre-adaptation reference. Ground-truth factors, identities, and ranks are used only for evaluation, never for selecting gates, update policies, or checkpoints.

S-RQ1: Can MoRe identify hierarchical components? Table 1 shows that linear recovery stays at 0.99 up to 24 latent dimensions and decreases to 0.93 and 0.85 at 64 and 128 dimensions, while nonlinear recovery ranges from 0.96 to 0.98. Figure 2 shows clear diagonal alignment between true and estimated latents. As the dashed blocks highlight, permutations occur only within layers (z_1, z_2 and z_3, z_4 are grouped), consistent with Theorem 1: cross-layer mixing is excluded while within-layer transformations remain.

S-RQ2: Does MoRe correctly decide what to reuse and what to add? With linear mixing, the gate achieves accuracy 1.0 on all of C1–C7, including the selective cases C4 and C5 that require simultaneous reuse and expansion (target MCC 0.99980 and 0.99972). On the larger suite C11–C17 (7 scenarios, 3 data seeds, 5 training seeds; 105 fits), gate accuracy remains 1.0, scenario-mean target MCC ranges from 0.99913 to 0.99997, and old-library recovery matches its pre-adaptation reference. Table 2 reports the nonlinear decoder-matching variant. Reuse remains accurate under shared mixing, including noisy observations. Independently rotating the target observations, which violates the shared-mixing assumption, reduces gate accuracy to 0.50 although target MCC stays at 0.97. Recovering latents and matching them across stages are therefore distinct: the latter requires compatible observation mappings, as argued in Section 4.1.

S-RQ3: Must newly arriving modules be more specific? No. In C8, the initial stage observes z_1 and z_3, and the next stage observes a previously unseen fundamental factor z_2. Across 15 fits, MoRe infers target rank one, adds the new module with gate accuracy 1.0 and MCC 0.999997 ± 0.000001, and retains the old representation. In C9, z_2 and z_3 are co-observed, so their link becomes estimable. A masked update learns the new-to-old link while freezing the old-to-old transition block, with standardized edge-magnitude error 0.109 ± 0.042 versus 0.217 ± 0.029 for a separate post-hoc fit

Table 4: AMPS continual learning (30 runs; NMSE; width = active dims).

Method	Old inc.↓	Target↓	Mean↓	Width
MoRe	.0000	.0341	.0303	7.97
Frozen	.0000	.0379	.0323	7.87
Fine-tuning	.0111	.0265	.0322	7.97
CaSSLe-style	.0069	.0280	.0308	7.97
EWC-style	.0043	.0315	.0312	7.97
Replay (500)	.0030	.0300	.0298	7.97
Naive exp.	.0000	.0226	.0229	15.83

Table 5: AMPS linear-probe accuracy (%); all layers in the appendix.

Repr.	Dim	n=10	n=50	n=100	n=1k
PCA	256	77.0	84.6	84.8	84.9
MoRe_all	256	79.3	84.3	84.1	85.4
MoRe_L2	64	**80.0**	**84.5**	84.6	85.7
MoRe_L3	64	79.3	83.1	83.8	**86.1**

(target MCC 0.998 ± 0.007). Stage order thus need not match causal order; links become estimable once their endpoints are jointly observed.

S-RQ4: Does reuse control library growth over long sequences? Across all 24 orders of four curricula sharing eight primitives (3 data seeds, 3 fitting seeds; 216 runs), every run ends with exactly eight modules, and the sequential-to-joint representation CKA is 0.9999994 ± 0.0000036. In a 32-stage stream with dormant and recurring factors, MoRe keeps eight modules with gate accuracy 1.0, whereas naive expansion accumulates 76. This compactness has a small fit cost (target NMSE 2.09×10^{-5} versus 6.54×10^{-7}).

S-RQ5: What does selective adaptation buy under limited capacity? We evaluate 900 four-stage trajectories (4 streams: temporal drift, observation-coordinate drift, new structure, dormant recurrence; 3 capacities; 5 policies; 3 data and 5 training seeds), with storage matched across policies and a shared update quota for selective policies. Under dormant recurrence at capacity nine (Table 3), hierarchy-guided adaptation raises new-data MCC from 0.559 (preservation) to 0.901, with less forgetting than random updates or full fine-tuning; at capacities twelve and fifteen, its MCC rises to 0.950 and 0.986 while forgetting falls to 0.004 and 0.000. Under temporal drift, dimension-only assessment is insufficient: preservation yields prediction NMSE 1.064, versus 0.292 for hierarchy-guided adaptation and 0.266 for full fine-tuning, with forgetting 0.000, 0.093, and 0.259. A structure-blind policy matches hierarchy-guided adaptation here. Selective adaptation is thus a useful capacity–retention tradeoff, but hierarchy-based selection is not consistently superior (full comparisons in the appendix).

5.2 Real-World LLM Activation Experiments

Setup. We use activations from arXiv[1], AMPS[2], and Wikipedia[3], extracted with Pythia-1.4B [3] and Gemma-2-9B [30], and fit MoRe with $d = 4$ layers on dataset-specific PCA coordinates (80/20 passage split). Each dataset provides a reference for abstraction level: arXiv subject categories (e.g., `math.LO`), AMPS difficulty tiers assigned by keyword matching (e.g., *add* $\rightarrow$ tier 0, *eigenvalue* $\rightarrow$ tier 3), and Wikipedia article length as a specificity proxy. These labels serve evaluation only and are not ground-truth hierarchies (Appendix E.2).

RW-RQ1: Does MoRe uncover category-selective, broad-to-specific structure? For each module, we measure how concentrated its activation is on a single category (CV-normalized) and compare it with a shuffled-label null (Figure 3). On all three datasets, every layer mean lies well above its null: by at least $25\times$ on AMPS and about 3–$6\times$ on arXiv and Wikipedia. On AMPS, whose labels encode an explicit curriculum, mean concentration decreases monotonically from L0 to L3 (0.054, 0.046, 0.041, 0.039), consistent with a broad-to-specific ordering. On arXiv and Wikipedia, the pattern is not monotonic, and L3 has the highest mean (0.050 and 0.036). MoRe modules are therefore category-selective on all datasets, whereas the broad-to-specific ordering appears only where labels encode an explicit curriculum; metadata and length proxies do not establish a semantic hierarchy. Cross-layer $\mathbf{B}$-link chains on AMPS trace curriculum-consistent transitions (Appendix E.4.1).

[1] `https://www.kaggle.com/datasets/Cornell-University/arxiv`
[2] `https://huggingface.co/datasets/sarahpann/AMPS`
[3] `https://huggingface.co/datasets/wikimedia/wikipedia`

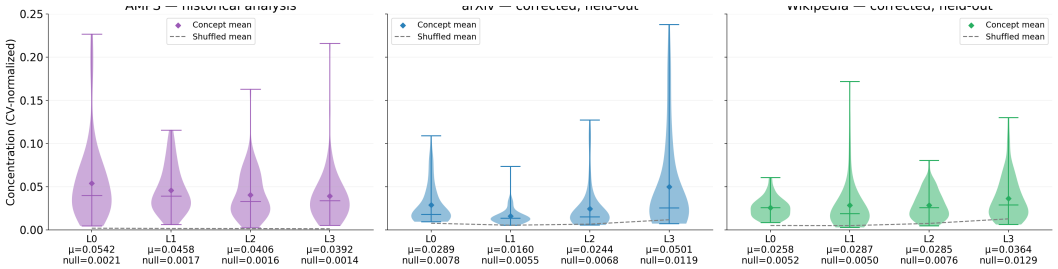

Figure 3: Per-layer category concentration of MoRe modules (Pythia-1.4B, 64 concepts per layer). Diamonds: concept means; dashed lines: shuffled-label nulls. arXiv and Wikipedia are evaluated on held-out passages with dataset-specific PCA; AMPS follows the original protocol.

RW-RQ2: Do MoRe representations improve label-efficient prediction? We train linear probes on frozen representations. On AMPS difficulty tiers (Table 5), every single-layer MoRe representation outperforms PCA at $n = 10$: MoRe_L2 reaches 80.0% versus 77.0% with 64 instead of 256 dimensions, and at $n = 50$ (84.5%) it matches PCA at $n = 100$ (84.8%); the advantage narrows as labels increase. On Wikipedia, MoRe achieves 54.96% versus 44.71% for PCA features at ten labels and 61.89% versus 57.94% at 100; on arXiv, MoRe reaches 48.11% and 63.77% at the same budgets.

RW-RQ3: Does MoRe learn continually without forgetting on real data? We evaluate six ordered AMPS transitions with five seeds (30 runs) against freezing, fine-tuning, distillation (CaSSLe-style [6]), regularization (EWC-style [12]), replay, and naive expansion; "-style" marks mechanism-level adaptations. We report the old-error increase $\max(0, E_{\text{old}} - E_{\text{old,ref}})$ and the mean $(E_{\text{old}} + E_{\text{target}})/2$ of post-adaptation reconstruction NMSE. As Table 4 shows, MoRe improves target fit over freezing with zero old-error increase and no replay; replay attains slightly lower mean error, and naive expansion fits better with about twice the active width. Across four AMPS curricula, final-library cross-order CKA is 0.984 ± 0.010.

Summary. MoRe recovers layer-structured latents, decides reuse and expansion correctly under shared mixing, and keeps the library compact; on LLM activations, it learns without forgetting or replay and yields category-selective, label-efficient modules.

6 Related Work

We review continual learning (CL) from a representation-level perspective. Existing methods have made substantial progress in reducing forgetting, but most decide what to preserve or update through task labels, replay buffers, architectural allocation, or optimization heuristics. MoRe instead seeks a data-grounded criterion: an identifiable modular representation that separates broadly reusable structure from more specific components. A more complete discussion is provided in Appendix F.

Supervised Continual Learning Supervised CL is commonly framed as learning a sequence of labeled tasks without catastrophic forgetting. Regularization and distillation methods, including Elastic Weight Consolidation, Learning without Forgetting, and Memory Aware Synapses, penalize updates that would harm previous tasks [12, 14, 1]; replay methods such as iCaRL preserve or approximate past-task information [23]. Architecture-based methods allocate or route capacity across tasks, for example through Progressive Networks, PathNet, Expert Gate, hard attention masks, or pruning-based reuse [24, 5, 2, 25, 19]. Recent variants use sparse experts, distribution-specialized MoE layers, adapters, LoRA subspaces, and prompt memories to reduce interference in large pretrained models [26, 4, 38, 8, 11, 34, 16, 36, 35, 27]. These approaches are effective, but their units of protection are usually parameters, experts, prompts, masks, or task-indexed modules. They therefore do not provide a principled criterion for distinguishing fundamental shared knowledge from task-specific specialization.

Unsupervised Continual Learning Unsupervised and self-supervised CL remove labels or task identities and evaluate whether representations remain transferable under sequential data. CURL introduced an unsupervised setting with task inference, expansion, and rehearsal [22]; streaming

SSL studies, LUMP, CaSSLe, PFR, and Kaizen show that replay, regularization, mixup, or temporal distillation can improve continual representation learning [10, 18, 6, 7, 29]. Branch-Tuning further analyzes stability and plasticity at the representation level and adds trainable branches before compressing them back into the network [17]. These works clarify that CL should be judged by representation quality, not only by task accuracy or training loss. However, their adaptation mechanisms still depend on replay, distillation, regularization, or branch modification, rather than on an identifiable decomposition of the representation itself.

