LEARNING REPRESENTATIONS OVER TEMPORAL SIMPLICIAL COMPLEXES

Anonymous authors
Paper under double-blind review

ABSTRACT

Temporal higher-order networks can be used to describe time-evolving complex systems in various domains, including biological, social, and collaboration networks, where groups of entities interact over time. Learning representations for these networks requires jointly modeling temporal evolution and higher-order topology while efficiently processing a continuous stream of interaction. Existing approaches typically rely on either clique-expanded graphs or hypergraphs: graph-based models offer computational efficiency but often fail to capture higher-order dependencies, whereas hypergraph-based models represent such dependencies explicitly at the expense of increased computational complexity and limited theoretical expressivity. To address these challenges, we introduce the Temporal Simplicial Complex (TSC), a formalism for representing time-evolving higher-order interactions as an event-driven sequence of simplicial complexes. We define TSC isomorphism and develop the Temporal Simplicial Weisfeiler-Lehman (TSWL) test to characterize the expressive power of temporal simplicial representation learning. Building on this framework, we propose the Temporal Simplicial State Space Model ($\texttt{TS}^3\texttt{M}$), an online representation learning architecture that uses state space models to efficiently propagate information over the TSC. We establish an expressiveness bound for $\texttt{TS}^3\texttt{M}$ in terms of its state dimension, showing that its expressive power increases with the state dimension and reaches that of the TSWL test in the infinite-dimensional limit. Numerical experiments on dynamic higher-order interaction prediction across standard benchmark datasets show that $\texttt{TS}^3\texttt{M}$ achieves competitive predictive performance while delivering substantially faster training and inference compared to existing approaches.

1 INTRODUCTION

Temporal-relational data naturally arise in various real-world domains, including social networks, academic collaborations, online forum discussions, e-commerce, and biochemical interactions, where interactions inherently involve multiple entities and evolve continuously over time (Benson et al., 2018; Patania et al., 2017). Learning expressive representations from such temporal relational data has driven the development of Dynamic Graph Neural Networks (DGNNs) capable of capturing both structural and temporal dependencies (Yu et al., 2023; Wang et al., 2022). However, DGNNs fundamentally rely on pairwise edges and use clique expansion (CE) to represent group interactions as independent pairwise interactions. This process discards higher-order structural dependencies and consequently limits their modeling capacity (Behrouz et al., 2023; Yoon et al., 2020).

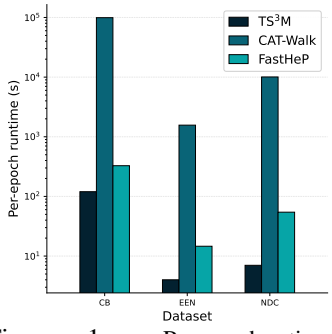


Figure 1: Per-epoch time(s) of training Hypergraph models (CAT-Walk, FastHep) and TSC Model (TS^3M) on different datasets.

To mitigate the limitations of CE graphs, prior works have adopted higher-order abstract structures such as hypergraphs and simplicial complexes (SCs) (Tian & Zafarani, 2024; Kim et al., 2024). While both frameworks generalize networks beyond pairwise relations, SCs additionally encode a hierarchy among higher-order interactions through closure under inclusion. This hierarchical structure with the closure property naturally induces a simple notion of neighborhood that supports

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

efficient computation, providing a principled algebraic foundation for higher-order message passing and theoretical expressiveness analysis. Despite advances in static simplicial learning, existing temporal higher-order methods predominantly focus on temporal hypergraphs (Behrouz et al., 2023; Xu et al., 2025; Peters et al., 2025). These methods either incur prohibitive computational costs that quadratically depend on hyperedge cardinality (CAT-Walk) (Behrouz et al., 2023) or employ flat aggregation schemes that disregard local higher-order topology (FastHep) Xu et al. (2025). This can be seen in Figure 1, wherein the hypergraph models incur large per-epoch training time in contrast to the proposed temporal simplicial learning method (discussed in Sec. 5). Moreover, current temporal higher-order architectures lack a unified theoretical framework for characterizing their expressive power.

To address this gap, we introduce the *Temporal Simplicial Complex* (TSC), a continuous-time, event-driven framework for modeling evolving higher-order relational systems. A TSC represents the evolution of higher-order interactions through insertion, deletion, and update events over simplices while preserving their intrinsic topological structure. Building upon this formulation, we develop the Temporal Simplicial Weisfeiler-Lehman (TSWL) test, establish a corresponding learnable Temporal Simplicial Network (TSN) architecture, and formulate an Embedding Reconstruction Problem (ERP), whose solution leads to the Temporal Simplicial State Space Model (TS^{3}M), an online TSN architecture. Our main contributions are summarized as follows

- We introduce TSCs, formalize temporal neighborhood operators, and define TSC isomorphism, providing foundations for representation learning over TSCs.
- We then introduce the TSWL test to characterize the expressive power of neural models for TSCs. We develop the corresponding TSN architecture and prove its invariance to TSWL equivalence.
- We formulate the ERP, a continuous-time representation learning objective for TSCs that reconstructs temporal simplex embedding trajectories through finite-dimensional latent states, and derive TS^{3}M, an online TSN architecture. We finally prove the dependence of the expressive power of TS^{3}M on its state dimension and attainment of TSWL distinguishing power in the infinite-dimensional limit.

In essence, these contributions provide a principled theoretical and algorithmic foundation for representation learning over TSCs. To evaluate our TS^{3}M framework, we conduct experiments on multiple real-world datasets for higher-order link prediction (HLP) in both transductive and inductive settings, as well as on dynamic simplicial closure prediction (DCP) tasks designed to assess the model's ability to capture higher-order interaction dynamics. These experiments empirically validate that TSC provides an effective and efficient framework for modeling higher-order interactions. Meanwhile, the results on DCP indicate that CE-based graph models degrade in performance as the event stream becomes increasingly dominated by higher-order events, whereas the TSC-based TS^{3}M performs consistently across datasets. We further examine the effects of each component of TS^{3}M on predictive performance. Our results on HLP demonstrate that TS^{3}M performs competitively against existing baselines, validating the utility of the TSC framework and the ERP formulation for capturing temporal higher-order dependencies. All proofs, ablation studies, and related work are provided in the Appendix[1].

2 Temporal Simplicial Complex

We recall the formal definition of abstract simplicial complexes (SCs) and their boundary and co-boundary relations, then extend them to temporal settings through temporal SCs (TSCs) and their corresponding relations to model the evolution of higher-order interactions in dynamic networks.

Simplicial complex. Let $\mathbb{V} := \{u_0, u_1, \ldots, u_{N-1}\}$ denote a finite set of nodes. Given $\mathbb{V}$, a k-simplex $\sigma_i^k := u_{i_0}, u_{i_1}, \ldots, u_{i_k}$ is a subset of $\mathbb{V}$ with cardinality $k+1$ (Figure 2(a) illustrates instances of a 1, 2, and 3-simplex). A SC $\mathcal{S}$, of order K is a nonempty set of k-simplices for $k = 0, 1, \ldots, K$ that is closed under inclusion, i.e., if $\sigma^k \in \mathcal{S}$, then $\gamma^{k'} \in \mathcal{S}$ for every non-empty $\gamma^{k'} \subseteq \sigma^k$. Here, the order is the maximum simplex dimension, $K := \max_{\sigma \in \mathcal{S}}(|\sigma| - 1)$.

An SC specifies neighborhood relations between simplices through their boundary and co-boundary simplices. The set of boundary simplices of a k-simplex $\sigma^k \in \mathcal{S}$ is defined as $\mathbb{B}(\sigma^k) := \{\gamma^{k-1} \in \mathcal{S} \mid$

[1] code: Anonymous GitHub

$\gamma^{k-1} \subset \sigma^k\}$ for $k > 0$, with $\mathbb{B}(\sigma^0) = \emptyset$. This boundary incidence relation is denoted by $\gamma^{k-1} \prec \sigma^k$. Similarly, the set of co-boundary simplices of σ^k is defined as $\mathbb{C}(\sigma^k) := \{\gamma^{k+1} \in \mathcal{S} \mid \sigma^k \subset \gamma^{k+1}\}$. Based on these relations, the lower and upper neighborhoods of σ^k are, respectively, given by

$$\mathbb{N}_{\downarrow}(\sigma^k) := \{\gamma^k \in \mathcal{S} \mid \mathbb{B}(\gamma^k) \cap \mathbb{B}(\sigma^k) \neq \emptyset\}, \text{ and } \mathbb{N}_{\uparrow}(\sigma^k) := \{\gamma^k \in \mathcal{S} \mid \mathbb{C}(\gamma^k) \cap \mathbb{C}(\sigma^k) \neq \emptyset\},$$

respectively (Barbarossa & Sardellitti, 2020). Thus, lower neighbors share a boundary simplex, whereas upper neighbors share a co-boundary simplex.

Temporal simplex and event. Let $\mathbb{T} = [t_1, t_2, \ldots, t_n]$ be a finite, non-decreasing sequence of event timestamps, with $t_j \in \mathbb{R}$ for $j = 1, \ldots, n$. Repeated timestamps allow multiple events to occur simultaneously. For each $t \in \mathbb{T}$, let $\mathbb{V}_t \subseteq \mathbb{V}$ denote the set of nodes observed up to and including time t. We define a temporal k-simplex at time $t \in \mathbb{T}$ as a tuple $\sigma_i^k(t) := (\sigma_i^k, t)$, where $\sigma_i^k = \{u_{i_0}, u_{i_1}, \ldots, u_{i_k}\} \subseteq \mathbb{V}_t$ is a subset of $k+1$ distinct observed nodes representing a higher-order interaction. We define the j-th event as $e_j := (\sigma_{i_j}^{k_j}(t_j), \boldsymbol{x}_j, \alpha_j)$, where $\boldsymbol{x}_j \in \mathbb{R}^{d_{k_j}}$ denotes the associated feature vector and $\alpha_j \in \mathbb{A} = \{\texttt{insertion}, \texttt{deletion}, \texttt{update}\}$ specifies the event type. The event acts on the underlying simplex $\sigma_{i_j}^{k_j}$ at time t_j. Given $\mathbb{T}$, the corresponding *event stream* is the ordered sequence of events $\mathcal{E}(\mathbb{T}) = (e_1, e_2, \ldots, e_n)$. Events with identical timestamps retain their order in the stream.

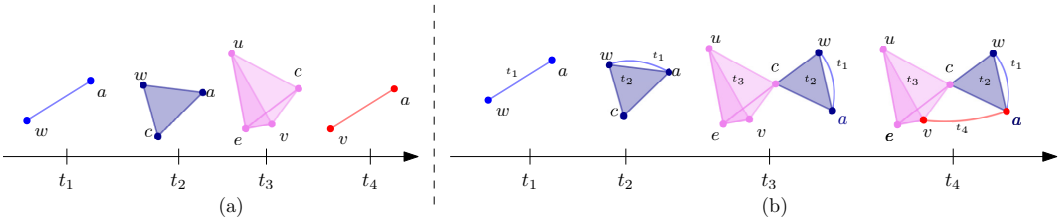


Figure 2: Illustration of an event stream with a 1-simplex at t_1, a 2-simplex at t_2, and a 3-simplex at t_3 in (a), along with the corresponding TSC in (b), where timestamped simplices indicate the occurrence of the event.

Temporal simplicial complex. A TSC $\mathcal{D}(t)$ up to time $t \in \mathbb{T}$ is defined as set of temporal simplices that is closed under inclusion at each timestamp; i.e., if $\sigma^k(t_j) \in \mathcal{D}(t)$, then $\gamma(t_j) \in \mathcal{D}(t)$ for all nonempty subsets $\gamma \subseteq \sigma^k$. Thus, an occurrence of a simplex entails occurrences of all its nonempty faces at the same timestamp. The TSC $\mathcal{D}(t)$ evolves through an event stream $\mathcal{E}(\mathbb{T})$ via a transition operator Φ such that $\mathcal{D}(t_i) = \Phi(\mathcal{D}(t_{i-1}), e(t_i))$ for $e(t_i) \in \mathcal{E}(\mathbb{T})$, where Φ preserves closure under inclusion and uses event type to modify $\mathcal{D}$. Here, t_{i-1} and t_i are consecutive event times, and $e(t_i)$ denotes the event or batch of simultaneous events at t_i. At time t, the set of unique k-simplices in $\mathcal{D}(t)$ is denoted by $\mathbb{S}_k(t) := \{\sigma^k \mid \exists\, t_j \leq t, :\ \sigma^k(t_j) \in \mathcal{D}(t)\}$, and $\mathbb{S}(t) := \bigcup_{k \geq 0} \mathbb{S}_k(t)$ represents the collection of all unique simplices in $\mathcal{D}(t)$. For each $\sigma \in \mathbb{S}(t)$, let $\boldsymbol{x}(\sigma, t) \in \mathbb{R}^f$ denote the feature vector associated with σ at time t. Under a fixed ordering of $\mathbb{S}(t)$, the feature matrix for $\mathcal{D}(t)$ is defined by stacking these vectors as rows of $\boldsymbol{X}(t) = [\boldsymbol{x}(\sigma, t)^\top]_{\sigma \in \mathbb{S}(t)} \in \mathbb{R}^{|\mathbb{S}(t)| \times f}$.

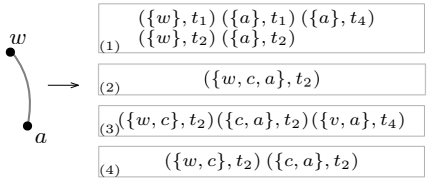


Figure 3: Neighbor of simplex $\{w, a\}$ in Figure 2 at time t_4, 1) boundary, 2) co-boundary 3) lower-adjacent and 4) upper adjacent.

Temporal neighborhoods. We use temporal boundary and co-boundary relations to define upper and lower temporal adjacencies in a TSC. For $\sigma^k \in \mathbb{S}_k(t)$, these are defined as

$$\begin{aligned}
\mathbb{B}(\sigma^k, t) &:= \left\{\gamma^{k-1}(t_i) \in \mathcal{D}(t) \mid \ \gamma^{k-1} \subset \sigma^k,\ \exists\, t_j \leq t_i :\ \sigma^k(t_j) \in \mathcal{D}(t)\ \right\}, \quad k > 0, \\
\mathbb{C}(\sigma^k, t) &:= \left\{\gamma^{k+1}(t_i) \mid \sigma^k \subset \gamma^{k+1} \text{ and } \gamma^{k+1}(t_i) \in \mathcal{D}(t)\right\}, \\
\mathbb{N}_{\uparrow}(\sigma^k, t) &:= \left\{(\gamma^k, t_m) \mid \ \gamma^k \in \mathbb{S}_k(t), \exists\, \eta^{k+1}(t_m) \in \mathbb{C}(\gamma^k, t) \cap \mathbb{C}(\sigma^k, t)\ \right\}, \\
\mathbb{N}_{\downarrow}(\sigma^k, t) &:= \left\{(\gamma^k, t_m) \mid \ \gamma^k \in \mathbb{S}_k(t),\ \exists\, \eta^{k-1}(t_m) \in \mathbb{B}(\gamma^k, t) \cap \mathbb{B}(\sigma^k, t)\ \right\}, \quad k > 0. \qquad (1)
\end{aligned}$$

For $k = 0$, we set $\mathbb{B}(\sigma^0, t) = \emptyset$ and $\mathbb{N}_{\downarrow}(\sigma^0, t) = \emptyset$. In the upper and lower temporal neighborhoods, a pair (γ^k, t_m) identifies a neighboring simplex together with the timestamp of the co-boundary

or boundary occurrence that induces the adjacency. Such a pair does not necessarily represent an occurrence $\gamma^k(t_m) \in \mathcal{D}(t)$. Consequently, the same simplex may appear multiple times in a temporal neighborhood, paired with distinct timestamps Figure 3 shows the neighbors of simplex $\{w, a\}$ at t_4 in the TSC shown in Figure 2(b). These temporal neighborhoods capture both structural and temporal context, motivating the problem of learning simplex representations in a TSC.

Problem Statement. Given a TSC $\mathcal{D}(\tau)$ at time $\tau \in \mathbb{T}$, generated by the events in $\mathcal{E}(\mathbb{T})$ up to time τ, and an associated feature matrix $\boldsymbol{X}(\tau)$, the goal is to learn a mapping

$$\Psi(\,\cdot\,; \boldsymbol{X}(\tau), \mathcal{D}(\tau)) : \mathbb{S}(\tau) \to \mathbb{R}^d$$

such that the representation $\Psi(\sigma; \boldsymbol{X}(\tau), \mathcal{D}(\tau))$ captures both the structural and temporal context of each simplex $\sigma \in \mathbb{S}(\tau)$.