Modular Network for Continual Learning Modularity is a natural strategy for the stability–plasticity trade-off [33], and recent work has unified adapters, MoE layers, sparse subnetworks, and prompt mixtures as modular deep learning mechanisms for CL [20]. Prior modular CL methods instantiate *architectural* modularity: modules are columns, paths, experts, masks, adapters, prompts, or sparse subnetworks selected by tasks, gates, priors, or heuristics [24, 5, 2, 25, 32, 21, 26]. MoRe is complementary but different. It learns *representational* modularity, where modules correspond to latent components organized by a hierarchy in the data-generating process. Time-delayed cross-layer dependencies make this hierarchy identifiable, providing a principled basis for reusing fundamental modules, aligning shared structure, and expanding only when new latent factors are needed, without requiring a task oracle.

7 Conclusion

We argued that the central challenge of continual learning is representational: incorporating new data with minimal interference requires a representation whose components reflect intrinsic structure in the data and can be selectively updated. We instantiated this view as MoRe, a framework that learns a hierarchical modular representation with identifiability guarantees from cross-layer time-delayed dependencies, paired with a two-phase adaptation procedure that interprets new data through existing modules and then selectively reuses, aligns, or expands them. Controlled synthetic experiments verify plasticity and stability across diverse transitions, and real-world LLM activations show that hierarchical structure emerges naturally and yields label-efficient representations.

Broader Impact. As foundational research, MoRe has no direct deployment path. Principled module reuse could lower the compute cost of adapting foundation models, broadening access; the interpretability of the learned hierarchy may also support auditing what a model has internalized. On the cautionary side, continually adapting systems can drift or entrench biases from skewed initial data—risks best mitigated by transparency and versioning of module libraries.

Acknowledgments and Disclosure of Funding

We would like to acknowledge the support from NSF Award No. 2229881, AI Institute for Societal Decision Making (AI-SDM), the National Institutes of Health (NIH) under Contract R01HL159805, grants from the MBZUAI-WIS Joint Program, and a gift from Aether AI.

References

[1] R. Aljundi, F. Babiloni, M. Elhoseiny, M. Rohrbach, and T. Tuytelaars. Memory aware synapses: Learning what (not) to forget. In *Proceedings of the European conference on computer vision (ECCV)*, pages 139–154, 2018.

[2] R. Aljundi, P. Chakravarty, and T. Tuytelaars. Expert gate: Lifelong learning with a network of experts. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3366–3375, 2017.

[3] S. Biderman, H. Schoelkopf, Q. G. Anthony, H. Bradley, K. O'Brien, E. Hallahan, M. A. Khan, S. Purohit, U. S. Prashanth, E. Raff, et al. Pythia: A suite for analyzing large language models across training and scaling. In *International conference on machine learning*, pages 2397–2430. PMLR, 2023.

[4] W. Chen, Y. Zhou, N. Du, Y. Huang, J. Laudon, Z. Chen, and C. Cui. Lifelong language pretraining with distribution-specialized experts. In *International Conference on Machine Learning*, pages 5383–5395. PMLR, 2023.

[5] C. Fernando, D. Banarse, C. Blundell, Y. Zwols, D. Ha, A. A. Rusu, A. Pritzel, and D. Wierstra. Pathnet: Evolution channels gradient descent in super neural networks. *arXiv preprint arXiv:1701.08734*, 2017.

[6] E. Fini, V. G. T. Da Costa, X. Alameda-Pineda, E. Ricci, K. Alahari, and J. Mairal. Self-supervised models are continual learners. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 9621–9630, 2022.

[7] A. Gomez-Villa, B. Twardowski, L. Yu, A. D. Bagdanov, and J. Van de Weijer. Continually learning self-supervised representations with projected functional regularization. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 3867–3877, 2022.

[8] N. Houlsby, A. Giurgiu, S. Jastrzebski, B. Morrone, Q. De Laroussilhe, A. Gesmundo, M. Attariyan, and S. Gelly. Parameter-efficient transfer learning for nlp. In *International conference on machine learning*, pages 2790–2799. PMLR, 2019.

[9] D. Hu, S. Yan, Q. Lu, L. Hong, H. Hu, Y. Zhang, Z. Li, X. Wang, and J. Feng. How well does self-supervised pre-training perform with streaming data? *arXiv preprint arXiv:2104.12081*, 2021.

[10] D. Hu, S. Yan, Q. Lu, L. Hong, H. Hu, Y. Zhang, Z. Li, X. Wang, and J. Feng. How well does self-supervised pre-training perform with streaming data? *arXiv preprint arXiv:2104.12081*, 2021.

[11] E. J. Hu, Y. Shen, P. Wallis, Z. Allen-Zhu, Y. Li, S. Wang, L. Wang, W. Chen, et al. Lora: Low-rank adaptation of large language models. *Iclr*, 1(2):3, 2022.

[12] J. Kirkpatrick, R. Pascanu, N. Rabinowitz, J. Veness, G. Desjardins, A. A. Rusu, K. Milan, J. Quan, T. Ramalho, A. Grabska-Barwinska, et al. Overcoming catastrophic forgetting in neural networks. *Proceedings of the national academy of sciences*, 114(13):3521–3526, 2017.

[13] H. Li, S. Lin, L. Duan, Y. Liang, and N. B. Shroff. Theory on mixture-of-experts in continual learning. *arXiv preprint arXiv:2406.16437*, 2024.

[14] Z. Li and D. Hoiem. Learning without forgetting. In B. Leibe, J. Matas, N. Sebe, and M. Welling, editors, *Computer Vision – ECCV 2016*, pages 614–629, Cham, 2016. Springer International Publishing.

[15] Z. Li, Y. Shen, K. Zheng, R. Cai, X. Song, M. Gong, G. Chen, and K. Zhang. On the identification of temporal causal representation with instantaneous dependence. In *The Thirteenth International Conference on Learning Representations*, 2025.

[16] Y.-S. Liang and W.-J. 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, 2024.

[17] W. Liu, F. Zhu, and C.-L. Liu. Branch-tuning: balancing stability and plasticity for continual self-supervised learning. *IEEE Transactions on Neural Networks and Learning Systems*, 2025.

[18] D. Madaan, J. Yoon, Y. Li, Y. Liu, and S. J. Hwang. Representational continuity for unsupervised continual learning. *arXiv preprint arXiv:2110.06976*, 2021.

[19] A. Mallya and S. Lazebnik. Packnet: Adding multiple tasks to a single network by iterative pruning. In *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, pages 7765–7773, 2018.

[20] J. Pfeiffer, S. Ruder, I. Vulić, and E. M. Ponti. Modular deep learning. *arXiv preprint arXiv:2302.11529*, 2023.

[21] E. M. Ponti, A. Sordoni, Y. Bengio, and S. Reddy. Combining parameter-efficient modules for task-level generalisation. In *Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics*, pages 687–702, 2023.

[22] D. Rao, F. Visin, A. Rusu, R. Pascanu, Y. W. Teh, and R. Hadsell. Continual unsupervised representation learning. *Advances in neural information processing systems*, 32, 2019.

[23] S.-A. Rebuffi, A. Kolesnikov, G. Sperl, and C. H. Lampert. icarl: Incremental classifier and representation learning. In *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, pages 2001–2010, 2017.

[24] A. A. Rusu, N. C. Rabinowitz, G. Desjardins, H. Soyer, J. Kirkpatrick, K. Kavukcuoglu, R. Pascanu, and R. Hadsell. Progressive neural networks. *arXiv preprint arXiv:1606.04671*, 2016.

[25] J. Serra, D. Suris, M. Miron, and A. Karatzoglou. Overcoming catastrophic forgetting with hard attention to the task. In *International conference on machine learning*, pages 4548–4557. PMLR, 2018.

[26] N. Shazeer, A. Mirhoseini, K. Maziarz, A. Davis, Q. Le, G. Hinton, and J. Dean. Outrageously large neural networks: The sparsely-gated mixture-of-experts layer. *arXiv preprint arXiv:1701.06538*, 2017.

[27] J. S. Smith, L. Karlinsky, V. Gutta, P. Cascante-Bonilla, D. Kim, A. Arbelle, R. Panda, R. Feris, and Z. 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, 2023.

[28] S. Tafazoli, F. M. Bouchacourt, A. Ardalan, N. T. Markov, M. Uchimura, M. G. Mattar, N. D. Daw, and T. J. Buschman. Building compositional tasks with shared neural subspaces. *Nature*, 650(8100):164–172, 2026.

[29] C. I. Tang, L. Qendro, D. Spathis, F. Kawsar, C. Mascolo, and A. Mathur. Kaizen: Practical self-supervised continual learning with continual fine-tuning. In *Proceedings of the IEEE/CVF winter conference on applications of computer vision*, pages 2841–2850, 2024.

[30] G. Team, T. Mesnard, C. Hardin, R. Dadashi, S. Bhupatiraju, S. Pathak, L. Sifre, M. Rivière, M. S. Kale, J. Love, et al. Gemma: Open models based on gemini research and technology. *arXiv preprint arXiv:2403.08295*, 2024.

[31] K. Tian, Z. Zhao, Y. Chen, N. Ge, S. Cao, X. Han, J. Gu, and S. Yu. Domain-specific schema reuse supports flexible learning to learn in the primate brain. *Nature Communications*, 2026.

[32] T. Veniat, L. Denoyer, and M. Ranzato. Efficient continual learning with modular networks and task-driven priors, 2021.

[33] L. Wang, X. Zhang, H. Su, and J. Zhu. A comprehensive survey of continual learning: Theory, method and application. *IEEE transactions on pattern analysis and machine intelligence*, 46(8):5362–5383, 2024.

[34] X. Wang, T. Chen, Q. Ge, H. Xia, R. Bao, R. Zheng, Q. Zhang, T. Gui, and X.-J. Huang. Orthogonal subspace learning for language model continual learning. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 10658–10671, 2023.

[35] Z. Wang, Z. Zhang, S. Ebrahimi, R. Sun, H. Zhang, C.-Y. Lee, X. Ren, G. Su, V. Perot, J. Dy, et al. Dualprompt: Complementary prompting for rehearsal-free continual learning. In *European conference on computer vision*, pages 631–648. Springer, 2022.

[36] Z. Wang, Z. Zhang, C.-Y. Lee, H. Zhang, R. Sun, X. Ren, G. Su, V. Perot, J. Dy, and T. Pfister. Learning to prompt for continual learning. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 139–149, 2022.

[37] W. Yao, G. Chen, and K. Zhang. Temporally disentangled representation learning. *Advances in Neural Information Processing Systems*, 35:26492–26503, 2022.

[38] J. Yu, Y. Zhuge, L. Zhang, P. Hu, D. Wang, H. Lu, and Y. He. Boosting continual learning of vision-language models via mixture-of-experts adapters. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 23219–23230, 2024.

[39] K. Zhang, S. Xie, I. Ng, and Y. Zheng. Causal representation learning from multiple distributions: A general setting. *arXiv preprint arXiv:2402.05052*, 2024.