3 Temporal Simplicial Weisfeiler–Lehman (TSWL) Test

We introduce the Temporal Simplicial Weisfeiler–Lehman (TSWL) test to characterize the expressive power of temporal simplicial representations. We first define temporal simplicial complex (TSC) isomorphism and then extend the SWL refinement procedure (Bodnar et al., 2021) to temporal settings.

Definition 1 (TSC Isomorphism)**.** *Given node sets $\mathbb{V}, \mathbb{V}'$, feature histories $\{\boldsymbol{X}(t_j)\}_{t_j \le t}$ and $\{\boldsymbol{X}'(t_j)\}_{t_j \le t}$, and a common set of observed timestamps $\mathbb{T}$, two TSCs $\mathcal{D}(t)$ and $\mathcal{D}'(t)$ are* isomorphic*, i.e., $\mathcal{D}(t) \approx \mathcal{D}'(t)$, if and only if there exists a bijection $f : \mathbb{V}' \to \mathbb{V}$ whose induced maps on simplices and temporal simplices, defined by $f(\sigma') := \{f(u') \mid u' \in \sigma'\}$ and $f(\sigma'(t_j)) := (f(\sigma'), t_j)$, satisfy the following conditions for every $t_j \in \mathbb{T}$ with $t_j \le t$:*

a) $\boldsymbol{x}(f(\sigma'), t_j) = \boldsymbol{x}'(\sigma', t_j),\ \forall\, \sigma' \in \mathbb{S}'(t_j)$, *b)* $\sigma'(t_j) \in \mathcal{D}'(t) \Longleftrightarrow f(\sigma'(t_j)) \in \mathcal{D}(t),\ \forall\, \emptyset \neq \sigma' \subseteq \mathbb{V}'$.

Thus, simplex representation matrix should transform equivariantly under the induced permutation of its rows for TSC isomorphism. While isomorphism specifies the required consistency under relabeling, it does not by itself characterize the discriminative power of a representation. To formalize expressiveness, we introduce TSWL, a color refinement procedure for TSCs. TSWL iteratively refines simplex colors by aggregating information from their temporal neighborhoods [cf. Equation 1] incorporating adjacency and evaluating colorings at each observed timestamp.

Initial coloring. Given an observation horizon τ, let $\mathbb{T}_\tau = \{t_j \in \mathbb{T} \mid t_j \le \tau\}$. For each $t_j \in \mathbb{T}_\tau$ and $\sigma \in \mathbb{S}(t_j)$, define $c_\sigma^{(0)}(t_j) := c^{(0)}(\boldsymbol{x}(\sigma, t_j))$, where $c^{(0)} : \mathcal{X} \to \mathcal{C}$ is injective. $\mathcal{X}$ is the simplex feature space, and $\mathcal{C}$ is the color space.

Color refinement. Let $c_\sigma^{(\ell)}(t_i)$ denote the color assigned to simplex $\sigma \in \mathbb{S}(t_i)$ after the ℓ-th refinement iteration at timestamp t_i. TSWL updates this color as

$$c_\sigma^{(\ell+1)}(t_i) = \mathrm{HASH}\Big(c_\sigma^{(\ell)}(t_i), c_{\sigma,\uparrow}^{(\ell)}(t_i), c_{\sigma,\downarrow}^{(\ell)}(t_i), c_{\sigma,\mathbb{B}}^{(\ell)}(t_i), c_{\sigma,\mathbb{C}}^{(\ell)}(t_i)\Big)\,, \tag{2}$$

$$c_{\sigma,\downarrow}^{(\ell)}(t_i) = \{\!\{(c_\gamma^{(\ell)}(t_i), c_{\gamma\cap\sigma}^{(\ell)}(t_i), t_i - t_j) \mid (\gamma, t_j) \in \mathbb{N}_\downarrow(\sigma, t_i)\}\!\}, \tag{3}$$

$$c_{\sigma,\uparrow}^{(\ell)}(t_i) = \{\!\{(c_\gamma^{(\ell)}(t_i), c_{\gamma\cup\sigma}^{(\ell)}(t_i), t_i - t_j) \mid (\gamma, t_j) \in \mathbb{N}_\uparrow(\sigma, t_i)\}\!\}, \tag{4}$$

$$c_{\sigma,\star}^{(\ell)}(t_i) = \{\!\{(c_\gamma^{(\ell)}(t_i), t_i - t_j) \mid \gamma(t_j) \in \star(\sigma, t_i)\}\!\}, \quad \star \in \{\mathbb{B}, \mathbb{C}\}, \tag{5}$$

where $\gamma \cap \sigma$ and $\gamma \cup \sigma$ are the shared boundary and co-boundary simplices and $\mathrm{HASH}(\cdot)$ is an injective mapping over multiset $\{\!\{\cdot\}\!\}$. Refinement is performed on the finite set of simplex–time pairs $\Omega_\tau := \{(\sigma, t_j) \mid t_j \in \mathbb{T}_\tau,\ \sigma \in \mathbb{S}(t_j)\}$ until the induced partition stabilizes. The partition is stable after L iterations if, for every $(\sigma, t_j), (\gamma, t_m) \in \Omega_\tau$,

$$c_\sigma^{(L)}(t_j) = c_\gamma^{(L)}(t_m) \iff c_\sigma^{(L+1)}(t_j) = c_\gamma^{(L+1)}(t_m).$$

Because each update retains the previous color as an argument, color classes can split but cannot merge. The partition therefore stabilizes after finitely many iterations. Stabilization does not require the numerical or symbolic color labels themselves to remain unchanged. We denote the terminal color by $c_\sigma(t_j) := c_\sigma^{(L)}(t_j)$ and, under a fixed ordering of $\mathbb{S}(t_j)$, $\boldsymbol{c}(t_j) = [c_\sigma(t_j)]_{\sigma \in \mathbb{S}(t_j)}$, which represents

the stable partition of the observed TSC at time t_j. TSWL terminates at horizon τ once the joint partition over all timestamps in $\mathbb{T}_\tau$ has stabilized.

Distinguishing TSCs. Given two TSCs, we compare their color multisets separately for each simplex dimension and timestamp. TSWL distinguishes the inputs if

$$\{\!\{c_\sigma^{(\ell)}(t_j) \mid \sigma \in \mathbb{S}_k(t_j)\}\!\} \neq \{\!\{c_\gamma'^{(\ell)}(t_j) \mid \gamma \in \mathbb{S}'_k(t_j)\}\!\}$$

for some iteration ℓ, dimension k, and timestamp $t_j \in \mathbb{T}_\tau$. Such a mismatch certifies non-isomorphism while failure to find one before stabilization does not imply isomorphism. Further analysis, including a comparison with Temporal Weisfeiler–Lehman (TWL) (Beddar-Wiesing & Moallemy-Oureh, 2025), is deferred to Appendix C.

Definition 2 (TSWL Equivalence)**.** *Let σ and γ be simplices observed by time τ in the same or different TSCs, compared over $\mathbb{T}_\tau$ using shared initial coloring and hash functions. They are TSWL-equivalent up to τ, denoted by $\sigma \equiv_{\mathrm{TSWL},\tau} \gamma$, if and only if $c_\sigma^{(\ell)}(t_j) = c_\gamma^{(\ell)}(t_j)$ for every $\ell \geq 0$ and every $t_j \in \mathbb{T}_\tau$. We omit the horizon subscript when it is clear from context.*

This equivalence groups simplices that TSWL cannot distinguish, and Proposition 1 establishes that TSC isomorphisms preserve these color histories.

Proposition 1 (TSWL Isomorphism Invariance)**.** *Let $\mathcal{D}(\tau)$ and $\mathcal{D}'(\tau)$ be isomorphic TSCs such that $\mathcal{D}(\tau) \approx \mathcal{D}'(\tau)$. Then, for every $\sigma' \in \mathbb{S}'(\tau)$, the corresponding simplices σ' and $\sigma = f(\sigma')$ are TSWL-equivalent up to τ, i.e., $\sigma' \equiv_{\mathrm{TSWL},\tau} \sigma$.*

The result follows by induction on the refinement iteration and identical initial colors for isomorphic TSC. The complete proof is provided in Appendix E.2.

4 Temporal Simplicial Networks

We introduce Temporal Simplicial Networks (TSNs), a representation learning framework that combines simplicial neighborhood encoding with temporal sequence encoding. TSNs mirror the two components of simplex-level TSWL equivalence: color refinement at each observed timestamp and sequence matching of the resulting color trajectories. For a query time τ, let $\mathbb{D}(\tau) := (\mathcal{D}(t_j))_{t_j \in \mathbb{T}_\tau}$ denote the historical TSCs ordered chronologically. A TSN combines a structural encoder Q, which maps each $\mathcal{D}(t_j)$ and its features to simplex embeddings, with a sequence encoder κ that maps variable-length embedding trajectories to fixed-dimensional representations. We establish sufficient conditions under which this construction matches TSWL's ability to distinguish simplices.

Definition 3 (Temporal Simplicial Network)**.** *For a query timestamp $\tau \in \mathbb{T}$, a* TSN *processes historical TSCs in $\mathbb{D}(\tau)$ to produce a temporal representation $\boldsymbol{m}(\sigma, \tau) \in \mathbb{R}^D$ for each simplex $\sigma \in \mathbb{S}(\tau)$. For each $t_j \in \mathbb{T}_\tau$, set*

$$\boldsymbol{h}_\sigma(t_j) := Q(\boldsymbol{X}(t_j), \mathcal{D}(t_j))_\sigma \in \mathbb{R}^d, \sigma \in \mathbb{S}(t_j)$$

The temporal representation of σ at time τ is defined as $\boldsymbol{m}(\sigma, \tau) = \kappa\Big(\big(\boldsymbol{h}_\sigma(t_j)\big)_{t_j \in \mathbb{T}_\tau}\Big)$, where $\kappa : \mathcal{H}^* \to \mathbb{R}^D$ is a sequence encoder and $\mathcal{H}^* := \bigcup_{r=0}^{\infty} \mathbb{R}^{d \times r}$ is the space of arbitrary length sequences, including the empty sequence. Sequence entries follow increasing timestamp order to preserve causality.

The expressiveness of a TSN depends on both stages: Q must preserve the distinctions made by TSWL at each timestamp, while κ must preserve distinctions between the resulting embedding trajectories. We formalize the requirement on Q first.

Definition 4 (Maximally Expressive Structural Encoder)**.** *A structural encoder Q is* maximally expressive *relative to TSWL on a class of attributed TSCs if, for every timestamp t_j, every pair of TSCs $\mathcal{D}(t_j), \mathcal{D}'(t_j)$ in that class, and all $\sigma \in \mathbb{S}(t_j)$ and $\gamma \in \mathbb{S}'(t_j)$,*

$$Q(\boldsymbol{X}(t_j), \mathcal{D}(t_j))_\sigma = Q(\boldsymbol{X}'(t_j), \mathcal{D}'(t_j))_\gamma \iff c_\sigma(t_j) = c'_\gamma(t_j).$$

Here, $c_\sigma(t_j)$ and $c'_\gamma(t_j)$ are terminal TSWL colors.

Remark. A message-passing architecture of Q is described in Appendix D. Exact correspondence with stable TSWL colors requires injective initial feature encoding and injective message, multiset aggregation, and update maps on the relevant input domains, together with sufficient refinement depth. See Lemma 2.

Theorem 1 (TSWL Expressiveness of TSNs)**.** *Let Q be a maximally expressive structural encoder. If the sequence encoder κ is injective on the embedding sequences generated on this class, including, then TSN matches simplex-level TSWL equivalence. Specifically, for TSCs $\mathcal{D}(\tau)$ and $\mathcal{D}'(\tau)$ compared over the same timestamps $\mathbb{T}_\tau$,*

$$\sigma \equiv_{\mathrm{TSWL},\tau} \gamma \iff \boldsymbol{m}(\sigma,\tau) = \boldsymbol{m}'(\gamma,\tau), \qquad \sigma \in \mathbb{S}(\tau), \quad \gamma \in \mathbb{S}'(\tau),$$

where $\boldsymbol{m}$ and $\boldsymbol{m}'$ are computed in their respective TSCs using the same Q and κ. This includes the case in which both simplices belong to the same TSC.

The equivalence follows because, by injectivity of κ, equal temporal representations imply equal ordered embedding sequences. The maximally expressive Q then makes this equivalent to equality of the corresponding TSWL color histories. The complete proof is provided in Appendix E.3.

This theorem identifies conditions for attaining TSWL-level expressiveness. This, however, does not imply that every finite-dimensional sequence encoder satisfies the required injectivity. To develop an efficient online TSN architecture, we next introduce the *Temporal Simplicial State Space Model* (TS^{3}M) and analyze how its state dimension controls its expressive power.

5 TS^{3}M: Temporal Simplicial State Space Models

Theorem 1 establishes that a TSN matches the discriminative power of the TSWL test provided its sequence encoder κ is injective and Q is maximally expressive. The remaining challenge is to construct a finite-dimensional representation that summarizes the embedding trajectory while preserving TSWL equivalence, which we formalize as the *Embedding Reconstruction Problem* (ERP).

Given $\mathcal{D}(\tau)$, historically observed TSCs $\mathbb{D}(\tau)$, a structural encoder Q, and simplices $\mathbb{S}(\tau)$, let $\hat{\boldsymbol{x}}(\sigma,t) := \boldsymbol{x}(\sigma,t) \oplus \rho\boldsymbol{m}(\sigma,t)$ define the *temporal features* [2] for a simplex $\sigma \in \mathbb{S}(t)$ at time $t \le \tau$, where $\boldsymbol{x}(\sigma,t)$ denotes raw input features, $\boldsymbol{m}(\sigma,t)$ is the memory state of σ, $\rho \in [0,1]$ is the memory coefficient, and $\oplus$ denotes feature concatenation. Accordingly, $\hat{\boldsymbol{X}}(t) := \boldsymbol{X}(t) \oplus \rho\boldsymbol{M}(t)$ represents the temporal feature matrix over a TSC $\mathcal{D}(t) \in \mathbb{D}(\tau)$.

ERP formulation. Let $\boldsymbol{g}(t) \in \mathbb{R}^D$ denote a vector of orthogonal polynomial basis functions defined over the interval $[0,\tau]$. For each simplex σ, ERP computes the memory state vector $\boldsymbol{m}(\sigma,\tau) \in \mathbb{R}^D$ by projecting its embedding trajectory $\{Q(\hat{\boldsymbol{X}}(t),\mathcal{D}(t))_\sigma\}_{t\le\tau}$ onto this basis[3]:

$$Q(\hat{\boldsymbol{X}}(t),\mathcal{D}(t))_\sigma = \boldsymbol{m}(\sigma,\tau)^\top \boldsymbol{g}(t) + \boldsymbol{r}_\sigma(t), \qquad \forall\, t \le \tau, \tag{6}$$

where $\boldsymbol{r}_\sigma(t)$ represents the reconstruction residual. Stacking equation 6 across all simplices in $\mathcal{D}(\tau)$, the ERP objective under a weight measure $\mu(t)$ over $[0,\tau]$ minimizes the weighted mean-squared reconstruction error

$$\min_{\boldsymbol{M}(\tau)} \int_0^\tau \left\| Q(\hat{\boldsymbol{X}}(t),\mathcal{D}(t)) - \boldsymbol{M}(\tau)\boldsymbol{g}(t) \right\|_2^2 \mu(t)\mathrm{d}t, \tag{7}$$

where $\boldsymbol{M}(\tau) \in \mathbb{R}^{|\mathbb{S}(\tau)|\times D}$ stacks the optimal state representations $\boldsymbol{m}(\sigma,\tau)^\top$ row-wise. For orthogonal polynomial basis and corresponding weight function pairs given in (Gu et al., 2020), the unique minimizer is given by

$$\boldsymbol{M}(\tau) = \int_0^\tau Q(\hat{\boldsymbol{X}}(t),\mathcal{D}(t))\boldsymbol{g}(t)^\top \mu(t)\,\mathrm{d}t \tag{8}$$

Note that ERP provides a generalized framework for constructing topological reconstruction problem. Under different parametrization of Q, other variants of topological HiPPOs (Li et al., 2024; Raghuvanshi et al., 2026) can be derived as shown in Appendix B.

[2] Memory-augmented features allow the encoder to use previously accumulated information (Rossi et al., 2020; Souza et al., 2022).

[3] ERP presented here assumes scaler embeddings, i.e., $d = 1$, for multi-channel inputs coefficients are can be derived for each channel independently along the similar lines.

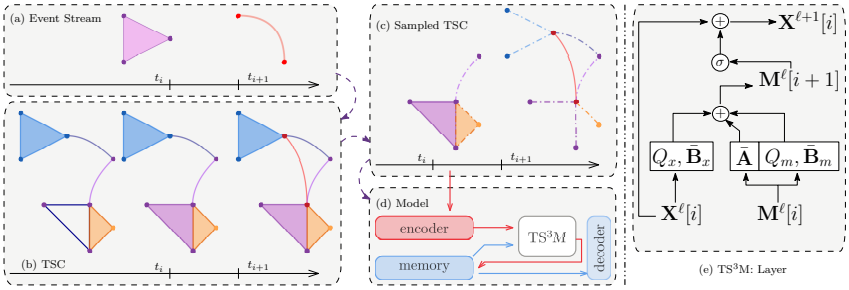


Figure 4: Illustration of event stream, TSC, sampling over TSC and single layer TS^{3}M diagram.

As ERP produced a finite dimensional representation, it expressivity is bounded by the dimention of state vector $\boldsymbol{m}(\sigma, t)$ as formalized in Theorem 2.

Theorem 2 (Expressivity of finite-dimensional ERP)**.** *Let $\boldsymbol{M}_D(\tau)$ denote the ERP representation obtained by projecting the embedding trajectory generated memory-less features ($\rho = 0$) encoded by a maximally expressive structural encoder Q onto the first D basis functions of a complete polynomial family. For $D' > D$, the representation $\boldsymbol{M}_{D'}(\tau)$ is at least as expressive as $\boldsymbol{M}_D(\tau)$.*

Moreover, as $D \to \infty$, completeness of the polynomial basis ensures that the ERP representation distinguishes any distinct embedding trajectories in $L^2([0, \tau], \mu)$. Consequently, the infinite-dimensional ERP representation attains the distinguishing power of the TSWL test.

The proof of Theorem 2 relegated to Appendix E.4.

Through equation 8 we established the relationship between the state matrix $\boldsymbol{M}(\tau)$ and the embedding trajectory. To avoid recomputing the full ERP solution upon observing new events, we follow (Gu et al., 2020) and characterize $\boldsymbol{M}(\tau)$ as the solution to a linear ODE, enabling efficient online updates.

Theorem 3 (Continuous State Evolution)**.** *For a TSC $\mathcal{D}(\tau)$ and a structural encoder Q with memory coefficient ρ, the state evolution in response to a new event $e(\tau)$ is governed by:*

$$\frac{\mathrm{d}}{\mathrm{d}\tau}\boldsymbol{M}(\tau) = -\boldsymbol{M}(\tau)\frac{\boldsymbol{A}^\top}{T} + Q(\mathcal{D}(\tau), \boldsymbol{X}(\tau) \oplus \rho\boldsymbol{M}(\tau))\frac{\boldsymbol{B}^\top}{T}, \tag{9}$$

where $\boldsymbol{A} \in \mathbb{R}^{D\times D}$ and $\boldsymbol{B} \in \mathbb{R}^{D\times 1}$ are the HiPPO transition matrices and T is normalizing coefficient $T \in (0, \tau]$ associated with the polynomial basis $\boldsymbol{g}(t)$.

Theorem 3 follows by differentiating the integral expression for $\boldsymbol{M}(\tau)$ in equation 8 with respect to τ using Leibniz's rule, and using the recurrence satisfied by the orthogonal polynomial basis $\boldsymbol{g}(t)$, yields the stated ODE. Full derivation in Appendix E.5.

Although Q can take arbitrary functional forms—yielding various topological HiPPO variants, as shown in Appendix B—in this work we focus on simple additive parameterization:

$$Q(\boldsymbol{X}(t) \oplus \boldsymbol{M}(t), \mathcal{D}(t)) := Q_x(\boldsymbol{X}(t), \mathcal{D}(t)) + \rho Q_m(\boldsymbol{M}(t), \mathcal{D}(t)), \quad \text{st. } \rho, \in [0, 1], \tag{10}$$

this parameterization allows flexible control over architecture of Q, where different encoders architectures can be used for states and features respectively [4]. Using this for equation 3 yields

$$\frac{\mathrm{d}\boldsymbol{M}(\tau)}{\mathrm{d}\tau} = -\frac{\boldsymbol{M}(\tau)}{T}\boldsymbol{A}^\top + \rho\frac{Q_m(\boldsymbol{M}(\tau), \mathcal{D}(\tau))}{T}\boldsymbol{B}^\top + \frac{Q_x(\boldsymbol{X}(\tau), \mathcal{D}(\tau))}{T}\boldsymbol{B}^\top. \tag{11}$$

The simplest form such Q is a single-layer linear aggregation, given in Appendix D.2.

Discretization. Through ERP, we derived the continuous-time state evolution equation equation 11. Since data in a TSC arrives as timestamped events $e(t_i)$, implementing this equation requires a discrete-time recurrence for $\boldsymbol{M}[i] := \boldsymbol{M}(t_i)$ over a timestamps in $\mathbb{T}$. We adopt piece-wise constant approximation along with zero-order hold to arrive at discrete $\boldsymbol{M}, \boldsymbol{A}, \boldsymbol{B}$.. More details on how to arrive at the discrete models can be found in Appendix F.

[4] With $\rho = 0$, the encoder depended only on features, as seen is other variants of topological HiPPO

6 Model Architecture

In this section, we present TS^{3}M as a multilayer state space model. As illustrated in Figure 4, the computational pipeline operates across three primary stages

Sampling and feature extraction. In this stage, observed events are grouped into batches of B sequential events, and at most N_s neighbors are sampled for each observed simplex. Together, the observed simplices and their sampled neighbors constitute the active pool of $N_B \ll N(t_i)$ simplices to be updated. Each simplex σ in this pool is assigned an input feature vector $\hat{\boldsymbol{x}}_\sigma$. For simplices directly associated with an observed event $E(t)$, $\hat{\boldsymbol{x}}_\sigma$ is constructed by concatenating raw input features $\boldsymbol{x}(\sigma, t)$ from the event with a non-learnable temporal encoding $\boldsymbol{\phi}(t)$, *i.e.*, $\hat{\boldsymbol{x}}_\sigma = [\boldsymbol{x}(\sigma, t) \| \boldsymbol{\phi}(t)]$. If raw event features are unavailable, structural properties such as simplex order and node degree are used alongside $\boldsymbol{\phi}(t)$. Conversely, sampled neighbor simplices are assigned zero input vectors $\mathbf{0}$.

State update. The pool of N_B simplices and their assigned feature matrix $\boldsymbol{X}[i]$ serve as the input to the state update given in Theorem 6 at time t_i. Following the layer-stacking paradigm of Mamba Gu & Dao (2024), we process these features through L cascading layers. For a given layer $\ell \in \{1, \ldots, L\}$ with input $\boldsymbol{X}^\ell[i]$ and state $\boldsymbol{M}^\ell[i]$, the state update and layer output $\boldsymbol{X}^{\ell+1}[i+1]$ are evaluated as

$$\boldsymbol{Z}^\ell[i] = \mathrm{RMSNorm}(\boldsymbol{X}^\ell[i]), \quad \boldsymbol{X}^{\ell+1}[i+1] = \boldsymbol{X}^\ell[i] + \mathrm{GeLU}(\boldsymbol{M}^\ell[i+1]), \tag{12}$$

$$\boldsymbol{M}^\ell[i+1] = \boldsymbol{M}^\ell[i]\bar{\boldsymbol{A}}^\top + \rho Q_m(\boldsymbol{M}^\ell[i], \hat{\mathcal{D}}(t_i))\bar{\boldsymbol{B}}_m^\top + Q_x(\boldsymbol{Z}^\ell[i], \hat{\mathcal{D}}(t_i))\bar{\boldsymbol{B}}_x^\top, \tag{13}$$

where $\bar{\boldsymbol{B}}_x$ and $\bar{\boldsymbol{B}}_m$ denote the discretized counterparts of the learnable input and state projection matrices $\boldsymbol{B}_x$ and $\boldsymbol{B}_c$, respectively. While Equation 12 incorporates a residual connection to promote stable gradient propagation across deep layers. Using a sampled local neighborhood reduces the update complexity from $\mathcal{O}(BN(t_i)D^2)$ to $\mathcal{O}(N_B D^2)$, where $N_B \ll N(t_i)$ per batch of events, enabling scalable online learning. A detailed analysis of the effects of sampling and state dimension D on predictive performance and per-epoch time is presented in Appendix H.

Decoder. For downstream tasks, the decoder combines the final-layer states of a query simplex and its neighboring simplices with their temporal embeddings. For simplex σ, the temporal embedding is $\psi(\sigma, t) = \texttt{Temporal-encoder}(t - t_\sigma)$, where t_σ denotes the last update time of simplex σ. The combined input vector $\bar{\boldsymbol{m}}(\sigma, t)$ is defined as

$$\bar{\boldsymbol{m}}(\sigma, t) = \Big[\boldsymbol{m}^L(\sigma, t_\sigma) \oplus \psi(\sigma, t) \oplus \mathrm{Agg}\left(\left\{\boldsymbol{m}^L(\gamma, t_\gamma) \oplus \psi(\gamma, t) \mid (\gamma, t_\gamma) \in \mathbb{N}(\sigma, t)\right\}\right)\Big], \tag{14}$$

where $\mathbb{N} := \{\mathbb{N}_\downarrow, \mathbb{N}_\uparrow, \mathbb{B}, \mathbb{C}\}$ specifies the neighborhoods of σ, and $\mathrm{Agg}(\cdot)$ is an aggregation function (e.g., mean pooling or attention). The final prediction is computed via a task-specific learnable transformation $\boldsymbol{W}$ as $\text{TS}^3\text{M}(\sigma, t) = \boldsymbol{W}\bar{\boldsymbol{m}}(\sigma, t)$.

7 Numerical Experiments

Higher-order link prediction (HLP). Following (Behrouz et al., 2023), we evaluate TS^{3}M on HLP using random negative samples across three evaluation settings (randomly masking 10% of nodes for inductive evaluation): *transductive* (no nodes masked), *weakly inductive* (≥ 1 masked node per test sample), and *strongly inductive* (all nodes in each test sample masked during training). Table 1 reports the mean AUC-ROC $\pm$ standard deviation across five randomly initialized runs.

Baselines and datasets. We compare TS^{3}M against CAT-Walk (Behrouz et al., 2023) and its eight baselines from deep hypergraph learning (HyperSAGNN, NHP, CHESHIRE), shallow methods (HPRA, HPLSF), and CE techniques (CE-CAW, CE-EvolveGCN, CE-GCN). Furthermore, we include unsupervised simplicial heuristics from (Benson et al., 2018) and FastHeP (Xu et al., 2025). Evaluation spans seven dynamic higher-order datasets across four domains: social (CHS, CPS, CB), email (EEN, EEU), QA forums (MQT), and drug registration. Appendix I provides complete dataset, baseline, and implementation details.

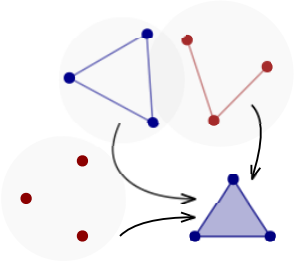


Figure 5: Simplicial closure, triplets forming 2-simplex in an evolving TSC.

Table 1: Performance HLP: mean AUC (%) ± std. Bold indicates the best result, while underline indicates results within one standard deviation of the best result.

	Methods	NDC	CHS	CPS	CB	EEN	EEU	MQT
Strongly Inductive	CE-GCN	52.31 ± 2.99	60.54 ± 2.06	52.34 ± 2.75	49.18 ± 3.61	63.04 ± 1.80	52.76 ± 2.41	56.10 ± 1.88
	CE-EvolveGCN	49.78 ± 3.13	46.12 ± 3.83	58.01 ± 2.56	54.00 ± 1.84	57.31 ± 4.19	44.16 ± 1.27	64.08 ± 2.75
	CE-CAW	76.45 ± 0.29	83.73 ± 1.42	80.31 ± 1.46	75.38 ± 1.25	70.81 ± 1.13	72.99 ± 0.20	70.14 ± 1.89
	NHP	70.53 ± 4.95	65.29 ± 3.80	70.86 ± 3.42	69.82 ± 2.19	49.71 ± 6.09	65.35 ± 2.07	68.23 ± 3.34
	HyperSAGNN	79.05 ± 2.48	88.12 ± 3.01	80.13 ± 1.38	79.51 ± 1.27	73.09 ± 2.60	78.01 ± 1.24	73.66 ± 1.95
	CHESHIRE	72.24 ± 2.63	82.54 ± 0.88	77.26 ± 1.01	79.43 ± 1.58	70.03 ± 2.55	69.98 ± 2.71	N/A
	FastHeP	**99.32 ± 1.67**	94.03 ± 1.40	94.98 ± 0.59	**95.34 ± 0.86**	88.91 ± 2.92	92.78 ± 1.98	92.63 ± 1.38
	CAT-Walk	98.89 ± 1.82	**96.03 ± 1.50**	**95.32 ± 0.89**	93.54 ± 0.56	73.45 ± 2.92	91.68 ± 2.78	88.03 ± 3.38
	TS^{3}M	97.23 ± 0.38	95.30 ± 0.89	94.08 ± 0.20	92.32 ± 0.47	**89.06 ± 3.28**	**93.34 ± 0.54**	**93.63 ± 0.77**
Weakly Inductive	CE-GCN	51.80 ± 3.29	50.33 ± 3.40	52.19 ± 2.54	52.38 ± 2.75	50.81 ± 2.87	49.60 ± 3.96	55.13 ± 2.76
	CE-EvolveGCN	55.39 ± 5.16	57.85 ± 3.51	51.50 ± 4.07	55.63 ± 3.41	45.66 ± 2.10	52.44 ± 2.38	61.79 ± 1.63
	CE-CAW	77.61 ± 1.05	83.77 ± 1.41	82.98 ± 1.06	79.51 ± 0.94	80.54 ± 1.02	73.54 ± 1.19	77.29 ± 0.86
	NHP	75.17 ± 2.02	67.25 ± 5.19	71.92 ± 1.83	69.58 ± 4.07	60.38 ± 4.45	67.19 ± 4.33	70.46 ± 3.52
	HyperSAGNN	79.45 ± 2.18	88.53 ± 1.26	85.08 ± 1.45	80.12 ± 2.00	78.86 ± 0.63	77.26 ± 2.09	78.15 ± 1.41
	CHESHIRE	79.03 ± 1.24	88.40 ± 1.06	83.55 ± 1.27	79.67 ± 0.83	74.53 ± 0.91	77.31 ± 0.95	N/A
	FastHeP	99.45 ± 0.98	93.76 ± 1.92	94.65 ± 2.18	98.66 ± 0.27	88.95 ± 4.37	94.52 ± 2.87	94.98 ± 2.53
	CAT-Walk	99.16 ± 1.08	**94.68 ± 2.37**	**96.53 ± 1.39**	98.38 ± 0.21	64.11 ± 7.96	91.98 ± 2.41	90.28 ± 2.81
	TS^{3}M	**99.51 ± 0.07**	94.42 ± 0.65	93.89 ± 0.20	**99.44 ± 0.06**	**89.15 ± 1.65**	**96.79 ± 0.56**	**95.45 ± 0.48**
Transductive	HPRA	70.83 ± 0.01	94.91 ± 0.00	89.86 ± 0.06	79.48 ± 0.03	78.62 ± 0.00	72.51 ± 0.00	83.18 ± 0.00
	HPLSF	76.19 ± 0.82	92.14 ± 0.29	88.57 ± 1.09	79.31 ± 0.52	75.73 ± 0.05	75.27 ± 0.31	83.45 ± 0.93
	CE-GCN	66.83 ± 3.74	62.99 ± 3.02	59.14 ± 3.87	64.42 ± 3.11	58.06 ± 3.80	64.19 ± 2.79	55.18 ± 5.12
	CE-EvolveGCN	67.08 ± 3.51	65.19 ± 2.26	63.15 ± 1.32	69.30 ± 2.27	69.98 ± 5.38	64.36 ± 4.17	72.56 ± 1.72
	CE-CAW	76.30 ± 0.84	81.63 ± 0.97	86.53 ± 0.84	76.99 ± 1.02	79.57 ± 0.14	78.19 ± 1.10	81.73 ± 2.48
	NHP	82.39 ± 2.81	76.85 ± 3.08	80.04 ± 3.42	80.27 ± 2.53	63.17 ± 3.79	78.90 ± 4.39	79.14 ± 3.36
	HyperSAGNN	80.76 ± 2.64	94.98 ± 1.30	90.77 ± 2.05	82.84 ± 1.61	83.59 ± 0.98	79.61 ± 2.35	84.07 ± 2.50
	CHESHIRE	84.91 ± 1.05	95.11 ± 0.94	91.62 ± 1.18	86.81 ± 1.24	82.27 ± 0.86	86.38 ± 1.23	N/A
	FastHeP	98.52 ± 1.22	94.86 ± 0.92	95.81 ± 2.18	97.16 ± 1.27	88.45 ± 3.30	95.72 ± 1.83	94.15 ± 1.52
	CAT-Walk	**98.72 ± 1.38**	**95.30 ± 0.43**	**97.91 ± 3.30**	88.15 ± 1.46	80.45 ± 5.30	**96.74 ± 1.28**	91.63 ± 1.41
	TS^{3}M	98.53 ± 1.42	95.20 ± 0.45	94.17 ± 0.19	**98.41 ± 0.11**	**90.89 ± 0.06**	95.43 ± 0.21	**94.59 ± 0.47**
	Heuristic	94.50	63.44	59.35	56.16	65.53	74.11	75.03

Dynamic closure prediction (DCP). DCP evaluates a model's ability to capture higher-order link-formation mechanisms by predicting which candidate triplet will close into a 2-simplex. The candidate pool consists of nodes observed up to query time τ, together with triplets that close at $\tau + 1$. The model must identify which candidate will form a filled triangle (Figure 5 illustrates closure). We report AUC for DCP on CHS, CPS, EEN and EEU in Figure 6.