Appendices for *"MoRe: Modular Representations for Continual Representation Learning on Sequential Data"*

Appendices Contents

A Definitions and Proofs **14**

A.1 Definitions 14

A.2 Proof for Theorem 1 15

A.3 Definitions and Proof of Theorem 2 17

A.4 Discussion of the Conditions 19

B Implementation Details **19**

B.1 Initial Estimation 19

B.2 Nonlinear Method 19

B.3 Nonlinear Preservation-First Learning 19

B.4 Layer-wise Estimation 20

C Adaptation Algorithm Details **22**

C.1 Self-Consistency Score 22

C.2 Coordinate Expansion 22

C.3 Hard Concrete Gates 23

C.4 Loss Function 23

C.5 Algorithm 23

D Limitations **24**

E Experimental Details **24**

E.1 Synthetic Experiments 24

E.1.1 Layer-wise Estimation Results 24

E.2 Real-world Experiments on LLM Activations 25

E.2.1 Preprocessing of LLM Activations 25

E.2.2 S-RQ2&3 Settings 27

E.3 Transition scenarios 27

E.4 Hyperparameter selection 28

E.4.1 Extracted Low-to-High Layer Relations 28

F Extended Related Work **28**

A Definitions and Proofs

A.1 Definitions

Definition 1 ((Informal) Intimate Neighbor Set [15],[39]). *The intimate neighbor set of $V_i \in \mathbf{V}$ contains all such $V_j \in \mathbf{V}$ that are adjacent to V_i in the Markov network $\mathcal{M}_{\mathbf{V}}$ and also adjacent to all other neighbors of V_i.*

A.2 Proof for Theorem 1

Theorem 1 (Identifiability of Latent Variables)**.** *Suppose the observed sequential data $\mathbf{X}_t$ follow the data-generating process with modular representations, and that the true and estimated models are observationally equivalent. We formulate the following conditions informally below; their mathematical treatment is given in Appendix A.2:*

- *C1 (Smooth and Positive Density) The latent transition density varies smoothly and remains positive throughout the latent domain.*
- *C2 (Sufficient Variability) Different past latent values produce sufficiently distinct changes in the current conditional distribution to rule out cross-layer mixing.*
- *C3 (Sparse Latent Process for Each Layer) Within-layer dependency patterns distinguish individual latents from their neighbors, either directly or through a child.*
- *C2-ind (Reduced Sufficient Variability) Without instantaneous dependencies, variation with the past need only distinguish individual latents.*

Under C1–C2, any recovered latent is a mixture of only the latents within the same layer in the true model. Further assuming C3 implies component-wise identifiability, i.e., each recovered latent is a function of only one true latent. Additionally, in the special case without instantaneous dependencies, C1 and C2-ind suffice for component-wise identifiability, which excludes cross-layer mixing, even though C2 may not hold.

Both the true and estimated models have d layers with m latents per layer. Their latent domains are $\mathbb{R}^{md}$, and their mixing functions are invertible onto their common observation image. The induced latent transformation is assumed to be a C^3 diffeomorphism. Both models satisfy the independent-noise and acyclic within-layer instantaneous structure specified in the generative model.

The conditions in the theorem are defined as follows.

C1 (Smooth and Positive Density). The conditional density $p(\mathbf{Z}_t \mid \mathbf{Z}_{t-1})$ is positive and three times continuously differentiable in the current and past latents on $\mathbb{R}^{md} \times \mathbb{R}^{md}$.

C2 (Sufficient Variability). Define

$$\mathbf{W}^a_{t,i} \triangleq \left(\frac{\partial^2 \log p(\mathbf{Z}_t \mid \mathbf{Z}_{t-1})}{\partial Z^a_{t,i} \partial Z^{l'}_{t-1,o}} \right)^{\mathsf{T}}_{l' \in [d],\, o \in [m]}, \tag{9}$$

$$\mathbf{U}^a_{t,i} \triangleq \left(\frac{\partial^3 \log p(\mathbf{Z}_t \mid \mathbf{Z}_{t-1})}{\partial (Z^a_{t,i})^2 \partial Z^{l'}_{t-1,o}} \right)^{\mathsf{T}}_{l' \in [d],\, o \in [m]}, \tag{10}$$

$$\mathbf{V}^a_{t,i,j} \triangleq \left(\frac{\partial^3 \log p(\mathbf{Z}_t \mid \mathbf{Z}_{t-1})}{\partial Z^a_{t,i} \partial Z^a_{t,j} \partial Z^{l'}_{t-1,o}} \right)^{\mathsf{T}}_{l' \in [d],\, o \in [m]}. \tag{11}$$

Here $i \in [m]$, $a \in [d]$, and $i < j$ for $\mathbf{V}^a_{t,i,j}$. For every fixed value of $\mathbf{Z}_t$, the collection

$$\{\mathbf{W}^a_{t,i}, \mathbf{U}^a_{t,i}\}_{a,i} \cup \{\mathbf{V}^a_{t,i,j}\}_{a,\, i<j}$$

is linearly independent as a collection of functions of $\mathbf{Z}_{t-1}$.

C3 (Sparse Latent Process for Each Layer). For any latent $Z^a_{t,i}$, either its intimate neighbor set is empty, or it has a direct child whose intimate neighbor set is empty. The intimate neighbor sets are defined in Definition 1 for the contextual network used by the applicable identification result.

The component-wise refinement invokes the corresponding contextual variability and estimation assumptions of [15, Theorem 2, arXiv version 4], including its minimum-edge estimation requirement. These additional prerequisites are used only for this refinement.

C2-ind **Reduced Sufficient Variability.** For every fixed value of $\mathbf{Z}_t$, the collection

$$\{\mathbf{W}^a_{t,i}, \mathbf{U}^a_{t,i}\}_{a,i}$$

is linearly independent as functions of $\mathbf{Z}_{t-1}$. This condition replaces C2 when both models have mutually independent current latents conditional on their respective past.

Proof. Assume the true model $(g, \{f_i\}_{i=1}^d, p(\boldsymbol{\epsilon}_t))$ and the estimated model $(\hat{g}, \{\hat{f}_i\}_{i=1}^d, p(\hat{\boldsymbol{\epsilon}}_t))$ induce the same distribution over the observations $\mathbf{X}_t$. Define the latent transformation

$$h \triangleq g^{-1} \circ \hat{g},$$

so that

$$\mathbf{Z}_t = h(\hat{\mathbf{Z}}_t).$$

Let

$$\mathbf{H}_t \triangleq \frac{\partial h(\hat{\mathbf{Z}}_t)}{\partial \hat{\mathbf{Z}}_t}$$

denote its Jacobian matrix.

Step 1: Conditional density transformation. Conditioning on $\mathbf{X}_{t-1}$, the change-of-variables formula gives

$$p(\hat{\mathbf{Z}}_t \mid \mathbf{X}_{t-1}) = p(\mathbf{Z}_t \mid \mathbf{X}_{t-1}) |\det(\mathbf{H}_t)|.$$

Since $\mathbf{X}_{t-1} = g(\mathbf{Z}_{t-1}) = \hat{g}(\hat{\mathbf{Z}}_{t-1})$ and both mixing functions are invertible, conditioning on $\mathbf{X}_{t-1}$ is equivalent to conditioning on either latent past. Therefore,

$$\log p(\hat{\mathbf{Z}}_t \mid \hat{\mathbf{Z}}_{t-1}) = \log p(\mathbf{Z}_t \mid \mathbf{Z}_{t-1}) + \log |\det(\mathbf{H}_t)|. \tag{12}$$

Step 2: Cross-layer conditional independence. By the modular representation assumption and independent exogenous noise,

$$p(\mathbf{Z}_t \mid \mathbf{Z}_{t-1}) = \prod_{a=1}^{d} p(\mathbf{Z}_t^a \mid \mathbf{Z}_{t-1}).$$

The same factorization holds across estimated layers. Thus, for $b \neq b'$ and any k, l,

$$\frac{\partial^2}{\partial \hat{Z}_{t,k}^b \partial \hat{Z}_{t,l}^{b'}} \log p(\hat{\mathbf{Z}}_t \mid \hat{\mathbf{Z}}_{t-1}) = 0.$$

Applying this to Eq. (12) gives

$$\frac{\partial^2 \log p(\mathbf{Z}_t \mid \mathbf{Z}_{t-1})}{\partial \hat{Z}_{t,k}^b \partial \hat{Z}_{t,l}^{b'}} + \frac{\partial^2 \log |\det(\mathbf{H}_t)|}{\partial \hat{Z}_{t,k}^b \partial \hat{Z}_{t,l}^{b'}} = 0. \tag{13}$$

Step 3: Chain rule expansion. Using the chain rule,

$$\begin{aligned}
&\frac{\partial^2 \log p(\mathbf{Z}_t \mid \mathbf{Z}_{t-1})}{\partial \hat{Z}_{t,k}^b \partial \hat{Z}_{t,l}^{b'}} \\
&= \sum_{i,a} \frac{\partial \log p(\mathbf{Z}_t \mid \mathbf{Z}_{t-1})}{\partial Z_{t,i}^a} \frac{\partial^2 Z_{t,i}^a}{\partial \hat{Z}_{t,k}^b \partial \hat{Z}_{t,l}^{b'}} \\
&\quad + \sum_{i,a} \sum_{j,a'} \frac{\partial^2 \log p(\mathbf{Z}_t \mid \mathbf{Z}_{t-1})}{\partial Z_{t,i}^a \partial Z_{t,j}^{a'}} \frac{\partial Z_{t,i}^a}{\partial \hat{Z}_{t,k}^b} \frac{\partial Z_{t,j}^{a'}}{\partial \hat{Z}_{t,l}^{b'}}.
\end{aligned} \tag{14}$$

The true-layer factorization makes the mixed derivatives zero when $a \neq a'$. For $a = a'$, separate the terms with $i = j$ and combine the terms with $i \neq j$ symmetrically over $i < j$.

Step 4: Differentiation with respect to lagged latents. Since $\mathbf{H}_t$ depends only on $\hat{\mathbf{Z}}_t$,

$$\frac{\partial}{\partial Z_{t-1,o}^{l'}} \log |\det(\mathbf{H}_t)| = 0.$$

Differentiate Eq. (13) with respect to all $Z^{l'}_{t-1,o}$ and stack the resulting equations. Using the definitions in C2 gives

$$\begin{aligned}\mathbf{0} = &\sum_{i,a} \frac{\partial^2 Z^a_{t,i}}{\partial \hat{Z}^b_{t,k} \partial \hat{Z}^{b'}_{t,l}} \mathbf{W}^a_{t,i} \\ &+ \sum_{i,a} \frac{\partial Z^a_{t,i}}{\partial \hat{Z}^b_{t,k}} \frac{\partial Z^a_{t,i}}{\partial \hat{Z}^{b'}_{t,l}} \mathbf{U}^a_{t,i} \\ &+ \sum_{a} \sum_{i<j} \left(\frac{\partial Z^a_{t,i}}{\partial \hat{Z}^b_{t,k}} \frac{\partial Z^a_{t,j}}{\partial \hat{Z}^{b'}_{t,l}} \right. \\ &\qquad \left. + \frac{\partial Z^a_{t,j}}{\partial \hat{Z}^b_{t,k}} \frac{\partial Z^a_{t,i}}{\partial \hat{Z}^{b'}_{t,l}} \right) \mathbf{V}^a_{t,i,j}. \end{aligned} \tag{15}$$

Step 5: Apply sufficient variability (C2). For fixed current latents, the coefficients in Eq. (15) do not depend on the past. By C2, all coefficients must vanish:

$$\frac{\partial^2 Z^a_{t,i}}{\partial \hat{Z}^b_{t,k} \partial \hat{Z}^{b'}_{t,l}} = 0, \tag{16}$$

$$\frac{\partial Z^a_{t,i}}{\partial \hat{Z}^b_{t,k}} \frac{\partial Z^a_{t,i}}{\partial \hat{Z}^{b'}_{t,l}} = 0, \tag{17}$$

$$\frac{\partial Z^a_{t,i}}{\partial \hat{Z}^b_{t,k}} \frac{\partial Z^a_{t,j}}{\partial \hat{Z}^{b'}_{t,l}} + \frac{\partial Z^a_{t,j}}{\partial \hat{Z}^b_{t,k}} \frac{\partial Z^a_{t,i}}{\partial \hat{Z}^{b'}_{t,l}} = 0, \quad i < j. \tag{18}$$

Step 6: Conclusion. Nonsingularity of $\mathbf{H}_t$ ensures that every row contains a nonzero entry. Equation (17) implies that the nonzero entries of each row lie within one estimated layer.