Results and discussion. Table 1 shows that TS^{3}M performs competitively with temporal hypergraph models (FastHeP and CAT-Walk), supporting the use of temporal simplices for modeling higher-order interactions. Its state-space formulation also captures temporal dynamics efficiently: as shown in Figure 1, TS^{3}M has substantially lower per-epoch cost than hypergraph models due to its limited first-hop neighborhood and inexpensive online state updates. In contrast, CE-graph models are effective mainly on dyadic-dominated datasets (CHS: $\sim$ 92%; CPS: $\sim$ 90%), but degrade significantly on datasets rich in higher-order interactions (EEN, EEU, MQT, CB), performing marginally above heuristic baselines. This contrast highlights the benefit of explicitly modeling higher-order structure. Additional ablations in Appendix H show that memory-augmented features improve prediction, while larger neighborhood samples provide additional gains at the cost of slower state updates. Increasing the state dimension also helps, although performance eventually plateaus.

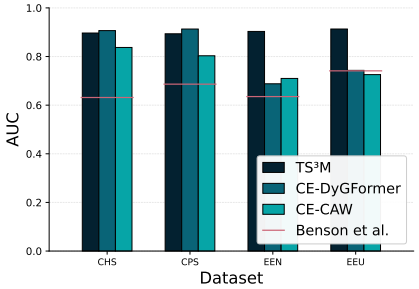


Figure 6: AUC for TS^{3}M and CE models DCP.

8 Conclusion

We introduced TSC, a simplicial-complex-based framework for modeling temporal higher-order interactions. We formalized TSC isomorphism and developed a TSWL test for distinguishing TSCs. Building on this theory, we proposed TSN, an abstract learning framework that achieves TSWL expressivity for a class of functions. We further developed TS^{3}M, an efficient model within this framework, and characterized the relationship between its state dimension and expressive power. Experiments on HLP and DCP demonstrate the effectiveness of modeling temporal interactions with TSCs and the efficiency of state-based updates, establishing TS^{3}M as a fast and scalable alternative to temporal hypergraph models.

AI USE STATEMENT

We have not used generative AI tools for any of the tasks requiring mandatory disclosure under the ICLR 2027 AI Policy for Authors (e.g., generating synthetic datasets, developing theoretical models or conceptual frameworks, formulating mathematical claims, providing critical ingredients for proofs, proposing or refining hypotheses, designing or giving feedback on research methodology or experiments, implementing methods, translation, cleaning or reformatting datasets, qualitative or thematic data analysis, or interpreting results); these tasks were performed solely by the authors. Additionally, we used generative AI tools for tasks under the recommended-disclosure category: editing the manuscript for grammar and clarity, drafting portions of the text, conducting a literature survey to help identify and summarize related work, and reorganizing existing experiment code for readability and structure. All AI-assisted text, literature summaries, and code changes were reviewed and verified by the authors, who take full responsibility for the final content of this work, including text, claims, and artifacts produced with the aid of generative AI.

REPRODUCIBILITY STATEMENT

To support reproducibility, we provide the full model implementation as supplementary material, along with the exact hyperparameter settings used for each dataset in Table 3 and the experimental setup detailed in Table 2. All reported results were obtained using fixed training and test seeds for each dataset split; these seeds are included in the released code so that our evaluation numbers can be regenerated exactly. Proofs of our theoretical claims (TSC isomorphism, the TSWL test, and the expressiveness bound for TS³M) are given in full in the appendix, together with the assumptions under which they hold.

REFERENCES

Sergio Barbarossa and Stefania Sardellitti. Topological Signal Processing Over Simplicial Complexes. *IEEE Transactions on Signal Processing*, 68:2992–3007, 2020. ISSN 1941-0476. doi: 10.1109/TSP.2020.2981920. URL https://ieeexplore.ieee.org/document/9044758.

Claudio Battiloro, Lucia Testa, Lorenzo Giusti, Stefania Sardellitti, Paolo Di Lorenzo, and Sergio Barbarossa. Generalized Simplicial Attention Neural Networks. *IEEE Transactions on Signal and Information Processing over Networks*, 10:833–850, 2024. ISSN 2373-776X. doi: 10.1109/TSIPN.2024.3485473. URL https://ieeexplore.ieee.org/abstract/document/10735150.

Silvia Beddar-Wiesing and Alice Moallemy-Oureh. Weisfeiler-Lehman meets Events: An Expressivity Analysis for Continuous-Time Dynamic Graph Neural Networks, August 2025. URL http://arxiv.org/abs/2508.18052. arXiv:2508.18052 [cs.LG].

Ali Behrouz, Farnoosh Hashemi, Sadaf Sadeghian, and Margo Seltzer. CAT-Walk: Inductive Hypergraph Learning via Set Walks. November 2023. URL https://openreview.net/forum?id=QG4nJBNEar.

Austin R. Benson, Rediet Abebe, Michael T. Schaub, Ali Jadbabaie, and Jon Kleinberg. Simplicial closure and higher-order link prediction. *Proceedings of the National Academy of Sciences of the United States of America*, 115(48):E11221–E11230, November 2018. ISSN 0027-8424. doi: 10.1073/pnas.1800683115. URL https://pmc.ncbi.nlm.nih.gov/articles/PMC6275482/.

Cristian Bodnar, Fabrizio Frasca, Yu Guang Wang, Nina Otter, Guido Montúfar, Pietro Liò, and Michael Bronstein. Weisfeiler and Lehman Go Topological: Message Passing Simplicial Networks, June 2021. URL http://arxiv.org/abs/2103.03212. arXiv:2103.03212 [cs].

Alberto Ceria and Huijuan Wang. Temporal-topological properties of higher-order evolving networks. *Scientific Reports*, 13(1):5885, April 2023. ISSN 2045-2322. doi: 10.1038/s41598-023-32253-9. URL https://www.nature.com/articles/s41598-023-32253-9.

Can Chen, Chen Liao, and Yang-Yu Liu. Teasing out missing reactions in genome-scale metabolic networks through hypergraph learning. *Nature Communications*, 14(1):2375, April 2023.

ISSN 2041-1723. doi: 10.1038/s41467-023-38110-7. URL https://doi.org/10.1038/s41467-023-38110-7.

Zifeng Ding, Yifeng Li, Yuan He, Antonio Norelli, Jingcheng Wu, Volker Tresp, Michael M. Bronstein, and Yunpu Ma. DyGMamba: Efficiently Modeling Long-Term Temporal Dependency on Continuous-Time Dynamic Graphs with State Space Models. August 2025. URL https://openreview.net/forum?id=Dlaio5e3vq.

Stefania Ebli, Michaël Defferrard, and Gard Spreemann. Simplicial neural networks. In *Topological Data Analysis and Beyond workshop at NeurIPS*, 2020. URL https://arxiv.org/abs/2010.03633.

James H. Fowler. Connecting the Congress: A Study of Cosponsorship Networks. *Political Analysis*, 14(4):456–487, October 2006. ISSN 1047-1987, 1476-4989. doi: 10.1093/pan/mpl002. URL https://www.cambridge.org/core/journals/political-analysis/article/abs/connecting-the-congress-a-study-of-cosponsorship-networks/B42907E13C3D1F12BBC7618C8E0EECED.

Tony Gracious and Ambedkar Dukkipati. Dynamic Representation Learning with Temporal Point Processes for Higher-Order Interaction Forecasting. *Proceedings of the AAAI Conference on Artificial Intelligence*, 37(6):7748–7756, June 2023. ISSN 2374-3468. doi: 10.1609/aaai.v37i6.25939. URL https://ojs.aaai.org/index.php/AAAI/article/view/25939.

Albert Gu and Tri Dao. Mamba: Linear-Time Sequence Modeling with Selective State Spaces, May 2024. URL http://arxiv.org/abs/2312.00752. arXiv:2312.00752 [cs].

Albert Gu, Tri Dao, Stefano Ermon, Atri Rudra, and Christopher Re. HiPPO: Recurrent Memory with Optimal Polynomial Projections, October 2020. URL http://arxiv.org/abs/2008.07669. arXiv:2008.07669 [cs].

Sravanthi Gurugubelli and S. Chepuri. SaNN: Simple Yet Powerful Simplicial-aware Neural Networks. 2024. URL https://www.semanticscholar.org/paper/SaNN%3A-Simple-Yet-Powerful-Simplicial-aware-Neural-Gurugubelli-Chepuri/ab48c421d4f0d6e6a4393667b5d60750e22138eb.

Sunwoo Kim, Soo Yong Lee, Yue Gao, Alessia Antelmi, Mirko Polato, and Kijung Shin. A Survey on Hypergraph Neural Networks: An In-Depth and Step-By-Step Guide. In *Proceedings of the 30th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, KDD '24, pp. 6534–6544, New York, NY, USA, August 2024. Association for Computing Machinery. ISBN 979-8-4007-0490-1. doi: 10.1145/3637528.3671457. URL https://dl.acm.org/doi/10.1145/3637528.3671457.

Joshin Krishnan, Rohan Money, Baltasar Beferull-Lozano, and Elvin Isufi. Simplicial Vector Autoregressive Models. *IEEE Transactions on Signal Processing*, 72:5454–5469, 2024. ISSN 1941-0476. doi: 10.1109/TSP.2024.3503063. URL https://ieeexplore.ieee.org/document/10758681.

Tarun Kumar, K Darwin, Srinivasan Parthasarathy, and Balaraman Ravindran. HPRA: Hyperedge Prediction using Resource Allocation. In *Proceedings of the 12th ACM Conference on Web Science*, WebSci '20, pp. 135–143, New York, NY, USA, 2020. Association for Computing Machinery. ISBN 978-1-4503-7989-2. doi: 10.1145/3394231.3397903. URL https://doi.org/10.1145/3394231.3397903.

Jintang Li, Ruofan Wu, Xinzhou Jin, Boqun Ma, Liang Chen, and Zibin Zheng. State Space Models on Temporal Graphs: A First-Principles Study. November 2024. URL https://openreview.net/forum?id=UaJErAOssN.

Rossana Mastrandrea, Julie Fournet, and Alain Barrat. Contact Patterns in a High School: A Comparison between Data Collected Using Wearable Sensors, Contact Diaries and Friendship Surveys. *PLOS ONE*, 10(9):e0136497, September 2015. ISSN 1932-6203. doi: 10.1371/journal.pone.0136497. URL https://journals.plos.org/plosone/article?id=10.1371/journal.pone.0136497.

Marco Montagna, Simone Scardapane, and Lev Telyatnikov. Topological Deep Learning with State-Space Models: A Mamba Approach for Simplicial Complexes. In *2025 International Joint Conference on Neural Networks (IJCNN)*, pp. 1–8, June 2025. doi: 10.1109/IJCNN64981.2025.11227272. URL https://ieeexplore.ieee.org/document/11227272. ISSN: 2161-4407.

Aldo Pareja, Giacomo Domeniconi, Jie Chen, Tengfei Ma, Toyotaro Suzumura, Hiroki Kanezashi, Tim Kaler, Tao Schardl, and Charles Leiserson. EvolveGCN: Evolving Graph Convolutional Networks for Dynamic Graphs. *Proceedings of the AAAI Conference on Artificial Intelligence*, 34(04):5363–5370, April 2020. ISSN 2374-3468. doi: 10.1609/aaai.v34i04.5984. URL https://ojs.aaai.org/index.php/AAAI/article/view/5984.

Alice Patania, Giovanni Petri, and Francesco Vaccarino. The shape of collaborations. *EPJ Data Science*, 6(1):18, August 2017. ISSN 2193-1127. doi: 10.1140/epjds/s13688-017-0114-8. URL https://doi.org/10.1140/epjds/s13688-017-0114-8.

H. A. (Bart) Peters, Alberto Ceria, and Huijuan Wang. Higher-order temporal network prediction and interpretation. *PLOS ONE*, 20(5):e0323753, May 2025. ISSN 1932-6203. doi: 10.1371/journal.pone.0323753. URL https://journals.plos.org/plosone/article?id=10.1371/journal.pone.0323753.

Ayushman Raghuvanshi, Thummaluru Siddartha Reddy, Sundeep Prabhakar Chepuri, and Mahesh Chandran. Learning Long Range Spatio-Temporal Representations over Continuous Time Dynamic Graphs with State Space Models. June 2026. URL https://openreview.net/forum?id=S2jfT6EMqr.

Emanuele Rossi, Ben Chamberlain, Fabrizio Frasca, Davide Eynard, Federico Monti, and Michael Bronstein. Temporal Graph Networks for Deep Learning on Dynamic Graphs, October 2020. URL http://arxiv.org/abs/2006.10637. arXiv:2006.10637 [cs.LG].

Amauri H. Souza, Diego Mesquita, Samuel Kaski, and Vikas K. Garg. Provably expressive temporal graph networks. October 2022. URL https://openreview.net/forum?id=MwSXgQSxL5s.

Juliette Stehlé, Nicolas Voirin, Alain Barrat, Ciro Cattuto, Lorenzo Isella, Jean-François Pinton, Marco Quaggiotto, Wouter Van den Broeck, Corinne Régis, Bruno Lina, and Philippe Vanhems. High-Resolution Measurements of Face-to-Face Contact Patterns in a Primary School. *PLOS ONE*, 6(8):e23176, August 2011. ISSN 1932-6203. doi: 10.1371/journal.pone.0023176. URL https://journals.plos.org/plosone/article?id=10.1371/journal.pone.0023176.

Hao Tian and Reza Zafarani. Higher-Order Networks Representation and Learning: A Survey. *ACM SIGKDD Explorations Newsletter*, 26(1):1–18, July 2024. ISSN 1931-0145. doi: 10.1145/3682112.3682114. URL https://dl.acm.org/doi/10.1145/3682112.3682114.

Yanbang Wang, Yen-Yu Chang, Yunyu Liu, Jure Leskovec, and Pan Li. Inductive Representation Learning in Temporal Networks via Causal Anonymous Walks, October 2022. URL http://arxiv.org/abs/2101.05974. arXiv:2101.05974 [cs.LG].

Felix Wu, Amauri Souza, Tianyi Zhang, Christopher Fifty, Tao Yu, and Kilian Weinberger. Simplifying Graph Convolutional Networks. In *Proceedings of the 36th International Conference on Machine Learning*, pp. 6861–6871. PMLR, May 2019. URL https://proceedings.mlr.press/v97/wu19e.html.

Ye Xu, Dan Rockmore, and Adam M. Kleinbaum. Hyperlink Prediction in Hypernetworks Using Latent Social Features. In Johannes Fürnkranz, Eyke Hüllermeier, and Tomoyuki Higuchi (eds.), *Discovery Science*, pp. 324–339, Berlin, Heidelberg, 2013. Springer. ISBN 978-3-642-40897-7. doi: 10.1007/978-3-642-40897-7_22.

Yuanyuan Xu, Wenjie Zhang, Ying Zhang, Xiwei Xu, and Xuemin Lin. Fast and Accurate Temporal Hypergraph Representation for Hyperedge Prediction. In *Proceedings of the 31st ACM SIGKDD Conference on Knowledge Discovery and Data Mining V.1*, KDD '25, pp. 1727–1738, New

York, NY, USA, July 2025. Association for Computing Machinery. ISBN 979-8-4007-1245-6. doi: 10.1145/3690624.3709327. URL https://dl.acm.org/doi/10.1145/3690624.3709327.

Naganand Yadati, Vikram Nitin, Madhav Nimishakavi, Prateek Yadav, Anand Louis, and Partha Talukdar. NHP: Neural Hypergraph Link Prediction. In *Proceedings of the 29th ACM International Conference on Information & Knowledge Management*, CIKM '20, pp. 1705–1714, New York, NY, USA, 2020. Association for Computing Machinery. ISBN 978-1-4503-6859-9. doi: 10.1145/3340531.3411870. URL https://doi.org/10.1145/3340531.3411870.

Maosheng Yang, Bishwadeep Das, and Elvin Isufi. Online Edge Flow Prediction Over Expanding Simplicial Complexes. In *ICASSP 2023 - 2023 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 1–5, June 2023. doi: 10.1109/ICASSP49357.2023.10096364. URL https://ieeexplore.ieee.org/document/10096364. ISSN: 2379-190X.

Hao Yin, Austin R. Benson, Jure Leskovec, and David F. Gleich. Local Higher-Order Graph Clustering. In *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '17, pp. 555–564, New York, NY, USA, 2017. Association for Computing Machinery. ISBN 978-1-4503-4887-4. doi: 10.1145/3097983.3098069. URL https://doi.org/10.1145/3097983.3098069.

Se-eun Yoon, Hyungseok Song, Kijung Shin, and Yung Yi. How Much and When Do We Need Higher-order Information in Hypergraphs? A Case Study on Hyperedge Prediction. In *Proceedings of The Web Conference 2020*, WWW '20, pp. 2627–2633, New York, NY, USA, April 2020. Association for Computing Machinery. ISBN 978-1-4503-7023-3. doi: 10.1145/3366423.3380016. URL https://dl.acm.org/doi/10.1145/3366423.3380016.

Le Yu, Leilei Sun, Bowen Du, and Weifeng Lv. Towards Better Dynamic Graph Learning: New Architecture and Unified Library. November 2023. URL https://openreview.net/forum?id=xHNzWHbklj.

Ruochi Zhang, Yuesong Zou, and Jian Ma. Hyper-SAGNN: a self-attention based graph neural network for hypergraphs, November 2019. URL http://arxiv.org/abs/1911.02613. arXiv:1911.02613 [cs.LG].

A Related works

A.1 Representation learning for Temporal Hypergraphs

Temporal hypergraphs have been widely adopted for modeling higher-order dynamic interactions, where hyperedges capture multi-node group relations over continuous time (Behrouz et al., 2023; Gracious & Dukkipati, 2023; Xu et al., 2025). However, existing representation learning methods suffer from two primary limitations. First, standard neighbor aggregation schemes encounter high spatio-temporal complexity due to exponentially expanding neighborhoods. Although (Xu et al., 2025) drastically reduced this overhead by introducing an online hyperedge-centric neighbor store, recent findings indicate that not all higher-order neighbor relations are informative in dynamic settings (Peters et al., 2025). Second, current temporal hypergraph architectures lack rigorous theoretical expressivity guarantees.

These limitations motivate our formulation of TSC. By establishing a canonical neighborhood definition restricted to lower and upper adjacencies, TSCs provide a computationally efficient structural representation that mitigates neighborhood explosion while enabling formal theoretical guarantees on model expressivity.

A.2 Representation Learning for Simplicial Complexes

Higher-order relational learning has gained significant traction due to the expressiveness of SC neural networks over traditional graph neural networks (GNNs) (Bodnar et al., 2021; Ebli et al., 2020). Early foundational work emphasized the structural relevance of higher-order motifs and non-dyadic link formations across various domains (Benson et al., 2018; Patania et al., 2017). However, much of the literature focuses on static SC or models temporal dynamic data by aggregating events over large time windows (Gurugubelli & Chepuri, 2024; Battiloro et al., 2024; Montagna et al., 2025; Ebli et al., 2020), where the train, validation, and test SC are constructed by aggregating all events within these three respective time windows. Such snapshot-based aggregations collapse fine-grained temporal ordering, losing critical chronological sequence context, limiting the focus of these works to structural representation learning.

Recent attempts to incorporate temporal information into topological representations have explored learning temporal signals over static SC (Krishnan et al., 2024), expanding SC (Yang et al., 2023), or post-hoc topological analysis (Ceria & Wang, 2023). Nevertheless, these methods do not handle continuous-time, online streaming events where simplices of arbitrary dimensions are dynamically created or removed. In contrast to these static or constrained settings, our work provides a principled online state-update framework for TSC.

A.3 Representation Learning over Dynamic Graphs and Sequence Models

Representation learning on dynamic graphs generally encompasses Discrete-Time Dynamic Graphs (DTDGs) and Continuous-Time Dynamic Graphs (CTDGs) (Rossi et al., 2020; Souza et al., 2022; Yu et al., 2023). While early dynamic graph architectures relied heavily on message-passing mechanisms or recurrent memory modules, they often struggled to capture long-range temporal and spatial dependencies.

To address these limitations, attention and State Space Models (SSMs) mechanism have been adapted to dynamic graph structures. Models such as DyGMamba (Ding et al., 2025) leverage selective state spaces for sequence modeling on dynamic graphs while DyGFormer Yu et al. (2023) uses transformer for modeling these sequence. While GHiPPO (Li et al., 2024) and CTT-HiPPO (Raghuvanshi et al., 2026) incorporate graph structural operators directly into the polynomial projection and continuous-state evolution. However, these methods remain strictly bounded by dyadic graph assumptions, i.e., pairwise edges. In this paper, we bridge this gap by introducing a unified ERP framework that generalizes existing SSM-based objectives for dynamic graphs, including both discrete- and continuous-time formulations, and extends them to TSC.

B VARIANTS OF TOPOLOGICAL SSM

In this section we will describe how other variants of topological HiPPO can be derived as a special case of ERP under prescribed parametrization of encoder Q.

CTT-HiPPO as a special case of ERP. The CTT-HiPPO framework of Raghuvanshi et al. (2026) is recovered as a particular instance of ERP by choosing the embedding function

$$\mathcal{Q} = \left[p_k(\boldsymbol{L}_k(t))^{-1}\boldsymbol{X}_k(t)\right]_{k=0}^{K}, \tag{15}$$

where $p_k(\boldsymbol{L}_k)$ denotes the polynomial filter acting on the k-Hodge Laplacian. Consequently, CTT-HiPPO corresponds to reconstructing polynomial graph-filter trajectories instead of TSN embedding trajectories.

Proof. CTT-HiPPO uses graph polynomial filters together with temporal polynomial bases to reconstruct the observed signal over graphs, solving

$$\min_{\boldsymbol{M}^0(\tau)} \int_0^\tau \left\|\boldsymbol{X}^0(t) - p(\boldsymbol{L}_0(\tau))\boldsymbol{M}^0(\tau)\boldsymbol{g}(t)\right\|_F^2 \mu(t)\,dt, \tag{16}$$

where $p(\boldsymbol{L}_0(\tau))$ is invertible graph filter over graph Laplacian $\boldsymbol{L}$, and $\boldsymbol{g}(t)$ is the orthonormal temporal polynomial basis vector of the underlying HiPPO measure (Gu et al., 2020), satisfying $\int_0^\tau \boldsymbol{g}(t)\boldsymbol{g}(t)^\top\mu(t)\,dt = \boldsymbol{I}$. Under these (standing) conditions, the optimum is

$$\boldsymbol{M}(\tau) = p(\boldsymbol{L}(\tau))^{-1}\int_0^\tau \boldsymbol{X}(t)\boldsymbol{g}(t)^\top\mu(t)\,\mathrm{d}t \tag{17}$$

Now consider the ERP objective

$$\min_{\boldsymbol{M}(\tau)} \int_0^\tau \|Q(\hat{\boldsymbol{X}}(t), \mathcal{D}(t)) - \boldsymbol{M}(\tau)\boldsymbol{g}(t)\|_F^2\mu(t)\,\mathrm{d}t \tag{18}$$

with $Q(\hat{\boldsymbol{X}}(t), \mathcal{D}(t))$ restricted to node states with $\rho = 0$ and defined as

$$Q(\boldsymbol{X}(t), \mathcal{D}(t)) := p(\boldsymbol{L}_0(\tau))^{-1}\boldsymbol{X}^0(t), \qquad \forall t \in [0, \tau]. \tag{19}$$

That is, instead of using the topology observed at time t, we fix the topological filter at the query time τ, and restrict Q to an inverse polynomial filter with no dependence on the state $\boldsymbol{M}(t)$. We inherit the same invertibility condition on $p(\boldsymbol{L}_0(\tau))$ and orthonormality of $\boldsymbol{g}(t)$ used by CTT-HiPPO above, ERP imposes no additional assumption here.

Substituting equation 19 into the general ERP optimum gives

$$\boldsymbol{M}^0(\tau) = \int_0^\tau Q(\hat{\boldsymbol{X}}(t), \mathcal{D}(t))\,\boldsymbol{g}(t)^\top\mu(t)\,dt = \int_0^\tau p(\boldsymbol{L}(\tau))^{-1}\boldsymbol{X}^0(t)\boldsymbol{g}(t)^\top\mu(t)\,\mathrm{d}t \tag{20}$$

Since $p(\boldsymbol{L}(\tau))^{-1}$ is evaluated at the fixed query time τ, it does not depend on the integration variable t and can be factored outside the integral:

$$\boldsymbol{M}^0(\tau) = p(\boldsymbol{L}(\tau))^{-1}\int_0^\tau \boldsymbol{X}^0(t)\boldsymbol{g}(t)^\top\mu(t)\,dt, \tag{21}$$

which is exactly the CTT-HiPPO optimum in equation 17. Thus, under the choice of Q in equation 19, ERP and CTT-HiPPO yield the *same optimal state matrix*, even though the two objectives differ in general (the filter $p(\boldsymbol{L}(\tau))^{-1}$ acts on the target $\boldsymbol{X}(t)$ in the CTT-HiPPO objective, and on the encoder output Q in the ERP instance) — showing that CTT-HiPPO is recovered as a special case of ERP. Notably, this reduction freezes the topological filter at the query time τ; the general ERP objective instead allows Q to depend on the full trajectory $\{\boldsymbol{L}(t)\}_{t\le\tau}$, so CTT-HiPPO is recovered only when this time-dependence is collapsed to a single evaluation at τ.

□

GHiPPO as a special case of ERP. Similarly, GraphSSM (Li et al., 2024) is obtained by selecting

$$\mathcal{Q} = \left[\left(\boldsymbol{I} + \alpha \frac{\boldsymbol{L}_k(t) + \boldsymbol{L}_k(t)^\top}{2} \right)^{-1} \boldsymbol{X}_k(t) \right]_{k=0}^{K}, \tag{22}$$

which corresponds to the quadratic smoothness reconstruction objective used in GHiPPO. Unlike CTT-HiPPO, GHiPPO incorporates the complete Laplacian history but remains a special case of the general ERP formulation.

Proof. B.1 REDUCTION TO GHIPPO

(Li et al., 2024), proposed a regularizer based objective GHiPPO for reconstruction problem,

$$\int_0^\tau \|\boldsymbol{X}^0(t) - \boldsymbol{M}^0(\tau)\boldsymbol{g}(t)\|_2^2 \, \mu(t)dt + \gamma \int_0^\tau \boldsymbol{g}(t)^\top \boldsymbol{M}^0(\tau)^\top \boldsymbol{L}_0(t) \boldsymbol{M}^0(\tau)\boldsymbol{g}(t)\mu(t) \, dt \tag{23}$$

promoting smooth reconstruction of the observed signal over the evolving topology. It characterizes the GHiPPO abstraction via the following continuous evolution (Li et al., 2024)[Theorem 1]

$$\frac{\boldsymbol{M}(\tau)}{\mathrm{d}\tau} = \boldsymbol{M}(\tau)\boldsymbol{A}^\top + (\boldsymbol{I} + \gamma \boldsymbol{L}(\tau))^{-1} \boldsymbol{X}(\tau) \tag{24}$$

Now, consider ERP with encoder $Q(\hat{\boldsymbol{X}}(t), \mathcal{D}(t)) = (\boldsymbol{I} + \gamma \frac{\boldsymbol{L}_0(t) + \boldsymbol{L}_0(t)^\top}{2})^{-1} \boldsymbol{X}(t)$, with $\rho = 0$. Solving for ERP with this Q, gives

$$\frac{\boldsymbol{M}(\tau)}{\mathrm{d}\tau} = \boldsymbol{M}(\tau)\boldsymbol{A}^\top + \left(\boldsymbol{I} + \gamma \frac{\boldsymbol{L}_0(\tau) + \boldsymbol{L}_0(\tau)^\top}{2} \right)^{-1} \boldsymbol{X}(\tau) \tag{25}$$

the exact same evolving equation as equation 24. Indicating that GrapSSM is a special case of ERP based SSM under restricted Q. □

C TSWL FOR DYNAMIC GRAPHS

In this section we discuss the expressivity of TSWL when applied to dynamic graphs, and present a detailed theoretical analysis for the same.

Theorem 4 (TSWL Expressivity Compared to TWL)**.** *TSWL with clique complex lifting is strictly more expressive than TWL on dynamic graphs.*

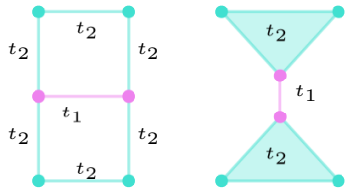


Figure 7: Two non-isomorphic TSCs that are indistinguishable by the TWL test on their CE temporal graphs but can be distinguished by TSWL test.

Consider a 1-dimensional TSC, i.e., a dynamic graph with vertex attributes. In this case, the upper adjacent neighbors of a 0-simplex are exactly its graph neighbors, while boundary and lower adjacency relations are absent. Consequently, the neighborhood aggregation used by TSWL reduces to the temporal neighborhood aggregation of TWL. Hence, restricted to 1-dimensional TSCs, TSWL recovers the TWL refinement procedure. In particular, TSWL is at least as expressive as TWL. Now consider the pair of non-isomorphic TSCs in Figure 7 and there corresponding projected dynamic graphs in Figure 8.

C.1 PROOF OF THEOREM 4

The color of each node indicate initial colors assigned to these node from using node features.

Figure 8 illustrates the color assignment at each successive iterations of TWL test on the project graphs. Here the stable the final readout for both the graphs is identical, and thus TWL fails to distinguish these graphs, while TSWL easily distinguishes them due to presence of close traingle in one while absence of such in another.

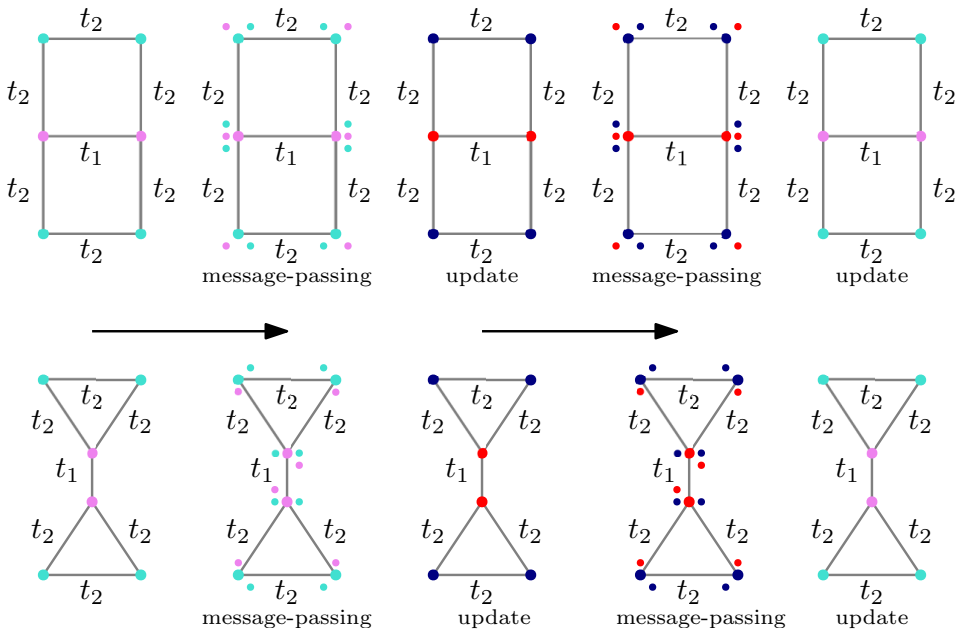


Figure 8: Successive iteration of TWL on pair of graph projection of pair of non-isomorphic TSC.

D Structural Encoder

In this section, we introduce a architecture for structural encoder Q using temporally weighted message passing.

D.1 Message Passing over TSC.

For an observed TSC $\mathcal{D}(t_i)$, let $\boldsymbol{h}_\sigma^{(\ell)}(t_i) \in \mathbb{R}^d$ denote the representation of simplex σ after the ℓ-th message passing iteration.

The initial simplex representation is obtained through an injective encoder

$$\boldsymbol{h}_\sigma^{(0)}(t_i) = \phi(\boldsymbol{x}_\sigma(t_i))\,, \tag{26}$$

where ϕ is shared across all simplices.