Moreover, two true latents in the same true layer cannot be associated with different estimated layers. Otherwise, selecting a nonzero derivative from each associated layer would make the first product in Eq. (18) nonzero and the second zero, a contradiction. Thus, all true latents in a layer are associated with the same estimated layer.

Since each layer contains m latents, two true layers cannot share one estimated layer: their $2m$ Jacobian rows would be supported on only m columns, contradicting nonsingularity. The resulting association is a permutation of layers. Continuity and nonsingularity make this association locally constant, and connectedness makes it globally constant. On the product latent domain, h therefore decomposes into invertible within-layer maps up to a layer permutation. Its inverse consequently excludes mixing between latents from different true layers.

For the component-wise refinement, apply the within-layer identification result under C3 and its additional contextual and estimation prerequisites. This yields component-wise identification within each recovered layer; this refinement is not used in the preceding cross-layer argument.

Finally, consider the special case without instantaneous dependencies in both models. All mixed-derivative functions $\mathbf{V}^a_{t,i,j}$ vanish. The estimated conditional independence now makes Step 2 valid for every pair of distinct estimated latents, including pairs within the same layer. Under C2-ind, Steps 4–5 therefore give

$$\frac{\partial Z^a_{t,i}}{\partial \hat{Z}^b_{t,k}} \frac{\partial Z^a_{t,i}}{\partial \hat{Z}^{b'}_{t,l}} = 0 \qquad \text{whenever } (b,k) \neq (b',l).$$

Each Jacobian row has exactly one nonzero entry, and nonsingularity gives a permutation of diagonal form. Hence the transformation is component-wise up to permutation, consistent with [37, Theorem 1]. This proves the special case without invoking C2 or the additional conditions for C3. □

A.3 Definitions and Proof of Theorem 2

A time-delayed dependency $u \to v$ means

$$Z_{t,v} \not\perp\!\!\!\perp Z_{t-1,u} \mid \mathbf{Z}_{t-1,-u},$$

where $\mathbf{Z}_{t-1,-u}$ contains all past latents except u.

Let $\ell(u)$ denote the true layer of u. In the general case, the block correspondence proved above defines $\widehat{\ell}(u) = \pi(\ell(u))$. In the conditionally independent case, $\widehat{\ell}(u)$ is the estimated layer containing $\sigma(u)$. Compatibility means

$$u \to v \Longrightarrow \widehat{\ell}(u) \leq \widehat{\ell}(v).$$

Conditions C4–C5 are:

- C4 Each estimated layer contains exactly m latents. A delayed effect from estimated layer a to b is permitted only if $a \leq b$. Instantaneous effects are permitted only within the same estimated layer.
- C5 For every pair of true layers $a < b$ and every pair of latents u, v with $\ell(u) = a$ and $\ell(v) = b$, the transition dependence $u \to v$ is present.

Proof. In the general case, the proof of Theorem 1 gives

$$\widehat{\mathbf{Z}}_t^{\pi(a)} = T_a(\mathbf{Z}_t^a),$$

with each T_a invertible. These transformations preserve conditional dependencies between entire layers.

If $u \to v$, with u in layer a and v in another layer b, then

$$\mathbf{Z}_t^b \not\perp\!\!\!\perp \mathbf{Z}_{t-1}^a \mid \mathbf{Z}_{t-1}^{-a}.$$

Otherwise, the target layer's conditional law would not depend on any past latent in layer a, contradicting $u \to v$. This dependence is preserved under the invertible block correspondence.

Suppose $\pi(a) > \pi(b)$. Resolving the acyclic instantaneous equations within estimated layer r gives

$$\widehat{\mathbf{Z}}_t^r = F_r\left(\widehat{\mathbf{Z}}_{t-1}^{\leq r}, \widehat{\boldsymbol{\epsilon}}_t^r\right).$$

Since the current exogenous noise is independent of the past, C4 excludes dependence on the past of any later estimated layer. The preserved dependence from $\pi(a)$ to $\pi(b)$ is therefore impossible, contradicting observational equivalence. Dependencies within a true layer have equal estimated layer indices. Thus compatibility holds in the general case.

In the conditionally independent case, the component-wise correspondence

$$\widehat{Z}_{t,\sigma(u)} = \phi_u(Z_{t,u})$$

preserves the conditional dependencies defining $u \to v$. Placing the matched source after the matched target again excludes a required dependence through C4. Compatibility therefore also holds in this case.

To establish uniqueness under C5, define

$$I_a = \{u : \ell(u) = a\}, \qquad J_a = \{u : \widehat{\ell}(u) = a\}.$$

In both cases, $|I_a| = |J_a| = m$. If $J_1 \neq I_1$, there exist $v \in J_1 \setminus I_1$ and $u \in I_1 \setminus J_1$. Condition C5 gives $u \to v$, but $\widehat{\ell}(u) > 1 = \widehat{\ell}(v)$, contradicting compatibility. Hence $J_1 = I_1$.

Assume $J_a = I_a$ for every $a < k$. If $J_k \neq I_k$, choose $v \in J_k \setminus I_k$ and $u \in I_k \setminus J_k$. The earlier layers have already been matched, so $\ell(v) > k$ and $\widehat{\ell}(u) > k$. Condition C5 again gives $u \to v$, whereas

$$\widehat{\ell}(u) > k = \widehat{\ell}(v),$$

contradicting compatibility. Therefore $J_k = I_k$. Induction establishes the entire ordered partition. □

A.4 Discussion of the Conditions

B Implementation Details

B.1 Initial Estimation

When $dm < n$, the exact likelihood follows from the rectangular change of variables, $\frac{1}{2}\log\det(\mathbf{J}_{\mathrm{Dec}_\phi}^\top \mathbf{J}_{\mathrm{Dec}_\phi})$, together with invertibility on the image achieved by reconstruction loss. For computational efficiency, we enforce invertibility with a reconstruction loss and omit the volume term.

B.2 Nonlinear Method

Constructing the library for nonlinear case. For nonlinear observations, the library instead retains a shared decoder Dec_ϕ, together with the encoder, normalization, latent calibration, and temporal-model parameters. A coordinate's observation effect is queried from this decoder:

$$\boldsymbol{\phi}_i(u) = \mathrm{Dec}_\phi(u\mathbf{e}_i) - \mathrm{Dec}_\phi(\mathbf{0}), \tag{19}$$

where $\mathbf{e}_i$ is the i-th coordinate unit vector. The response is computed when needed from the saved model; it is not a separately learned decoder or a stored historical sample. For a linear decoder, it reduces to $\boldsymbol{\phi}_i(u) = u(\mathbf{A}_0)_{:,i}$. For a nonlinear decoder, it provides a reference-context signature for matching and need not capture all interactions with other coordinates. The nonlinear estimation and calibration procedures are detailed in Appendix B.1.

Reuse assessment in nonlinear case. We fit a current-data encoder–decoder and select its width $\widehat{r}_s$ by validation reconstruction and prediction errors with a complexity penalty. Following temporal-profile calibration, we compare coordinate-specific responses of the saved and incoming decoders:

$$\boldsymbol{\phi}_i(u) = \mathrm{Dec}_{\mathrm{old}}(u\mathbf{e}_i) - \mathrm{Dec}_{\mathrm{old}}(\mathbf{0}), \tag{20}$$

$$\boldsymbol{\chi}_j^{(\alpha)}(u) = \mathrm{Dec}_s(\alpha u\mathbf{e}_j) - \mathrm{Dec}_s(\mathbf{0}). \tag{21}$$

These are queries of shared decoders, not independent generative functions whose outputs are summed. Let $\boldsymbol{\Phi}_i$ and $\mathbf{X}_j^{(\alpha)}$ stack the responses over a fixed query grid. Compatibility is measured by

$$S_{ij} = \max_{\alpha\in\mathcal{A}} \mathrm{clip}_{[0,1]} \frac{2\langle \boldsymbol{\Phi}_i, \mathbf{X}_j^{(\alpha)}\rangle}{\max\Big(\|\boldsymbol{\Phi}_i\|_2^2 + \|\mathbf{X}_j^{(\alpha)}\|_2^2, \epsilon\Big)}, \tag{22}$$

where $\mathcal{A}$ accounts for sign and scale ambiguity. We obtain a partial one-to-one matching by maximizing $\sum_{ij} P_{ij}(S_{ij} - \tau)$, subject to binary assignments and at most one match per row and column. Unmatched options allow modules to be rejected. Candidate matchings are fitted and selected using current validation fit and complexity penalties.

B.3 Nonlinear Preservation-First Learning

Let $\mathbf{a}$ contain matched coordinates expressed in old-coordinate units and $\mathbf{b}$ contain unmatched incoming coordinates. Let $\iota(\mathbf{a})$ embed matched coordinates into the old latent vector, setting inactive coordinates to zero. Let $\nu(\mathbf{a}, \mathbf{b})$ restore the incoming candidate's coordinate ordering and matched scales.

We retain the old decoder and define

$$\begin{aligned}\mathrm{Dec}_s^{\mathrm{reuse}}(\mathbf{a}, \mathbf{b}) = \mathrm{Dec}_{\mathrm{old}}(\iota(\mathbf{a})) + \mathrm{Dec}_{\mathrm{cand}}(\nu(\mathbf{a}, \mathbf{b})) \\ - \mathrm{Dec}_{\mathrm{cand}}(\nu(\mathbf{a}, \mathbf{0})).\end{aligned} \tag{23}$$

Therefore,

$$\mathrm{Dec}_s^{\mathrm{reuse}}(\mathbf{a}, \mathbf{0}) = \mathrm{Dec}_{\mathrm{old}}(\iota(\mathbf{a})). \tag{24}$$

The correction can model interactions between matched and new coordinates; it does not assume an additive decomposition into independent coordinate decoders. When no coordinate is matched, the incoming decoder is used directly.

The old model remains frozen. We optimize the target encoder, decoder correction, and temporal predictor using

$$\mathcal{L}_{\text{adapt}} = \mathcal{L}_{\text{rec}} + \lambda_{\text{pred}}\mathcal{L}_{\text{pred}} + \lambda_{\text{anchor}}\mathbb{E}_{\mathbf{X}\sim\mathcal{D}_s} \left\|\text{Enc}_s(\mathbf{X}) - \mathbf{z}_{\text{ref}}(\mathbf{X})\right\|_2^2, \tag{25}$$

where reconstruction and prediction are evaluated in observation space. The reference coordinates are produced by the fitted current-data candidate and expressed in the matched coordinate convention. They are neither ground-truth latents nor targets computed from historical observations. The target temporal predictor is learned rather than constrained to preserve every saved temporal mechanism.