For every refinement iteration, messages are independently constructed from the temporal upper, lower, boundary, and co-boundary neighborhoods,

$$\mathbb{M}_\uparrow^{(\ell)}(\sigma, t_i) = \left\{\phi_\uparrow(\boldsymbol{h}_\gamma^{(\ell)}(t), \boldsymbol{h}_{\gamma\cup\sigma}^{(\ell)}(t), \psi(t_i - t)) \mid (\gamma, t) \in \mathbb{N}_\uparrow(\sigma, t_i)\right\}, \tag{27}$$

$$\mathbb{M}_\downarrow^{(\ell)}(\sigma, t_i) = \left\{\phi_\downarrow(\boldsymbol{h}_\gamma^{(\ell)}(t), \boldsymbol{h}_{\gamma\cap\sigma}^{(\ell)}(t), \psi(t_i - t)) \mid (\gamma, t) \in \mathbb{N}_\downarrow(\sigma, t_i)\right\}, \tag{28}$$

$$\mathbb{M}_\mathbb{B}^{(\ell)}(\sigma, t_i) = \left\{\phi_\mathbb{B}(\boldsymbol{h}_\gamma^{(\ell)}(t), \psi(t_i - t)) \mid (\gamma, t) \in \mathbb{B}(\sigma, t_i)\right\}, \tag{29}$$

$$\mathbb{M}_\mathbb{C}^{(\ell)}(\sigma, t_i) = \left\{\phi_\mathbb{C}(\boldsymbol{h}_\gamma^{(\ell)}(t), \psi(t_i - t)) \mid (\gamma, t) \in \mathbb{C}(\sigma, t_i)\right\}, \tag{30}$$

where ψ denotes a temporal encoding function.

The message multisets are independently aggregated through permutation-invariant aggregation operators,

$$\boldsymbol{a}_\uparrow^{(\ell)}(\sigma, t_i) = \mathrm{AGG}_\uparrow(\mathbb{M}_\uparrow^{(\ell)}(\sigma, t_i)), \tag{31}$$

$$\boldsymbol{a}_\downarrow^{(\ell)}(\sigma, t_i) = \mathrm{AGG}_\downarrow(\mathbb{M}_\downarrow^{(\ell)}(\sigma, t_i)), \tag{32}$$

$$\boldsymbol{a}_\mathbb{B}^{(\ell)}(\sigma, t_i) = \mathrm{AGG}_\mathbb{B}(\mathbb{M}_\mathbb{B}^{(\ell)}(\sigma, t_i)), \tag{33}$$

$$\boldsymbol{a}_\mathbb{C}^{(\ell)}(\sigma, t_i) = \mathrm{AGG}_\mathbb{C}(\mathbb{M}_\mathbb{C}^{(\ell)}(\sigma, t_i)). \tag{34}$$

The simplex representation is then updated according to

$$\boldsymbol{h}_\sigma^{(\ell+1)}(t_i) = U\Big(\boldsymbol{h}_\sigma^{(\ell)}(t_i), \boldsymbol{a}_\uparrow^{(\ell)}(\sigma, t_i), \boldsymbol{a}_\downarrow^{(\ell)}(\sigma, t_i), \boldsymbol{a}_\mathbb{B}^{(\ell)}(\sigma, t_i), \boldsymbol{a}_\mathbb{C}^{(\ell)}(\sigma, t_i)\Big), \tag{35}$$

where U denotes a learnable update function. After L iterations, the embedding of simplex σ at timestamp t_i is given by

$$\boldsymbol{h}_\sigma(t_i) = \boldsymbol{h}_\sigma^{(L)}(t_i). \tag{36}$$

D.2 Simple Additive Encoder

In this section, we present a single layer implementation of a encoder with additive decomposition, $Q_\theta^1(\boldsymbol{Y}(t), \mathcal{D}(t)) =$

$$\begin{aligned}
&= \frac{\theta_0 \boldsymbol{Y}(t) + \theta_1 \boldsymbol{A}^\uparrow(\tau)(\Gamma(t) * \boldsymbol{Y}(t)) + \theta_2 \boldsymbol{A}^\downarrow(\tau)(\Gamma(t) * \boldsymbol{Y}(t)) + \theta_3 \partial(\Gamma(t) * \boldsymbol{Y}(t)) + \theta_4 \partial^\top(\Gamma(t) * \boldsymbol{Y}(t))}{\sum_i \theta_i} \\
&= \frac{\big(\theta_0 \boldsymbol{I} + \boldsymbol{\Phi}(\mathbf{t})\big)(\Gamma(t) * \boldsymbol{Y}(t))}{\sum_i \theta_i}
\end{aligned} \tag{37}$$

where $\theta = \{\theta_0, \theta_1, \theta_2, \theta_3, \theta_4\}$ are learnable parameters, ∂ and $\partial^\top$ are temporal boundary/co-boundary maps, and $\Gamma(t)$ is temporal scaling factor of form $[\Gamma(t)]_\sigma = \sum_{(t_j,\sigma)\in\mathcal{D}(t)} e^{-\alpha_t(t-t_j)}$ for $\alpha_t > 0$, and $\sigma \in S_\tau$. The $\boldsymbol{\Phi}(t) \in \mathbb{R}^{N\times N}$ is a block-tridiagonal operator induced by $\mathcal{D}(\tau)$,

$$\boldsymbol{\Phi}(t) = \begin{bmatrix}
\theta_1 \boldsymbol{A}_0^\uparrow(\tau) + \theta_2 \boldsymbol{A}_0^\downarrow(\tau) & \theta_3 \boldsymbol{B}_1(\tau) & 0 & \cdots & 0 \\
\theta_4 \boldsymbol{B}_1(\tau)^\top & \theta_1 \boldsymbol{A}_1^\uparrow(\tau) + \theta_2 \boldsymbol{A}_1^\downarrow(\tau) & \theta_3 \boldsymbol{B}_2(\tau) & \cdots & 0 \\
0 & \theta_4 \boldsymbol{B}_2(\tau)^\top & \ddots & \ddots & \vdots \\
\vdots & & \ddots & \ddots & \theta_3 \boldsymbol{B}_K(\tau) \\
0 & \cdots & \cdots & \theta_4 \boldsymbol{B}_K(\tau)^\top & \theta_1 \boldsymbol{A}_K^\uparrow(\tau) + \theta_2 \boldsymbol{A}_K^\downarrow(\tau)
\end{bmatrix}.$$

Each diagonal block $(\boldsymbol{\Phi}(t))_{k,k}$ aggregates over same-order upper/lower neighbors via $\boldsymbol{A}_k^\uparrow(\tau), \boldsymbol{A}_k^\downarrow(\tau)$, while the off-diagonal blocks $(\Phi(t))_{k,k+1} = \theta_3 \boldsymbol{B}_{k+1}(\tau)$ and $(\boldsymbol{\Phi}(t))_{k,k-1} = \theta_4 \boldsymbol{B}_k(\tau)^\top$ couple each order only to its immediate boundary and co-boundary. Similarly, multi-layer aggregation can be used to capture information from multi-hop neighbors.

Multiple such layer can be stacked to form Q; additionally feature mixing encoder can also be used as long as the joint encoder is defined as $Q(\hat{\boldsymbol{X}}(t), \mathcal{D}(t)) := Q_{\theta_x}(\boldsymbol{X}(t), \mathcal{D}(t)) + \rho Q_{\theta_m}(\boldsymbol{M}(t), \mathcal{D}(t))$.

E Proofs of theorems

We star by stating all the lemmas required for the proofs, followed by the proofs of the theorems.

E.1 Proof of Lemmas

Lemma 1 (Single step Correspondence TSWL and Q)**.** *Suppose the message functions $\phi_\uparrow, \phi_\downarrow, \phi_\mathbb{B}, \phi_\mathbb{C}$, the aggregation operators $\mathrm{AGG}_\uparrow, \mathrm{AGG}_\downarrow, \mathrm{AGG}_\mathbb{B}, \mathrm{AGG}_\mathbb{C}$, and the update function U are injective. If*

$$\boldsymbol{h}_\sigma^{(\ell)}(t) = \boldsymbol{h}_\gamma^{(\ell)}(t) \iff c_\sigma^{(\ell)}(t) = c_\gamma^{(\ell)}(t),$$

then

$$\boldsymbol{h}_\sigma^{(\ell+1)}(t) = \boldsymbol{h}_\gamma^{(\ell+1)}(t) \iff c_\sigma^{(\ell+1)}(t) = c_\gamma^{(\ell+1)}.(t)$$

Proof. We start by proving the forwards direction as first part of the proof, and then the reverse direction completing proof.

Embedding ⇒ Color

Given

$$\boldsymbol{h}_\sigma^{(\ell)}(\tau) = \boldsymbol{h}_\gamma^{(\ell)}(\tau) \iff c_\sigma^{(\ell)}(\tau) = c_\gamma^{(\ell)}(\tau).$$

We have to show that

$$\boldsymbol{h}_{\sigma}^{(\ell+1)}(\tau) = \boldsymbol{h}_{\gamma}^{(\ell+1)}(\tau) \implies c_{\sigma}^{(\ell+1)}(\tau) = c_{\gamma}^{(\ell+1)}(\tau).$$

The $(\ell+1)$-th layer embeddings are calculated as

$$\boldsymbol{h}_{\sigma}^{(\ell+1)}(\tau) = U\Big(\boldsymbol{h}_{\sigma}^{(\ell)}(\tau), \boldsymbol{a}_{\uparrow}^{(\ell)}(\sigma,\tau), \boldsymbol{a}_{\downarrow}^{(\ell)}(\sigma,\tau), \boldsymbol{a}_{\mathbb{B}}^{(\ell)}(\sigma,\tau), \boldsymbol{a}_{\mathbb{C}}^{(\ell)}(\sigma,\tau)\Big), \tag{38}$$

$$\boldsymbol{h}_{\gamma}^{(\ell+1)}(\tau) = U\Big(\boldsymbol{h}_{\gamma}^{(\ell)}(\tau), \boldsymbol{a}_{\uparrow}^{(\ell)}(\gamma,\tau), \boldsymbol{a}_{\downarrow}^{(\ell)}(\gamma,\tau), \boldsymbol{a}_{\mathbb{B}}^{(\ell)}(\gamma,\tau), \boldsymbol{a}_{\mathbb{C}}^{(\ell)}(\gamma,\tau)\Big). \tag{39}$$

Since U is injective, identical output by U implies identical inputs given to U, thus

$$\boldsymbol{h}_{\sigma}^{(\ell)}(\tau) = \boldsymbol{h}_{\gamma}^{(\ell)}(\tau),$$
$$\boldsymbol{a}_{\star}^{(\ell)}(\sigma,\tau) = \boldsymbol{a}_{\star}^{(\ell)}(\gamma,\tau),\ \star \in \{\uparrow,\downarrow,\mathbb{B},\mathbb{C}\}$$

Since each aggregation operator $\mathrm{AGG}_{\star}$ is injective and $a_{\star}^{(\ell)}(\sigma,\tau) = \mathrm{AGG}_{\star}(\mathbb{M}_{\star}^{(\ell)}(\tau))$ for $\star \in \{\uparrow,\downarrow,\mathbb{B},\mathbb{C}\}$, the equality in equation 40 implies that message multisets

$$\mathbb{M}_{\star}^{(\ell)}(\sigma,\tau) = \mathbb{M}_{\star}^{(\ell)}(\gamma,\tau). \tag{40}$$

For a message defined as equation 27, and injectivity of message function $\phi_{\star}$ we get,

$$\{\!\{(\boldsymbol{h}_{\sigma'}^{(\ell)}, \boldsymbol{h}_{\sigma\cup\sigma'}^{(\ell)}, \psi(\tau-t)) \mid (\sigma',t) \in \mathbb{N}_{\uparrow}(\sigma,\tau)\}\!\}$$
$$= \{\!\{(\boldsymbol{h}_{\gamma'}^{(\ell)}, \boldsymbol{h}_{\gamma\cup\gamma'}^{(\ell)}, \psi(\tau-t)) \mid (\gamma',t) \in \mathbb{N}_{\uparrow}(\gamma,\tau)\}\!\}, \tag{41}$$

and similarly for the remaining three neighborhood types. Since ψ is injective and given

$$\boldsymbol{h}_{\sigma}^{(\ell)}(\tau) = \boldsymbol{h}_{\gamma}^{(\ell)}(\tau) \iff c_{\sigma}^{(\ell)}(\tau) = c_{\gamma}^{(\ell)}(\tau),$$

we convert the equality multiset of embedding in equation 41 to equality of multiset of color for each neighborhood as

$$c_{\star}^{(\ell)}(\sigma,\tau) = c_{\star}^{(\ell)}(\gamma,\tau), \qquad \forall \star \in \{\uparrow,\downarrow,\mathbb{B},\mathbb{C}\}, \tag{42}$$

together with

$$c_{\sigma}^{(\ell)}(\tau) = c_{\gamma}^{(\ell)}(\tau).$$

We get,

$$c_{\sigma}^{(\ell+1)}(\tau) = \mathrm{HASH}\Big(c_{\sigma}^{(\ell)}(\tau), c_{\uparrow}^{(\ell)}(\sigma,\tau), c_{\downarrow}^{(\ell)}(\sigma,\tau), c_{\mathbb{B}}^{(\ell)}(\sigma,\tau), c_{\mathbb{C}}^{(\ell)}(\sigma,\tau)\Big), \tag{43}$$

$$c_{\gamma}^{(\ell+1)}(\tau) = \mathrm{HASH}\Big(c_{\gamma}^{(\ell)}(\tau), c_{\uparrow}^{(\ell)}(\gamma,\tau), c_{\downarrow}^{(\ell)}(\gamma,\tau), c_{\mathbb{B}}^{(\ell)}(\gamma,\tau), c_{\mathbb{C}}^{(\ell)}(\gamma,\tau)\Big). \tag{44}$$

For an injective HASH map we thus conclude that,

$$c_{\sigma}^{(\ell+1)}(\tau) = c_{\gamma}^{(\ell+1)}(\tau),$$

which completes this part of the proof. Next we will prove the reverse direction.

Color ⇒ Embedding.
Given

$$\boldsymbol{h}_{\sigma}^{(\ell)}(\tau) = \boldsymbol{h}_{\gamma}^{(\ell)}(\tau) \iff c_{\sigma}^{(\ell)}(\tau) = c_{\gamma}^{(\ell)}(\tau).$$

We have to show that

$$c_{\sigma}^{(\ell+1)}(\tau) = c_{\gamma}^{(\ell+1)}(\tau) \implies \boldsymbol{h}_{\sigma}^{(\ell+1)}(\tau) = \boldsymbol{h}_{\gamma}^{(\ell+1)}(\tau).$$

For $c_{\sigma}^{(\ell+1)}(\tau) = c_{\gamma}^{(\ell+1)}(\tau)$, the inputs must be identical since $\mathrm{HASH}(\cdot)$ is injective, therefore,

$$c_{\sigma}^{(\ell)}(\tau) = c_{\gamma}^{(\ell)}(\tau),$$
$$c_{\star}^{(\ell)}(\sigma,\tau) = c_{\star}^{(\ell)}(\gamma,\tau), \star \in \{\uparrow,\downarrow,\mathbb{B},\mathbb{C}\}.$$

Given $\boldsymbol{h}_{\sigma}^{(\ell)}(\tau) = \boldsymbol{h}_{\gamma}^{(\ell)}(\tau)$, and following the same arguments as in the forward direction, the equality of the TSWL neighborhood multisets together with the injectivity of ψ, $\phi_{\star}$, and $\mathrm{AGG}_{\star}$ implies

$$\boldsymbol{a}_{\star}^{(\ell)}(\sigma,\tau) = \boldsymbol{a}_{\star}^{(\ell)}(\gamma,\tau), \star \in \{\uparrow,\downarrow,\mathbb{B},\mathbb{C}\}. \tag{45}$$

Finally, using the injectivity of the update function U,

$$\begin{aligned}\boldsymbol{h}_{\sigma}^{(\ell+1)}(\tau) &= U\Big(\boldsymbol{h}_{\sigma}^{(\ell)}(\tau), \boldsymbol{a}_{\uparrow}^{(\ell)}(\sigma,\tau), \boldsymbol{a}_{\downarrow}^{(\ell)}(\sigma,\tau), \boldsymbol{a}_{\mathbb{B}}^{(\ell)}(\sigma,\tau), \boldsymbol{a}_{\mathbb{C}}^{(\ell)}(\sigma,\tau)\Big) \\ &= U\Big(\boldsymbol{h}_{\gamma}^{(\ell)}(\tau), \boldsymbol{a}_{\uparrow}^{(\ell)}(\gamma,\tau), \boldsymbol{a}_{\downarrow}^{(\ell)}(\gamma,\tau), \boldsymbol{a}_{\mathbb{B}}^{(\ell)}(\gamma,\tau), \boldsymbol{a}_{\mathbb{C}}^{(\ell)}(\gamma,\tau)\Big) \\ &= \boldsymbol{h}_{\gamma}^{(\ell+1)}(\tau),\end{aligned} \tag{46}$$

which completes the proof. □

Lemma 2 (Correspondence between Q Embeddings and TSWL Colors)**.** *Suppose the initial encoder, message functions, aggregation operators, and update function of TSN are injective. Then, for every timestamp $t_i \le \tau$,*

$$\boldsymbol{h}_{\sigma}^{(L)}(t_i) = \boldsymbol{h}_{\gamma}^{(L)}(t_i) \iff c_{\sigma}(t_i) = c_{\gamma}(t_i),$$

where $\boldsymbol{h}_{\sigma}^{(L)}(t_i)$ denotes the final-layer embedding of simplex σ at timestamp t_i, and $c_{\sigma}(t_i)$ denotes its stable TSWL color.

Proof. We prove the result by induction on the message passing layers.

At initialization ($\ell = 0$), the initial encoder is injective. Hence,

$$\boldsymbol{h}_{\sigma}^{(0)}(t_i) = \boldsymbol{h}_{\gamma}^{(0)}(t_i) \iff c_{\sigma}^{(0)}(t_i) = c_{\gamma}^{(0)}(t_i).$$

Assume that for some layer ℓ,

$$\boldsymbol{h}_{\sigma}^{(\ell)}(t_i) = \boldsymbol{h}_{\gamma}^{(\ell)}(t_i) \iff c_{\sigma}^{(\ell)}(t_i) = c_{\gamma}^{(\ell)}(t_i).$$

By Lemma 1, the above equivalence is preserved after one refinement step, i.e.,

$$\boldsymbol{h}_{\sigma}^{(\ell+1)}(t_i) = \boldsymbol{h}_{\gamma}^{(\ell+1)}(t_i) \iff c_{\sigma}^{(\ell+1)}(t_i) = c_{\gamma}^{(\ell+1)}(t_i).$$

Therefore, by induction, the equivalence holds for every message passing layer.

Let L denote the final message passing layer. Since TSWL has converged,

$$c_{\sigma}^{(\ell)}(t_i) = c_{\sigma}(t_i), \qquad \forall\, \ell \ge L,$$

where $c_{\sigma}(t_i)$ denotes the stable TSWL color of simplex σ. Consequently,

$$\boldsymbol{h}_{\sigma}^{(L)}(t_i) = \boldsymbol{h}_{\gamma}^{(L)}(t_i) \iff c_{\sigma}(t_i) = c_{\gamma}(t_i),$$

which proves the lemma. □

Lemma 3 (TSN Invariance to TSWL equivalence)**.** *Given a TSC $\mathcal{D}(\tau)$, a TSN as defined in 3 with Q and κ, if two simplices $\sigma, \gamma \in \mathbb{S}(\tau)$ are TSWL equivalent $\sigma \equiv_{TSWL} \gamma$ then TSN produces same embedding for both the simplices.*

Proof. Given, $\boldsymbol{m}(\sigma,\tau) = \kappa(\{Q(\hat{\boldsymbol{X}}(t), \mathcal{D}(t))_{\sigma}\}_{t\le\tau})$ and $\boldsymbol{m}(\gamma,\tau) = \kappa(\{Q(\hat{\boldsymbol{X}}(t), \mathcal{D}(t))_{\gamma}\}_{t\le\tau})$. From definition 2, TSWL equivalence implies colors $c_{\sigma}(t) = c_{\gamma}(t)$ for all $t \le \tau$.

From Lemma 2 it follows that if stable colors are equal $c_{\sigma}(t) = c_{\gamma}(t)$ then embeddings generated by maximally expressive Q are also equal, i.e., $Q(\hat{\boldsymbol{X}}(t), \mathcal{D}(t))_{\sigma} = Q(\hat{\boldsymbol{X}}(t), \mathcal{D}(t))_{\gamma}, \quad \forall t \le \tau$. Therefore, the sequence $(Q(\hat{\boldsymbol{X}}(t), \mathcal{D}(t))_{\sigma})_{t\le\tau}$ and $(Q(\hat{\boldsymbol{X}}(t), \mathcal{D}(t))_{\gamma})_{t\le\tau}$ are also equal. For any arbitrary function $\kappa : \bigcup_{k=1}^{\infty}(\mathbb{R})^k \mapsto \mathbb{R}^D$ we have

$$\kappa((Q(\hat{\boldsymbol{X}}(t), \mathcal{D}(t))_{\sigma})_{t\le\tau}) = \kappa((Q(\hat{\boldsymbol{X}}(t), \mathcal{D}(t))_{\gamma})_{t\le\tau}),$$

thus $\boldsymbol{m}(\sigma,\tau) = \boldsymbol{m}(\gamma,\tau)$ if $\sigma \equiv_{TSWL} \gamma$. □

This proves that TSN is invariant to TSWL equivalence for any arbitrary function κ

E.2 Proof of proposition 1

In this section, we will prove the invariance of TSWL test to TSC isomorphism, formally if TSCs $\mathcal{D}(t)$ and $\mathcal{D}'(t)$ are isomorphic under bijective map on node set $f : \mathbb{V} \to \mathbb{V}'$, then $\{f(u) \mid u \in \sigma\} = \{u' \mid u' \in \sigma'\} \implies \sigma \equiv_{TSWL} \sigma'$.

Proof. Let $\mathcal{D}(t_i)$ and $\mathcal{D}'(t_i)$ be two isomorphic TSCs via the node bijection $f : \mathbb{V} \to \mathbb{V}'$, denoted as $\mathcal{D}(t_i) \approx \mathcal{D}'(t_i)$. We aim to show that for any simplex $\sigma(t_j) \in \mathcal{D}(t_i)$ and its image $\sigma'(t_j) = f(\sigma(t_j)) \in \mathcal{D}'(t_i)$, we have $\sigma \equiv_{\text{TSWL}} \sigma'$.

By definition of attributed TSC isomorphism, the bijection f satisfies two properties for all $t_j \leq t_i$:

$$\sigma(t_j) \in \mathcal{D}(t_i) \iff f(\sigma(t_j)) \in \mathcal{D}'(t_i), \tag{47}$$

$$\boldsymbol{x}(\sigma, t_j) = \boldsymbol{x}'(f(\sigma), t_j). \tag{48}$$

We prove by induction on the refinement iteration $\ell \geq 0$ that $c_\sigma^{(\ell)}(t_j) = c'^{(\ell)}_{f(\sigma)}(t_j)$ for all timestamps $t_j \leq t_i$.

Base case ($\ell = 0$). By feature preservation equation 48, corresponding simplices have identical feature vectors: $\boldsymbol{x}(\sigma, t_j) = \boldsymbol{x}'(f(\sigma), t_j)$. Because the initial coloring mapping $c^{(0)} : \mathcal{X} \to \mathcal{C}$ is a deterministic function, identical features map to identical initial colors:

$$c_\sigma^{(0)}(t_j) = c^{(0)}(\boldsymbol{x}(\sigma, t_j)) = c^{(0)}(\boldsymbol{x}'(f(\sigma), t_j)) = c'^{(0)}_{f(\sigma)}(t_j).$$

Inductive Step ($\ell \to \ell+1$): Assume that $c_\gamma^{(\ell)}(t_j) = c'^{(\ell)}_{f(\gamma)}(t_j)$ holds for all corresponding simplices at iteration ℓ.

Because f is a structural bijection preserving sub-simplex inclusion ($\gamma \prec \sigma \iff f(\gamma) \prec f(\sigma)$) and timestamps by equation 47, f induces an exact bijection between the temporal neighborhoods of σ and $f(\sigma)$:

$$\mathbb{N}_\star(f(\sigma), t_i) = \{(f(\gamma), t_j) \mid (\gamma, t_j) \in \mathbb{N}_\star(\sigma, t_i)\}, \quad \text{for } \star \in \{\uparrow, \downarrow, \mathbb{B}, \mathbb{C}\}.$$

Applying the induction hypothesis to the constituent colors in these neighborhoods yields identical input multisets to the HASH function:

$$c^{(\ell)}_{\sigma,\star}(t_i) = c'^{(\ell)}_{f(\sigma),\star}(t_i), \quad \forall \star \in \{\uparrow, \downarrow, \mathbb{B}, \mathbb{C}\}.$$

Since the HASH function is deterministic, identical inputs produce identical updated colors at iteration $\ell + 1$:

$$c_\sigma^{(\ell+1)}(t_j) = c'^{(\ell+1)}_{f(\sigma)}(t_j).$$

By induction, $c_\sigma^{(\ell)}(t_j) = c'^{(\ell)}_{f(\sigma)}(t_j)$ holds for all $\ell \geq 0$. Upon convergence at iteration L, the stable colors satisfy $c_\sigma(t_j) = c'_{f(\sigma)}(t_j)$ for all $t_j \leq t_i$, proving that $\sigma \equiv_{\text{TSWL}} f(\sigma)$. □

E.3 Proof of Theorem 1

Lemma 3 proves the invariance of TSN to TSWL equivalence for maximally expressive Q and arbitrary κ, to prove the maximal expressiveness of TSN we need to prove the reverse as well, that is, if TSN embedding are same for two simplices then these simplices are TSWL equivalent.

Proof. Suppose the initial encoder ϕ, the message functions, aggregation operators, update function U, and trajectory function $\kappa : \bigcup_{k=1}^{\infty}(\mathbb{R})^k : \mathbb{R}^D$ are injective. We will show that under these condition TSN is exactly as expressive as TSWL, i.e., $\sigma \equiv_{\text{TSWL}} \gamma \iff \boldsymbol{m}(\sigma, \tau) = \boldsymbol{m}(\gamma, \tau)$.

The forward implication,

$$\sigma \equiv_{\text{TSWL}} \gamma \implies \boldsymbol{m}(\sigma, \tau) = \boldsymbol{m}(\gamma, \tau),$$

follows directly from Lemma 3. It therefore remains to prove the reverse implication.

By Lemma 2,

$$\boldsymbol{h}_{\sigma}^{(L)}(t_i) = \boldsymbol{h}_{\gamma}^{(L)}(t_i) \iff c_{\sigma}(t_i) = c_{\gamma}(t_i), \qquad \forall\, t_i \leq \tau.$$

Hence,

$$\{\boldsymbol{h}_{\sigma}^{(L)}(t_i)\}_{t_i \leq \tau} = \{\boldsymbol{h}_{\gamma}^{(L)}(t_i)\}_{t_i \leq \tau} \iff \{c_{\sigma}(t_i)\}_{t_i \leq \tau} = \{c_{\gamma}(t_i)\}_{t_i \leq \tau}. \tag{49}$$

Since the trajectory function κ is injective,

$$\boldsymbol{m}(\sigma, \tau) = \kappa\Big(\{\boldsymbol{h}_{\sigma}^{(L)}(t_i)\}_{t_i \leq \tau}\Big), \tag{50}$$

$$\boldsymbol{m}(\gamma, \tau) = \kappa\Big(\{\boldsymbol{h}_{\gamma}^{(L)}(t_i)\}_{t_i \leq \tau}\Big), \tag{51}$$

satisfy

$$\boldsymbol{m}(\sigma, \tau) = \boldsymbol{m}(\gamma, \tau) \iff \{\boldsymbol{h}_{\sigma}^{(L)}(t_i)\}_{t_i \leq \tau} = \{\boldsymbol{h}_{\gamma}^{(L)}(t_i)\}_{t_i \leq \tau}.$$

Combining the above with Lemma 2 gives

$$\boldsymbol{m}(\sigma, \tau) = \boldsymbol{m}(\gamma, \tau) \iff \{c_{\sigma}(t_i)\}_{t_i \leq \tau} = \{c_{\gamma}(t_i)\}_{t_i \leq \tau}.$$

By Definition 2, equality of the stable TSWL color trajectories is equivalent to

$$\sigma \equiv_{\mathrm{TSWL}} \gamma.$$

Therefore,

$$\sigma \equiv_{\mathrm{TSWL}} \gamma \iff \boldsymbol{m}(\sigma, \tau) = \boldsymbol{m}(\gamma, \tau),$$

which proves that TSN is exactly as expressive as TSWL, and proves Theorem 1. □

E.4 Proof of Theorem 2

In this section we prove the relation of memory dimension in TS^{3}M and model expressivity for memory-less features, i.e., $\rho = 0$.

Part A.

Proof. Let $\boldsymbol{g}_D = [g_0, \ldots, g_{D-1}]$ and $\boldsymbol{g}_{D'} = [g_0, \ldots, g_{D'-1}]$ be two nested polynomial bases from the same polynomial family, with $D' > D$. For a simplex σ, $Q(X(t), D(t))_{\sigma}$ are its structural embedding. The corresponding ERP representations are the polynomial projection coefficients

$$\boldsymbol{m}(\sigma, \tau) = \int_0^{\tau} Q(X(t), \mathcal{D}(t))_{\sigma} \boldsymbol{g}_D(t)^{\top} \mu(t) dt, \tag{52}$$

$$\boldsymbol{m}'(\sigma, \tau) = \int_0^{\tau} Q(X(t), \mathcal{D}(t))_{\sigma} \boldsymbol{g}_{D'}(t)^{\top} \mu(t) \mathrm{d}t \tag{53}$$

$$\tag{54}$$

Where, each component of vector $\boldsymbol{m}(\sigma, \tau)$ is given as

$$[\boldsymbol{m}(\sigma, \tau)]_i = \int_0^{\tau} Q(X(t), \mathcal{D}(t))_{\sigma} [\boldsymbol{g}_D(t)]_i \mu(t) dt, \tag{55}$$

thus, if $[\boldsymbol{g}_D(t)]_i = [\boldsymbol{g}_{D'}(t)]_j \implies [\boldsymbol{m}(\sigma, \tau)]_i = [\boldsymbol{m}'(\sigma, \tau)]_j$.

Using the above relation and the fact that $\boldsymbol{g}_D = \boldsymbol{g}_{D'}[:D]$ has the first D polynomials from bases $\boldsymbol{g}_{D'}$ we conclude that the first D coordinates of $\boldsymbol{m}'(\sigma, \tau)$,are exactly those of $\boldsymbol{m}(\sigma, \tau)$. Therefore, if $\boldsymbol{m}'(\sigma, \tau) = \boldsymbol{m}'(\gamma, \tau)$, then $\boldsymbol{m}(\sigma, \tau) = \boldsymbol{m}(\gamma, \tau)$. Thus, any pair of embedding trajectories distinguished by the D-dimensional representation is also distinguished by the D'-dimensional representation, so $\boldsymbol{m}'$ is at least as expressive as $\boldsymbol{m}$. □

This proves that model with state dimension $D' > D$ are at-least as expressive as models with D.

Part B.

Proof. Let $\sigma, \gamma \in S_{\tau}$ be two simplex in $\mathcal{D}(\tau)$, and $\boldsymbol{m}(\sigma, \tau)$ and $\boldsymbol{m}(\gamma, \tau)$ be there respective state vector. Let $\{Q(X(t), \mathcal{D}(t))_{\sigma}\}_{t \leq \tau}$ and $\{Q(X(t), \mathcal{D}(t)_{\gamma}\}_{t \leq \tau}$ be the embedding trajectories, such that

CONDITION 1. $Q(X(t), \mathcal{D}(t))_\sigma = Q(X(t), \mathcal{D}(t))_\gamma \quad \forall t \leq \tau.$
Under the condition that both of the simplices have same trajectory over maximally expressive encoder, we have

$$\boldsymbol{m}(\sigma, \tau) = \int_0^\tau Q(X(t), \mathcal{D}(t))_\sigma \boldsymbol{g}_D(t)^\top \mu(t)\, dt \tag{56}$$

$$\boldsymbol{m}(\gamma, \tau) = \int_0^\tau Q(X(t), \mathcal{D}(t))_\gamma \boldsymbol{g}_D(t)^\top \mu(t)\, dt \tag{57}$$

$$\boldsymbol{m}(\sigma, \tau) - \boldsymbol{m}(\gamma, \tau) = \int_0^\tau (Q(X(t), \mathcal{D}(t))_\sigma - Q(X(t), \mathcal{D}(t))_\gamma) \boldsymbol{g}_D(t)^\top \mu(t)\, dt \tag{58}$$

$$\tag{59}$$

Since $Q(X(t), \mathcal{D}(t))_\sigma = Q(X(t), \mathcal{D}(t)_\gamma$ (from condition 1)

$$\boldsymbol{m}(\sigma, \tau) - \boldsymbol{m}(\gamma, \tau) = \mathbf{0} \tag{60}$$

$$\boldsymbol{m}(\sigma, \tau) = \boldsymbol{m}(\gamma, \tau) \tag{61}$$

Therefore for any D $Q(X(t), \mathcal{D}(t))_\sigma = Q(X(t), \mathcal{D}(t))_\gamma \quad \forall t \leq \tau \implies \boldsymbol{m}(\sigma, \tau) = \boldsymbol{m}(\gamma, \tau)$.