Nonlinear Capacity-Constrained Adaptation The capacity-constrained extension stores a frozen initial backbone and coordinate-associated residual networks. Each allocated slot has an encoder residual mapping observations to a scalar, a decoder residual mapping the padded latent vector to observations, and a temporal residual producing a scalar prediction. The residual networks use one hidden layer with 32 tanh units and zero-initialized final layers.

Selecting a slot permits updates to its residual networks; unselected residual parameters remain fixed. Protected encoder coordinates depend only on the frozen backbone and their own frozen residuals, so their outputs are unchanged. Decoder and temporal outputs can nevertheless change indirectly because their latent inputs include other coordinates. Functional preservation is therefore evaluated separately from parameter preservation.

All allocated slots, including unused capacity, count toward persistent storage. The selective policies share a slot-update quota, whereas full fine-tuning additionally permits changes to the initial backbone. The comparison is storage-matched; full fine-tuning is not update-matched. Selection, fitting, and checkpoint decisions use current training and validation data, with old observations reserved for evaluation.

B.4 Layer-wise Estimation

We give the implementation details for the progressive estimator used when the joint objective in Eq. (3) is too expensive to optimize. Throughout this subsection, let $x_t := X_t^{\text{proj}}$ denote the projected observation used by the encoder. We index layers in the order in which they are estimated: layer 1 is the first, most fundamental module, and layer i is estimated after layers $1, \dots, i-1$ have been recovered. This notation matches the estimation order; if the generative-model section uses the reverse topological convention, the indices here should be interpreted after topological sorting of the layer-level time-delayed graph.

Principle. The layer-wise estimator follows directly from the hierarchical restriction in Eq. (2). Once $\hat{Z}_t^{<i} := (\hat{Z}_t^1, \dots, \hat{Z}_t^{i-1})$ has been learned, these earlier modules are treated as anchors. The next encoder f_i is trained so that $\hat{Z}_t^i = f_i(x_t)$ is (i) informative about the part of x_t not yet explained by previous modules, (ii) predictive from the allowed delayed parents $(\hat{Z}_{t-1}^i, \hat{Z}_{t-1}^{<i})$, and (iii) conditionally independent of the raw past observation once these allowed parents are known. The last condition prevents the new layer from hiding information that should have been captured by the already learned layers.

First-layer objective. For the first layer, define $\hat{Z}_t^1 = f_1(x_t)$. We use two conditional density models with matched architecture and noise prior,

$$q_1^0(\hat{z}_t^1 \mid \hat{z}_{t-1}^1), \qquad q_1^1(\hat{z}_t^1 \mid \hat{z}_{t-1}^1, x_{t-1}), \tag{26}$$

where q_1^0 is the admissible first-layer temporal model and q_1^1 is an augmented model that also sees the raw past. The conditional-independence requirement

$$\hat{Z}_t^1 \perp\!\!\!\perp X_{t-1} \mid \hat{Z}_{t-1}^1 \tag{27}$$

is enforced by the density-ratio estimate

$$\mathcal{L}_{\text{cmi}}^1 = \mathbb{E}_t\Big[\log q_1^1(\hat{Z}_t^1 \mid \hat{Z}_{t-1}^1, x_{t-1}) - \log q_1^0(\hat{Z}_t^1 \mid \hat{Z}_{t-1}^1)\Big]. \tag{28}$$

Minimizing Eq. (28) discourages any additional predictive information in x_{t-1} beyond $\hat{Z}^1_{t-1}$. To avoid collapse, we combine this term with an informativeness loss and a temporal prediction loss. In our implementation, the informativeness term is a reconstruction loss

$$\mathcal{L}^1_{\text{rec}} = \mathbb{E}_t \left[\|x_t - \tilde{g}_1(\hat{Z}^1_t)\|_2^2 \right], \tag{29}$$

and the prediction term is a JEPA-style loss with an exponential-moving-average target encoder $\bar{f}_1$,

$$\mathcal{L}^1_{\text{pred}} = \mathbb{E}_t \left[\|r_1(\hat{Z}^1_{t-1}) - \text{sg}(\bar{f}_1(x_t))\|_2^2 \right], \tag{30}$$

where r_1 is a latent predictor and $\text{sg}(\cdot)$ denotes stop-gradient. A variance floor

$$\mathcal{L}^1_{\text{var}} = \sum_k \max\{0, \rho - \text{Std}(\hat{Z}^1_{t,k})\} \tag{31}$$

is optionally used to rule out degenerate constant representations.

Conditional-density instantiation. The density models q^0_i and q^1_i can be implemented either as conditional normalizing flows or as Gaussian conditional density heads. For a flow implementation, with context c^0_t, we write

$$e_t = T_\phi(\hat{z}^i_t; c^0_t), \qquad \log q_\phi(\hat{z}^i_t \mid c^0_t) = \log p_E(e_t) + \log \left| \det \frac{\partial T_\phi(\hat{z}^i_t; c^0_t)}{\partial \hat{z}^i_t} \right|. \tag{32}$$

For the Gaussian-head implementation, $q_\phi(\hat{z} \mid c) = \mathcal{N}(\mu_\phi(c), \text{diag}\, \sigma^2_\phi(c))$. In both cases, q^0_i and q^1_i must share the same density family and noise prior; otherwise, the difference $\log q^1_i - \log q^0_i$ could reflect extra flexibility rather than additional information in the raw past.

Recursive estimation of later layers. After estimating layers $< i$, we freeze $f_1, \ldots, f_{i-1}$ and learn only f_i, its decoder head, its predictor, and its density models. Define the admissible context

$$c^0_{i,t} := (\hat{Z}^i_{t-1}, \hat{Z}^{<i}_{t-1}), \tag{33}$$

which contains the previous value of the new layer and the delayed values of all already recovered modules. The augmented context is

$$c^1_{i,t} := (\hat{Z}^i_{t-1}, \hat{Z}^{<i}_{t-1}, x_{t-1}). \tag{34}$$

The layer-i CMI penalty is then

$$\mathcal{L}^i_{\text{cmi}} = \mathbb{E}_t \Big[\log q^1_i(\hat{Z}^i_t \mid c^1_{i,t}) - \log q^0_i(\hat{Z}^i_t \mid c^0_{i,t}) \Big], \tag{35}$$

which estimates $I(\hat{Z}^i_t; X_{t-1} \mid \hat{Z}^i_{t-1}, \hat{Z}^{<i}_{t-1})$. The reconstruction term is applied to the full representation recovered so far,

$$\mathcal{L}^i_{\text{rec}} = \mathbb{E}_t \left[\|x_t - \tilde{g}_i(\hat{Z}^{\le i}_t)\|_2^2 \right], \qquad \hat{Z}^{\le i}_t = (\hat{Z}^{<i}_t, \hat{Z}^i_t), \tag{36}$$

with previous encoders frozen. This allows the decoder to change while preventing later layers from overwriting earlier modules. When desired, one can further residualize the input by replacing x_t with $x_t - \tilde{g}_{i-1}(\hat{Z}^{<i}_t)$, but our experiments use the full-observation decoder for stability.

The complete layer-i encoder objective is

$$\mathcal{L}_i = \lambda_{\text{rec}}\mathcal{L}^i_{\text{rec}} + \lambda_{\text{pred}}\mathcal{L}^i_{\text{pred}} + \lambda_{\text{cmi}}\mathcal{L}^i_{\text{cmi}} + \lambda_0\mathcal{L}^i_0 + \lambda_1\mathcal{L}^i_1 + \lambda_{\text{var}}\mathcal{L}^i_{\text{var}} + \lambda_{\text{orth}}\mathcal{L}^i_{\text{orth}}, \tag{37}$$

where

$$\mathcal{L}^i_0 = -\mathbb{E}_t \log q^0_i(\hat{Z}^i_t \mid c^0_{i,t}), \qquad \mathcal{L}^i_1 = -\mathbb{E}_t \log q^1_i(\hat{Z}^i_t \mid c^1_{i,t}). \tag{38}$$

The optional decorrelation term

$$\mathcal{L}^i_{\text{orth}} = \sum_{j<i} \left\| \text{Cov}(\hat{Z}^i_t, \hat{Z}^j_t) \right\|_F^2 \tag{39}$$

helps avoid relearning an already recovered module, but it is not required by the theory because the main hierarchy constraint is imposed through the delayed contexts in Eq. (35).

Optimization schedule. For each layer, we alternate two updates. First, with encoder outputs detached, we fit the density estimators by minimizing $\mathcal{L}_0^i + \mathcal{L}_1^i$. Second, with the density estimators fixed, we update f_i using Eq. (37). In practice, we warm up the encoder with $\mathcal{L}_{\text{pred}} + \mathcal{L}_{\text{rec}}$ before enabling the CMI penalty. This warm-up prevents early density-estimation noise from collapsing the representation. After warm-up, the prediction loss can be hinge-thresholded at the warm-up value so that the CMI term shapes the representation without sacrificing temporal predictability.

Layer-wise estimator. Algorithm 1 summarizes the recursive procedure.

Algorithm 1 Layer-wise estimation on the initial dataset

Require: Initial dataset $D_0 = \{x_t\}_{t=1}^T$, number of layers d, layer dimensions m_i, lag L, loss weights λ.

Fit the projection basis (V_0, μ_0) and set $x_t = (X_t - \mu_0)V_0$
Initialize the learned module list $\mathcal{M} \leftarrow \emptyset$
for $i = 1, \dots, d$ **do**
  Freeze all encoders in $\mathcal{M}$; compute $\hat{Z}_t^{<i}$ for all t
  Initialize f_i, decoder $\tilde{g}_i$, predictor r_i, target encoder $\bar{f}_i$, and density models (q_i^0, q_i^1)
  for training epochs **do**
    Form $\hat{Z}_t^i = f_i(x_t)$, admissible context $c_{i,t}^0$, and augmented context $c_{i,t}^1$
    Update (q_i^0, q_i^1) with $\hat{Z}_t^i$ detached by minimizing $\mathcal{L}_0^i + \mathcal{L}_1^i$
    Update $f_i, \tilde{g}_i, r_i$ by minimizing Eq. (37)
    Update the target encoder $\bar{f}_i \leftarrow \tau \bar{f}_i + (1-\tau) f_i$
  end for
  Add f_i to $\mathcal{M}$ and estimate the new transition block from $(\hat{Z}_{t-1}^{\le i}, \hat{Z}_t^{\le i})$
end for
return Encoder $f = (f_1, \dots, f_d)$, decoder $\tilde{g}_d$, transition estimate J_B, and projection basis (V_0, μ_0)

C Adaptation Algorithm Details

We provide the implementation details of the adaptation procedure introduced in Section C. Our experiments use a linear instantiation: f_{k-1} is parameterized by $\mathbf{F}_{k-1} \in \mathbb{R}^{d_{k-1} \times m_{k-1}}$ and the alignment T by $\mathbf{T} \in \mathbb{R}^{m_{k-1} \times m_{k-1}}$.

C.1 Self-Consistency Score

Given the inference-only forward pass $\mathbf{Z}_{t-1} = \mathbf{X}_{t-1}^{\text{proj}} \mathbf{F}_{k-1}$, $\mathbf{Z}_t = \mathbf{X}_t^{\text{proj}} \mathbf{F}_{k-1}$, and residuals $\mathbf{E}_t = \mathbf{Z}_t - \mathbf{Z}_{t-1} \mathbf{J}_B^{(k-1),\top}$, define the correlation matrices

$$\mathbf{H}_1[i,j] = \text{corr}(E_{i,t}, Z_{j,t-1}), \qquad \mathbf{H}_3[i,j] = \text{corr}(Z_{i,t-1}, Z_{j,t}). \tag{40}$$

The self-consistency score is

$$a_j \;=\; \big|\mathbf{H}_3[j,j]\big| \cdot \Big(1 - \tfrac{1}{m-1} \sum_{i \neq j} \big|\mathbf{H}_1[i,j]\big|\Big), \qquad a_j \in [0,1]. \tag{41}$$

Candidate new modules are assigned $a_j = 0$.

C.2 Coordinate Expansion

The residual $\mathbf{R} = \mathbf{X}_k - (\mathbf{X}^{\text{proj}} \mathbf{V}_{k-1}^\top + \boldsymbol{\mu}_{k-1})$ is centered, and its top k_{extra} right singular vectors are orthogonalized against $\mathbf{V}_{k-1}$ (Gram-Schmidt), then orthonormalized via thin QR, yielding $\mathbf{V}^{\text{new}} \in \mathbb{R}^{n \times k_{\text{extra}}}$ with $\mathbf{V}^{\text{new}\top} \mathbf{V}_{k-1} = \mathbf{0}$. The transition is extended as

$$\mathbf{J}_B^{\text{ext}} = \begin{bmatrix} \mathbf{J}_B^{(k-1)} & \mathbf{0} \\ \mathbf{J}_B^{\text{old}\to\text{new}} & \mathbf{J}_B^{\text{new}\to\text{new}} \end{bmatrix}, \tag{42}$$

with the zero block enforcing the layer-wise hierarchy mask of Theorem 2.

C.3 Hard Concrete Gates

Each gate $g_j \in [0,1]$ has logit ℓ_j and is sampled during training as

$$g_j = \mathrm{clip}_{[0,1]}\Big(\bar{s}\Big), \quad \bar{s} = s(\zeta-\gamma)+\gamma, \quad s = \sigma\big(\tfrac{\log u - \log(1-u)+\ell_j}{\tau}\big), \quad u \sim \mathrm{Uniform}(0,1), \tag{43}$$

with $\zeta = 1.1$, $\gamma = -0.1$, and τ annealed from τ_{start} to τ_{end}. At evaluation, $g_j = \mathbb{1}[\sigma(\ell_j) > 0.5]$. The expected L_0 regularizer is

$$\mathcal{L}_0(\mathbf{g}) = \sum_j \sigma\big(\ell_j - \tau \log(-\gamma/\zeta)\big). \tag{44}$$

Logits for existing modules are initialized from $\mathbf{a}$ as

$$\ell_j = C \cdot \big(2\,\mathrm{softmax}_T(\mathbf{a})_j - 1\big) \cdot \mathrm{clip}(\bar{a}/\eta, 0, 1), \quad \bar{a} = \tfrac{1}{m_{k-1}} \textstyle\sum_j a_j, \tag{45}$$

with hyperparameters C, T, η. Logits for candidate new modules are initialized at 0.

C.4 Loss Function

In the linear instantiation, the data loss is

$$\mathcal{L}_{\text{data}} = \|\mathbf{E}_t\|_1 - \tfrac{1}{2}\log\det\big(\mathbf{F}^{\text{active},\top}\mathbf{F}^{\text{active}} + \varepsilon\mathbf{I}\big), \tag{46}$$

where $\mathbf{F}^{\text{active}} = [\mathbf{F}_{k-1}\mathbf{T} \cdot \mathrm{diag}(\mathbf{g}^{\text{old}}),\ \mathbf{F}^{\text{new}}]$. The alignment regularizer is $\mathcal{R}(\mathbf{T}) = \|\mathbf{T}-\mathbf{I}\|_F^2$. The full objective is

$$\mathcal{L}_{\text{adapt}} = \mathcal{L}_{\text{data}} + \lambda_{\text{align}}\mathcal{R}(\mathbf{T}) + \lambda_{\text{sp}}^{\text{old}}\mathcal{L}_0(\mathbf{g}^{\text{old}}) + \lambda_{\text{sp}}^{\text{new}}\mathcal{L}_0(\mathbf{g}^{\text{new}}) + \lambda_J\big(\|\mathbf{J}_B^{\text{old}\to\text{new}}\|_1 + \|\mathbf{J}_B^{\text{new}\to\text{new}}\|_1\big). \tag{47}$$

C.5 Algorithm

Algorithm 2 MoRe Continual Adaptation at Stage k

Require: $\mathcal{D}_k$; previous $\mathbf{F}_{k-1}, \mathbf{V}_{k-1}, \boldsymbol{\mu}_{k-1}, \mathbf{J}_B^{(k-1)}$
Require: Hyperparameters $\{\lambda.\}, C, T, \eta$, warmup fraction r, epochs N, expansion budget k_{extra}
// Phase 1: Fitting and Assessment
$\mathbf{X}^{\text{proj}} \leftarrow (\mathbf{X}_k - \boldsymbol{\mu}_{k-1})\mathbf{V}_{k-1}$
Compute $\mathbf{Z}_{t-1}, \mathbf{Z}_t, \mathbf{E}_t, \mathbf{H}_1, \mathbf{H}_3$ ▷ inference only
$a_j \leftarrow |\mathbf{H}_3[j,j]| \cdot \big(1 - \frac{1}{m-1}\sum_{i\neq j}|\mathbf{H}_1[i,j]|\big)$
$\mathbf{V}^{\text{new}} \leftarrow$ orthogonalized top-k_{extra} singular vectors of residual $\mathbf{R}$
Allocate $\mathbf{F}^{\text{new}}, \mathbf{J}_B^{\text{old}\to\text{new}}, \mathbf{J}_B^{\text{new}\to\text{new}}, \mathbf{T} \leftarrow \mathbf{I}$
Initialize $\boldsymbol{\ell}^{\text{old}}$ from $\mathbf{a}$; $\boldsymbol{\ell}^{\text{new}} \leftarrow \mathbf{0}$
// Phase 2: Selective Updating
for epoch $= 1$ **to** N **do**
  Anneal τ; sample $\mathbf{g}^{\text{old}}, \mathbf{g}^{\text{new}}$ via Hard Concrete
  Compute $\mathcal{L}_{\text{adapt}}$; update $\mathbf{T}, \mathbf{F}^{\text{new}}, \mathbf{J}_B^{\text{old}\to\text{new}}, \mathbf{J}_B^{\text{new}\to\text{new}}$
  if epoch $> r \cdot N$ **then**
    Update gate logits $\boldsymbol{\ell}^{\text{old}}, \boldsymbol{\ell}^{\text{new}}$
  end if
end for
// Commit
Binarize: $g_j \leftarrow \mathbb{1}[\sigma(\ell_j) > 0.5]$
$\mathbf{V}_k \leftarrow [\mathbf{V}_{k-1}, \mathbf{V}^{\text{new}}[:, \mathbf{g}^{\text{new}} = 1]]$
$\mathbf{F}_k \leftarrow [\mathbf{F}_{k-1}\mathbf{T},\ \mathbf{F}^{\text{new}}[:, \mathbf{g}^{\text{new}} = 1]]$
Assemble $\mathbf{J}_B^{(k)}$ from $\mathbf{J}_B^{(k-1)}$ and selected new blocks
Store $\mathbf{g}^{(k)}$
Ensure: $(\mathbf{V}_k, \boldsymbol{\mu}_{k-1}, \mathbf{F}_k, \mathbf{J}_B^{(k)}, \mathbf{g}^{(k)})$

D Limitations

Our experiments are designed primarily as concept verification rather than benchmark competition. The synthetic settings, while controlled and informative for isolating mechanism, cover a limited range of transitions, dimensionalities, and noise structures; the real-world LLM activation experiments are similarly scoped to demonstrate the emergence of hierarchical structure rather than to claim downstream gains across diverse tasks. The implementation evaluated here uses a linear instantiation of the framework, which is principled but restrictive: scaling to nonlinear encoders, longer adaptation chains, and richer real-world distribution shifts is left to future work. Several theoretical assumptions—unidirectional cross-layer time-delayed structure, sufficient variability, and within-layer sparsity—may be only approximately satisfied in practice, and a systematic study of robustness to such violations remains open. Finally, our method addresses one adaptation step at a time; long-horizon stability across many sequential stages, and the design of memory mechanisms for reactivating long-dormant modules, are natural next steps toward a complete continual learning system.

E Experimental Details

E.1 Synthetic Experiments

E.1.1 Layer-wise Estimation Results

We focus especially on first-layer estimation because, in the layer-wise estimator, accurate recovery of the initial layer provides the anchor for estimating subsequent layers. Once an earlier layer has been recovered, the same estimation principle can be applied recursively to later layers by conditioning on the already learned lower-level structure. The experiments below therefore test whether the learned scalar representation preferentially aligns with ground-truth first-layer coordinates, and whether this behavior persists when the generator becomes nonlinear and when the hierarchy is extended to three layers.

We next visualize first-layer estimation on synthetic temporal data. In all panels, the horizontal axis is the learned scalar first-layer representation, while the vertical axis is a ground-truth latent coordinate from the synthetic generator. Because recovered latent coordinates are identifiable only up to the usual component-wise indeterminacies, and up to sign and scaling in the scalar linear diagnostics considered here, we report absolute correlations; a strong negative correlation is therefore as meaningful as a strong positive one. The main diagnostic question is whether the learned representation aligns more strongly with first-layer latents than with deeper-layer latents.

Figure 4 shows the two-layer linear non-Gaussian case. The learned representation exhibits a strong monotone relationship with the true first-layer latent and a substantially weaker relationship with the second-layer latent. Quantitatively, the final absolute correlation with the first-layer latent is about 0.90, versus about 0.41 for the second-layer latent. This is the expected qualitative pattern for successful first-layer recovery: the representation tracks the first layer closely while remaining relatively insensitive to the other latent coordinate.

Figure 5 reports the corresponding two-layer nonlinear setting. The same qualitative pattern persists, but the first-layer alignment becomes even stronger: the final absolute correlation with the first-layer latent is about 0.97, while the second-layer correlation remains around 0.42. This indicates that the method is not restricted to the linear case and can still isolate the first layer when the latent dynamics and/or observation mapping become nonlinear.

For the more challenging three-layer setting with two latent variables per layer, Figures 6 and 7 provide two complementary views. Figure 6 concentrates on the two first-layer coordinates. One coordinate is recovered very strongly ($|corr| \approx 0.94$), while the other is recovered more moderately ($|corr| \approx 0.49$). This is consistent with the current estimator being one-dimensional: it captures one dominant direction in the first-layer subspace, but it does not reconstruct the full two-dimensional first-layer subspace.

Figure 7 expands the same comparison to all six latent coordinates. The dominant first-layer coordinate still yields the largest absolute correlation, which is the key sign of first-layer recovery. At the same time, some non-first-layer coordinates remain substantially correlated, especially one coordinate in the third layer. We therefore interpret the three-layer result as a partial but meaningful

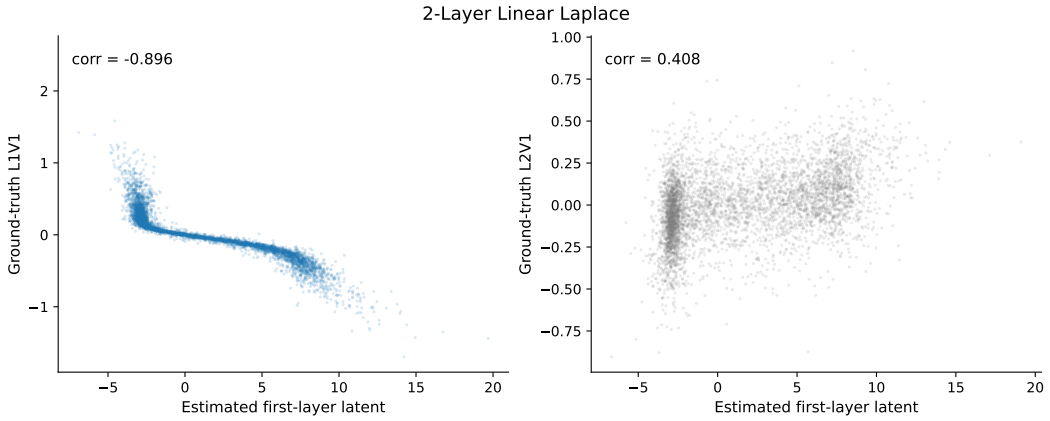

Figure 4: Two-layer linear non-Gaussian synthetic experiment. Each panel plots the learned scalar first-layer representation against one ground-truth latent coordinate. The left panel shows strong alignment with the first-layer latent, while the right panel shows a noticeably weaker relationship with the second-layer latent.

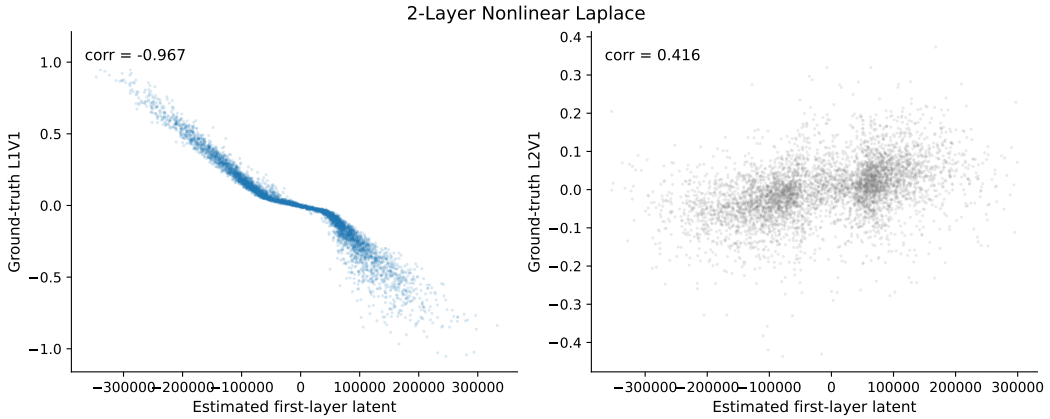

Figure 5: Two-layer nonlinear synthetic experiment. The learned representation remains strongly aligned with the first-layer latent and only weakly aligned with the second-layer latent, showing that the first-layer recovery pattern persists beyond the linear setting.

success: the estimator continues to prioritize the first layer, but the separation from the remaining latent coordinates is no longer as clean as in the simpler two-layer case.

Overall, these four figures support a cautious version of the intended conclusion. In the two-layer settings, the learned scalar representation aligns much more strongly with the first-layer latent than with the second-layer latent, under both linear Laplace and nonlinear Laplace data generation. In the harder three-layer setting, the largest absolute correlation is still attained by a first-layer coordinate, but one third-layer coordinate and one second-layer coordinate remain substantially correlated. Thus, the three-layer experiment provides evidence of preferential first-layer estimation in a deeper hierarchy, rather than full separation of all layers or full recovery of the two-dimensional first-layer subspace.

E.2 Real-world Experiments on LLM Activations

E.2.1 Preprocessing of LLM Activations

LLM Activations We extract hidden-state activations from the pretrained language model Pythia-1.4B, by registering forward hooks at a single intermediate layer (layer 16), chosen to capture rich semantic representations at approximately 60% of network depth. For each dataset, we stream up

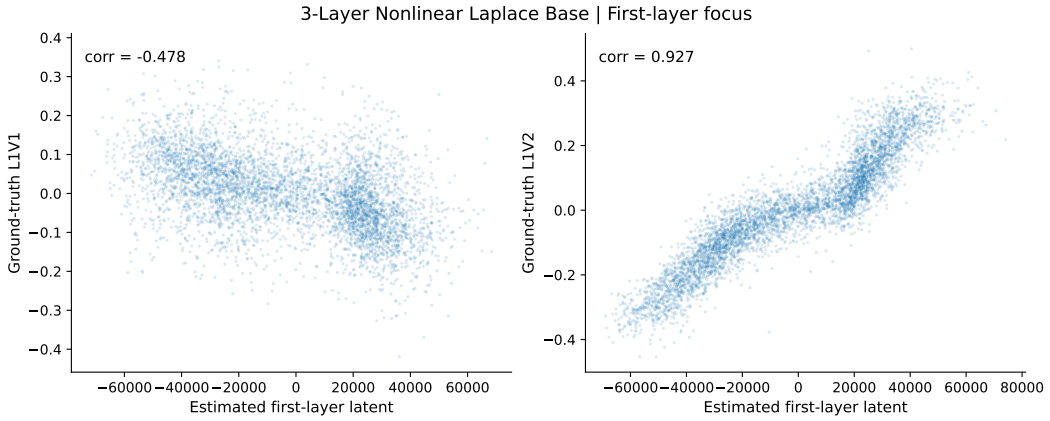

Figure 6: Three-layer synthetic experiment, first-layer focus. The learned scalar representation is plotted against the two first-layer ground-truth coordinates. The estimator strongly captures one dominant first-layer direction, while the second first-layer coordinate is represented more weakly.

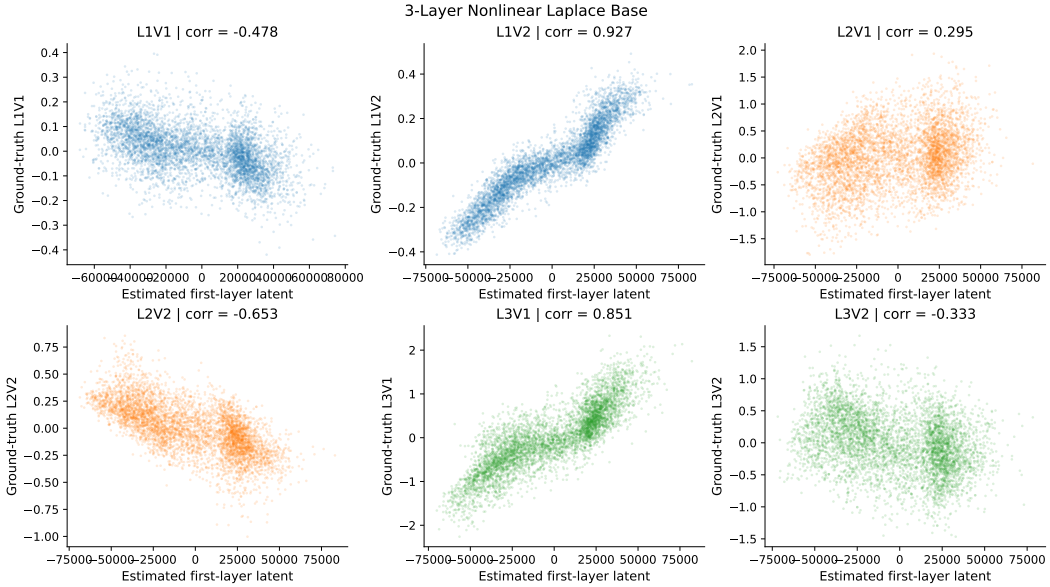

Figure 7: Three-layer synthetic experiment, full cross-layer comparison. Each panel compares the learned scalar first-layer representation with one ground-truth latent coordinate among the six coordinates spanning all three layers. The strongest alignment is attained on a first-layer coordinate, while several non-first-layer coordinates still exhibit sizable, though smaller, correlations.

to 50,000 passages (arXiv title+abstract concatenations, AMPS problem+solution pairs, Wikipedia articles), tokenize and truncate to 512 tokens, and run the frozen LLM to extract per-token hidden states, i.e., activations. The resulting token-level activation hidden dimension of Pythia-1.4B is 2048. They are compressed to 256 dimensions via Incremental PCA separately. Consecutive token windows of length $p + 1$ are constructed within passage boundaries to form training samples for MoRe, preserving local sequential structure without crossing document boundaries. We set the number of hierarchy layers to $L = 4$, providing sufficient depth to capture fundamental-to-specific concept gradients while remaining interpretable. For the VAR model order p, we evaluate $p \in \{10, 50, 100\}$: order 10 captures short-range token-level dependencies where hierarchical transitions are strongest (B-matrix weights 5–7$\times$ larger than at $p = 100$ on AMPS), order 50 captures mid-range discourse structure, and order 100 serves as an upper bound confirming that the signal degrades at longer lags — consistent with hierarchical concept transitions in structured text being primarily a local phenomenon. For the inter-layer sparsity penalty $\lambda_{\text{sp}} \in \{10^{-4}, 10^{-5}\}$, both values produce consistent

ID	Name	Active factors in data2	New dims
Data1: z_1, z_2			
C1	No shift	z_1, z_2	0
C2	Full expand	z_1, z_2, z_3, z_4	2
C3	Drop all old	z_3, z_4	2
C4*	Selective	z_1, z_3, z_4	2
C5*	Selective	z_2, z_3, z_4	2
C6	Keep one	z_1	0
C7	Keep one	z_2	0
Data1: z_1, z_3			
C8	New fundamental factor	z_2	1
C9*	New parent, retained child	z_2, z_3	1
Data1: $z_{1:5}$*; three layers of five factors*			
C11	No shift	$z_{1:5}$	0
C12	Full expand	$z_{1:15}$	10
C13	Drop all old	$z_{6:15}$	10
C14*	Selective	$z_{1:2}, z_{6:15}$	10
C15*	Selective	$z_{3:5}, z_{6:15}$	10
C16	Keep two	$z_{1:2}$	0
C17	Keep three	$z_{3:5}$	0

Table 6: Transition scenarios, grouped by the active factors in data1. We write $z_{a:b} = \{z_a, \ldots, z_b\}$. New dimensions count factors present in data2 but absent from data1. Starred cases require simultaneous reuse and expansion. C8 introduces a previously unseen fundamental factor; C9 additionally retains its child, allowing estimation of the dependency $z_2 \to z_3$. For C11–C17, the three layers are $z_{1:5}$, $z_{6:10}$, and $z_{11:15}$.

Gini concentration gradients on AMPS (L0 $\mu = 0.054$ vs. L2 $\mu \approx 0.039$ for both settings), confirming that hierarchy recovery is not sensitive to the precise sparsity level within this range. Unless otherwise stated, we report results at $p = 10$ and $\lambda_{\text{sp}} = 10^{-4}$, which yields the strongest B-matrix signal and clearest concentration gradient across all three datasets.

E.2.2 S-RQ2&3 Settings

E.3 Transition scenarios

All experiments use a shared data1 dataset generated from two active latent factors (z_1, z_2) observed through a random linear mixing matrix. The seven data2 transition scenarios vary which factors are retained, dropped, or newly introduced:

Evaluation metrics **Plasticity.** (target MCC): mean correlation coefficient between estimated and ground-truth latents on data2, maximized over permutations. Higher is better; scratch and fine-tune serve as upper-bound references.

Stability. (forgetting): difference between old-factor MCC before and after adaptation, $\Delta = \text{MCC}_{\text{old}}^{\text{before}} - \text{MCC}_{\text{old}}^{\text{after}}$. Negative values indicate backward transfer (adaptation improved old-factor recovery).

Gate accuracy (S-RQ3 only): fraction of binary gate decisions matching the ground-truth factor assignment, averaged over old and new modules.

Baselines

- **Scratch**: train a new encoder on data2 from random initialization; no knowledge transfer.
- **Fine-tune**: initialize from the data1 encoder and fine-tune all parameters on data2; no explicit stability mechanism.
- **Expansion only**: freeze the data1 encoder entirely and learn new modules for data2 factors; perfect stability by construction but cannot adapt reused modules.

- **MoRe w/o T**: full MoRe gating and sparsity losses, but the alignment matrix T is removed; tests the contribution of T to stability.
- **MoRe w/o $w_{\text{sp}}^{\text{old}}$**: MoRe without the sparsity penalty on old gates; tests whether old-module regularization is necessary for correct gate decisions.
- **MoRe w/o $w_{\text{sp}}^{\text{new}}$**: MoRe without the sparsity penalty on new gates; tests whether new-module regularization is necessary.
- **MoRe**: the full proposed method with uniform gate initialization, alignment T, and both sparsity penalties.

E.4 Hyperparameter selection

Hyperparameters are selected per scenario via random search over 8 configurations ($T = 50{,}000$ time steps, 3 seeds), evaluated by a composite score combining gate accuracy, target MCC, and forgetting. The selected ranges are: $w_{\text{sp}}^{\text{old}} \in \{0.005, 0.01, 0.05\}$, $w_{\text{sp}}^{\text{new}} \in \{0.001, 0.01, 0.05\}$, $w_{\text{align}} \in \{0.005, 0.02\}$, noise floor $\in \{0.3, 0.5\}$, $\tau_{\text{end}} \in \{0.2, 0.5\}$. All other hyperparameters are fixed: learning rate 5×10^{-3}, 2,000 training epochs, observation dimension 8, noise scale 0.1.

E.4.1 Extracted Low-to-High Layer Relations

Table 7: Representative cross-layer B-matrix links from the best setting (order $p = 10$, $\lambda_{\text{sp}} = 10^{-4}$, Pythia-1.4B). Each row shows a directed temporal dependency from a source-layer concept to a target-layer concept, annotated with the top-3 activating categories and the mean B-matrix weight (strength). AMPS links follow the curriculum direction (fundamental → specific); arXiv links show abstract mathematics predicting applied domains.

Dataset	Block	Source concept (top categories)	Target concept (top categories)	Strength
AMPS — curriculum hierarchy (arithmetic → advanced linear algebra)				
AMPS	L0 → L2	`L0_add`, `L0_simplify`, `L1_factor`	`L3_determinant`, `L2_matrix`, `L1_equation`	0.078
AMPS	L0 → L1	`L1_equation`, `L1_factor`, `L0_simplify`	`L0_multiply`, `L1_equation`, `L2_matrix`	0.063
AMPS	L1 → L3	`L1_factor`, `L0_simplify`, `L0_add`	`L3_determinant`, `L2_matrix`, `L0_multiply`	0.045
arXiv — abstract mathematics → applied domains				
arXiv	L1 → L3	`math.NT`, `math.LO`, `math.GM`	`math.CT`, `math.LO`, `physics.ins-det`	0.011
arXiv	L1 → L2	`math.CT`, `cs.LO`, `cs.SE`	`math.GN`, `q-fin.CP`, `math.GM`	0.010

Note: B-link strengths are 5–7× stronger on AMPS than arXiv (0.045–0.078 vs. 0.010–0.011), consistent with AMPS having cleaner sequential structure through its step-by-step curriculum format. arXiv strengths are weaker because domain category labels provide only a coarse proxy for abstraction level.

F Extended Related Work

We review the literature from a relatively high-level perspective to show that a principled way for continual representation learning is missing, and the key can be a modular representation structure.

Supervised Continual Learning Continual learning (CL) is commonly framed as the problem of acquiring new capabilities without catastrophic forgetting. In supervised CL, many representative methods address this problem by constraining parameter updates, replaying information from previous

tasks, or isolating parameters across tasks. Regularization and distillation methods—such as Elastic Weight Consolidation [12], Learning without Forgetting [14] and Memory Aware Synapses [1]—penalize changes that would harm previous tasks. Replay-based and exemplar-based methods, such as iCaRL [23], maintain or approximate past-task information while jointly learning classifiers and representations. Architecture-based methods instead allocate separate or partially separate computation to different tasks: Progressive Networks add task-specific columns with lateral transfer, PathNet learns task-specific paths through a shared module pool, and Expert Gate learns a gating mechanism that routes inputs to task-specialized experts [24, 5, 2]. A closely related line of work explicitly conditions computation on the task, such as Hard Attention to the Task [25] that learns nearly-binary attention masks that protect units used by previous tasks.

The Mixture-of-Experts (MoE) framework [26], in which a trainable gating network selects a sparse combination of feed-forward experts per example, provides a natural substrate for continual learning since new experts can be added without disturbing old ones. Recent analyses further formalize how expert specialization and routing reduce interference in continual task streams [13]. Lifelong-MoE [4] expands and freezes distribution-specialized experts and gating dimensions for continual language pre-training, while MoE-Adapters [38] attach task-specific adapter experts to a frozen vision-language backbone with a distribution-discriminative auto-selector. With large pre-trained models, parameter-efficient fine-tuning (PEFT) methods adapt only small task-specific components, such as adapters or low-rank updates, while freezing most backbone parameters [8, 11]. Building on this, recent CL methods constrain successive PEFT updates to interfere as little as possible with prior knowledge: O-LoRA [34] learns sequential LoRA updates in mutually orthogonal low-rank subspaces, and InfLoRA [16] reparameterizes pre-trained weights through an interference-eliminating subspace. Prompt-based continual learning extends the PEFT idea by learning small prompt memories or complementary prompts to manage task-specific and task-invariant knowledge without replay [36, 35, 27]. Despite their empirical strengths, these supervised CL methods predominantly treat forgetting as a parameter- or task-level phenomenon: experts, paths, prompts, and low-rank updates are defined and indexed by *tasks* rather than by the latent structure of the data, and consequently offer no principled way to separate fundamental, broadly shared knowledge from task-specific specializations. We argue that the underlying issue is fundamentally representational: tasks require representations that are simultaneously distinct (so updates do not interfere) and structured (so that what is shared is reused). Once such rich representations are obtained, a few labeled samples should suffice to specialize to any new task. MoRe takes this view explicitly, deriving its modular decomposition from identifiable factors of the data-generating process rather than from task labels.

Unsupervised Continual Learning A second line of work studies continual learning when labels or task identities are unavailable. CURL [22] introduced an unsupervised continual representation-learning setting in which the model infers tasks, dynamically expands to capture new concepts, and uses rehearsal to reduce forgetting. In the SSL setting, Hu et al. [10] evaluate sequential self-supervised pre-training on streaming ImageNet and DomainNet, using downstream transfer tasks to measure whether learned representations both adapt and avoid forgetting; they find that simple replay or parameter regularization can make sequential SSL close to joint pre-training in many settings, while severe distribution shifts remain challenging. LUMP [18] further observes that unsupervised representations forget less than supervised ones and proposes mixup interpolation between current and replayed instances. CaSSLe [6] converts SSL objectives into temporal distillation by mapping current representations to past ones, improving continual self-supervised visual representation learning across several SSL losses. PFR [7] and Kaizen [29] extend this predictor-based framework to mitigate plasticity loss and to jointly train feature extractors and classifiers. Most closely related to our motivation, Branch-Tuning [17] analyzes stability and plasticity using centered kernel alignment and proposes branch expansion/compression, fixing existing components while adding trainable convolutional branches for new data and then reparameterizing them back into the network. These methods clarify that continual SSL should be evaluated at the representation level, rather than only by immediate training losses. However, their adaptation mechanisms still operate by replay, distillation, regularization, or branch modification. MoRe instead asks what representation structure makes such selective adaptation principled: we posit and identify a hierarchical modular representation, enabling adaptation to be localized to the modules whose latent factors have changed.

Modular Network for Continual Learning Modularity is a long-standing strategy for managing the stability–plasticity trade-off [33]. Recent work has unified adapters, MoE layers, sparse subnet-

works, and prompt mixtures under a common modular framework that highlights CL as a canonical setting where modularity should help [20]. Progressive networks add a new column for each task and connect it laterally to previous columns [24]; PathNet evolves task-specific paths through a fixed pool of modules [5]; Expert Gate learns a network of experts and uses gating autoencoders to select the relevant expert at test time [2]; hard-attention and pruning approaches learn task-specific masks or reuse freed parameters to protect old tasks [25, 19]; and MNTDP [32] maintains a library of modules and decides per task which to reuse and which to add via a data-driven prior. Modular adapter approaches extend the same idea to parameter-efficient adaptation of pretrained transformers [21], and sparse mixture-of-experts architectures similarly route inputs to subsets of capacity [26]. These approaches instantiate *architectural* modularity: modules are usually experts, subnetworks, masks, prompts, or adapters that are introduced or selected according to tasks, heuristics, or learned gates. MoRe is complementary but different: it learns *representational* modularity, where modules correspond to latent components organized by a hierarchy in the data-generating process, with identifiability guarantees obtained from time-delayed cross-layer dependencies. The decomposition is therefore not merely a convenient allocation of parameters but a principled reflection of structure in the data, providing a basis for preserving fundamental knowledge while updating more specific components without requiring a task oracle.

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: The paper formalizes the modular representation, proves identifiability, and gives an adaptation algorithm with experiments, matching the abstract.

   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 included a limitation section in the Appendix.

   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: Assumptions C1–C5 are stated with each theorem; full proofs are in the appendix with sketches in the main text.

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: Data generation, model setup, and training details are in the experiments section and appendix, and we release anonymized code. `https://anonymous.4open.science/r/NeurIPS2026Submission18156/README.md`

   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: `https://anonymous.4open.science/r/NeurIPS2026Submission18156`

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 listed the details either in Appendix or in the released anonymous code.

   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: Most results are aggregated over multiple seeds with mean and std reported.

   Guidelines:

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

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

8. **Experiments compute resources**

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

   Answer: [Yes]

   Justification: All the experiments are conducted on Slurm cluster with Nvidia A6000 GPU.

   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: We follow the Code of Ethics; the paper uses synthetic data and public LLM activations, so with no human subjects.

   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: Foundational representation-learning work with no direct deployment path; briefly discussed in the broader impact section.

    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: No new pretrained models, generative models, or scraped datasets are released.

    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: Third-party libraries and the pretrained model used for activation extraction are cited and used under their licenses.

    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: The released code includes a README with setup and run instructions.

    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: No crowdsourcing or 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: No 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: LLMs are only a source of activation data for evaluation, not part of the method. Except for this, AI tools were used for copy-editing and presentation-level assistance.

    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.