CONDITION 2. $Q(X(t), \mathcal{D}(t))_\sigma \neq Q(X(t), \mathcal{D}(t))_\gamma \quad \exists\, t \leq \tau.$
Let $R_{\sigma,\gamma}(t) = Q(X(t), \mathcal{D}(t))_\sigma - Q(X(t), \mathcal{D}(t))_\gamma$ denote the difference between the embedding trajectories of simplices σ and γ.We assume $R_{\sigma,\gamma}(t) \in L^2([0, \tau], \mu)$ be square integrable wrt. measure μ. Let $\mathcal{V}_D = \text{span}\{[\boldsymbol{g}_D]_0, [\boldsymbol{g}_D]_1, \ldots, [\boldsymbol{g}_D]_{D-1}, \}$ be the vector span by these basis.

The difference in there embedding is given as

$$\boldsymbol{m}(\sigma, \tau) - \boldsymbol{m}(\gamma, \tau) = \int_0^\tau (Q(X(t), \mathcal{D}(t))_\sigma - Q(X(t), \mathcal{D}(t))_\gamma) \boldsymbol{g}_D(t)^\top \mu(t)\, dt \tag{62}$$

$$\boldsymbol{m}(\sigma, \tau) - \boldsymbol{m}(\gamma, \tau) = \int_0^\tau R_{\sigma,\gamma}(t) \boldsymbol{g}_D(t)^\top \mu(t)\, dt \tag{63}$$

$$R_{\sigma,\gamma}(t) \in \mathcal{V}_D^\perp \implies \boldsymbol{m}(\sigma, \tau) = \boldsymbol{m}(\gamma, \tau). \tag{64}$$

Hence, two distinct embedding trajectories can remain indistinguishable under a finite-dimensional ERP representation when their difference lies in the orthogonal complement of the polynomial subspace $\mathcal{V}_D$.

When the state dimension is increased from D to D', where $D' > D$, the corresponding polynomial subspaces satisfy $\mathcal{V}_D \subset \mathcal{V}_{D'}$, and consequently $\mathcal{V}_{D'}^\perp \subset \mathcal{V}_D^\perp$.

Thus, increasing the state dimension reduces the subspace of trajectories that can be indistinguishable by ERP. If the polynomial family is complete in $L^2([0, \tau], \mu)$, then $\bigcap_{D=1}^\infty \mathcal{V}_D^\perp = \{\mathbf{0}\}$.

Consequently, in the infinite-dimensional limit, $\boldsymbol{m}(\sigma, \tau) = \boldsymbol{m}(\gamma, \tau) \iff R_{\sigma,\gamma} = \mathbf{0} \in L^2([0, \tau], \mu)$
When the state dimension is increased from D to D', st $D' > D$, we get $\mathcal{V}_{D'}^\perp \subset \mathcal{V}_D^\perp$, and as $D \to \infty$, $\mathcal{V}_D^\perp = \mathbf{0}$, and under these condition, $\boldsymbol{m}(\sigma, \tau) = \boldsymbol{m}(\gamma, \tau) \iff Q(X(t), \mathcal{D}(t))_\sigma = Q(X(t), \mathcal{D}(t))_\gamma$. □

E.5 PROOF OF THEOREM 3

Proof. For a general simplicial encoder $Q(\hat{\boldsymbol{X}}(t), \mathcal{D}(t)) \in \mathbb{R}^{N(\tau)\times 1}$, polynomial basis $\boldsymbol{g}_\tau(t) \in \mathbb{R}^{\mathrm{d}\times 1}$ and weight function $\mu_\tau(t) \in \mathbb{R}$, both indexed by τ to make their dependence on τ explicit through shifting and scaling and coefficient matrix $\boldsymbol{M}(\tau) \in \mathbb{R}^{N(\tau)\times D}$, the ERP reconstruction objective over the observed interval $[0, \tau]$ is

$$\min_{\boldsymbol{M}(\tau)} \int_0^\tau \left\| Q(\hat{\boldsymbol{X}}(t), \mathcal{D}(t)) - \boldsymbol{M}(\tau)\boldsymbol{g}_\tau(t) \right\|_F^2 \mu_\tau(t)\, \mathrm{d}t \tag{65}$$

Static optimum. Writing the objective via the trace identity $\|\boldsymbol{y}\|_2^2 = \mathrm{trace}(\boldsymbol{y}^\top \boldsymbol{y})$ and differentiating with respect to $\boldsymbol{M}(\tau)$ gives

$$\frac{\mathrm{d}}{\mathrm{d}\boldsymbol{M}(\tau)} \int_0^\tau \left\| Q(\hat{\boldsymbol{X}}(t), \mathcal{D}(t)) - \boldsymbol{M}(\tau)\boldsymbol{g}_\tau(t) \right\|_F^2 \mu_\tau(t)\, dt \tag{66}$$

$$\implies \int_0^\tau 2\Big(\boldsymbol{M}(\tau)\boldsymbol{g}_\tau(t)\boldsymbol{g}_\tau(t)^\top - Q(\hat{\boldsymbol{X}}(t), \mathcal{D}(t))\boldsymbol{g}_\tau(t)^\top \Big) \mu_\tau(t)\, \mathrm{d}t \tag{67}$$

Setting this to $\mathbf{0}$ and factoring out $\boldsymbol{M}(\tau)$(recall $\boldsymbol{M}(\tau)$ does not depend on the integration variable t) gives

$$\boldsymbol{M}(\tau) \int_0^\tau \boldsymbol{g}_\tau(t)\boldsymbol{g}_\tau(t)^\top \mu_\tau(t)\, dt = \int_0^\tau Q(\hat{\boldsymbol{X}}(t), \mathcal{D}(t))\boldsymbol{g}_\tau(t)^\top \mu_\tau(t)\, \mathrm{d}t \tag{68}$$

For any orthonormal polynomial–measure pair $(\boldsymbol{g}_\tau, \mu_\tau)$ as characterized in (Gu et al., 2020), $\int_0^\tau \boldsymbol{g}_\tau(t)\boldsymbol{g}_\tau(t)^\top \mu_\tau(t)\, dt = \boldsymbol{I}_D$, so the unique stationary point is

$$\boldsymbol{M}(\tau) = \int_0^\tau Q(\hat{\boldsymbol{X}}(t), \mathcal{D}(t))\boldsymbol{g}_\tau(t)^\top \mu_\tau(t)\, dt, \tag{69}$$

i.e., $\boldsymbol{M}(\tau)$ is precisely the coefficient matrix of the $L^2(\mu_\tau)$-projection of Q onto the degree-$(D{-}1)$ polynomial subspace. □

E.6 Continuous Evolution under Translated Legendre (LegT)

For the translated Legendre (LegT) measure with fixed window T, the measure and orthonormal basis are

$$\mu_\tau(t) = \frac{1}{T}\, \mathbb{I}_{[\tau - T, \tau]}(t), \qquad \boldsymbol{g}_\tau(t)[n] = \sqrt{2n+1}\, P_n\!\left(\frac{2(t-\tau)}{T} + 1 \right), \tag{70}$$

with P_n the n-th Legendre polynomial. Differentiating the measure with respect to τ gives

$$\frac{\partial}{\partial \tau} \mu_\tau(t) = \frac{1}{T}\big(\delta_\tau(t) - \delta_{\tau - T}(t) \big), \tag{71}$$

and differentiating the basis, using the Legendre recurrence $(z+1)P_n'(z) = nP_n(z) + (2n-1)P_{n-1}(z) + (2n-3)P_{n-2}(z) + \dots$ (Gu et al., 2020), gives

$$\frac{\partial}{\partial \tau} \boldsymbol{g}_\tau(t)[n] = -\frac{2}{T}\sqrt{2n+1} \Big[(2n-1)^{1/2} \boldsymbol{g}_\tau(t)[n-1] + (2n-3)^{1/2} \boldsymbol{g}_\tau(t)[n-2] + \dots \Big]. \tag{72}$$

Unlike LegS, the LegT window has *two* moving boundaries (τ and $\tau - T$), so differentiating $\boldsymbol{M}(\tau)[n] = \int_{\tau-T}^{\tau} Q(\hat{\boldsymbol{X}}(t), \mathcal{D}(t))\, \boldsymbol{g}_\tau(t)[n]\, \mu_\tau(t)\, dt$ via Leibniz's rule picks up boundary terms at both endpoints:

$$\begin{aligned} \frac{\mathrm{d}}{\mathrm{d}\tau} \boldsymbol{M}(\tau)[n] = &\int_{\tau-T}^{\tau} Q(\hat{\boldsymbol{X}}(t), \mathcal{D}(t))\, \frac{\partial \boldsymbol{g}_\tau(t)[n]}{\partial \tau}\, \mu_\tau(t)\, dt \\ &+ \frac{1}{T} Q(\hat{\boldsymbol{X}}(\tau), \mathcal{D}(\tau))\, \boldsymbol{g}_\tau(\tau)[n] - \frac{1}{T} Q(\hat{\boldsymbol{X}}(\tau - T), \mathcal{D}(\tau - T))\, \boldsymbol{g}_\tau(\tau - T)[n]. \end{aligned} \tag{73}$$

Using $P_n(1) = 1$ and $P_n(-1) = (-1)^n$, the boundary basis values are $\boldsymbol{g}_\tau(\tau)[n] = \sqrt{2n+1}$ and $\boldsymbol{g}_\tau(\tau - T)[n] = \sqrt{2n+1}(-1)^n$.

Reconstruction approximation. The trailing-edge term requires $Q(\hat{\boldsymbol{X}}(\tau - T), \mathcal{D}(\tau - T))$, the encoder value at a point that has already left the observation window and is no longer directly accessible online. Following (Gu et al., 2020), we approximate it using the current reconstruction:

$$Q(\hat{\boldsymbol{X}}(\tau - T), \mathcal{D}(\tau - T)) \approx \sum_{k=0}^{D-1} \boldsymbol{M}(\tau)[i] \sqrt{2k+1} (-1)^k, \tag{74}$$

i.e., we assume the current coefficients $\boldsymbol{M}(\tau)$ already compress the history accurately enough to recover the function value at the trailing edge.

Substituting this approximation and collecting terms yields

$$\frac{\mathrm{d}\boldsymbol{M}(\tau)}{\mathrm{d}\tau} = -\frac{1}{T}\boldsymbol{M}(\tau)\boldsymbol{A}^\top + \frac{1}{T}Q(\hat{\boldsymbol{X}}(\tau), \mathcal{D}(\tau))\boldsymbol{B}^\top, \tag{75}$$

$$\boldsymbol{A}[n,k] = \sqrt{(2n+1)(2k+1)}\begin{cases}1 & k \le n\\ (-1)^{n-k} & k \ge n\end{cases}, \qquad \boldsymbol{B}[n] = \sqrt{2n+1}. \tag{76}$$

Similar derivation can be done for other polynomials.

F Discretization

In this section, we provide the exact discretization of continuous update equation for linear encoder Q and approximate discretization for non-linear Q. Through ERP, we derived the continuous-time state evolution equation equation 11. Since data in a TSC arrives as timestamped events $e(t_i)$, implementing this equation requires a discrete-time recurrence for $\boldsymbol{M}[i] := \boldsymbol{M}(t_i)$ over a timestamps in $\mathbb{T}$.

Because equation 11 is generally nonlinear in $\boldsymbol{M}$ in general, the appropriate discretization depends on the choice of Q. When Q is linear in $\boldsymbol{M}$, equation 11 admits an exact discretization via vectorization and a zero-order-hold (ZOH), Theorem 5 given below.

Theorem 5 (Exact ZOH discretization for linear Q)**.** *Let $\boldsymbol{M}(\tau)$ satisfy equation 11 on $[t_i, t_{i+1})$ with $Q_m(\boldsymbol{M}(\tau), \mathcal{D}(\tau)) = \boldsymbol{P}_{c,i}\boldsymbol{M}(\tau)\boldsymbol{u}$ linear in $\boldsymbol{M}$, and with $Q_x(\boldsymbol{X}(\tau), \mathcal{D}(\tau)) = \boldsymbol{P}_{x,i}\boldsymbol{X}[i]$ under ZOH. Define*

$$\hat{\boldsymbol{A}}_i := \boldsymbol{u}\boldsymbol{B}^\top \otimes \boldsymbol{P}_{c,i} - \boldsymbol{A} \otimes \boldsymbol{I}_{N(t_i)}, \quad \boldsymbol{x}_i := \mathrm{vec}\left(\boldsymbol{P}_{x,i}\boldsymbol{X}[i]\frac{\boldsymbol{B}^\top}{T}\right), \quad \bar{\boldsymbol{A}}_i := \exp\left(\frac{\Delta_i}{T}\hat{\boldsymbol{A}}_i\right),$$

where $\Delta_i = t_{i+1} - t_i$. Then state update for $\boldsymbol{M}[i+1]$ is given by the recurrence

$$\mathrm{vec}(\boldsymbol{M}[i+1]) = \bar{\boldsymbol{A}}_i\,\mathrm{vec}(\boldsymbol{M}[i]) \;+\; \alpha\hat{\boldsymbol{A}}_i^{-1}\big(\bar{\boldsymbol{A}}_i - \boldsymbol{I}\big)\boldsymbol{x}_i, \text{ and } \boldsymbol{M}[i+1] = \mathrm{unvec}\big(\mathrm{vec}(\boldsymbol{M}[i+1])\big). \tag{77}$$

Proof. Substituting $Q_m(\boldsymbol{M}(\tau), \mathcal{D}(\tau)) = \boldsymbol{P}_{c,k}\boldsymbol{M}(\tau)$ and the zero-hold input $Q_x(\boldsymbol{X}(\tau), \mathcal{D}(\tau)) = \boldsymbol{P}_{x,k}\boldsymbol{X}_k$ into Eq. equation 11 on $[t_i, t_{i+1})$, with normalization by the fixed constant T, gives

$$\frac{\mathrm{d}\boldsymbol{M}(\tau)}{\mathrm{d}\tau} = -\frac{1}{T}\boldsymbol{M}(\tau)\boldsymbol{A}^\top + \frac{1}{T}\boldsymbol{P}_{c,k}\boldsymbol{M}(\tau)\boldsymbol{B}^\top + \frac{1}{T}\boldsymbol{P}_{x,k}\boldsymbol{X}_k\boldsymbol{B}^\top. \tag{78}$$

Here $\boldsymbol{M}(\tau)$ is multiplied on the left by $\boldsymbol{P}_{c,k}$ and on the right by $\boldsymbol{A}^\top, \boldsymbol{B}^\top$, so equation 78 can not be discretized by a one-sided matrix exponential in general; we linearize it via vectorization. Using the identity $\mathrm{vec}(\boldsymbol{L}\boldsymbol{Y}\boldsymbol{R}^\top) = (\boldsymbol{R} \otimes \boldsymbol{L})\mathrm{vec}(\boldsymbol{Y})$ for matrices $\boldsymbol{L}, \boldsymbol{Y}, \boldsymbol{R}$, and letting $\boldsymbol{c}(\tau) := \mathrm{vec}(\boldsymbol{M}(\tau))$, $\boldsymbol{x}_k := \mathrm{vec}(\boldsymbol{P}_{x,k}\boldsymbol{X}_k\boldsymbol{B}^\top)$:

$$\mathrm{vec}\big(\boldsymbol{M}(\tau)\boldsymbol{A}^\top\big) = (\boldsymbol{A} \otimes \boldsymbol{I}_{N(\tau)})\,\boldsymbol{c}(\tau), \qquad \mathrm{vec}\big(\boldsymbol{P}_{c,k}\boldsymbol{M}(\tau)\boldsymbol{B}^\top\big) = (\boldsymbol{B}^\top \otimes \boldsymbol{P}_{c,k})\,\boldsymbol{c}(\tau). \tag{79}$$

Substituting into equation 78 and vectorizing both sides yields the constant-coefficient linear system

$$\frac{\mathrm{d}\boldsymbol{c}(\tau)}{\mathrm{d}\tau} = \frac{1}{T}\boldsymbol{M}[i]\,\boldsymbol{c}(\tau) + \frac{1}{T}\boldsymbol{x}_i, \qquad \boldsymbol{M}[i] := \boldsymbol{B}_c \otimes \boldsymbol{P}_{c,i} - \boldsymbol{A} \otimes \boldsymbol{I}_{N(\tau)}. \tag{80}$$

Since $\boldsymbol{M}[i]$ and $\boldsymbol{x}_i$ are constant on $[t_i, t_{i+1})$, equation 80 is a linear ODE with constant coefficients. Applying the integrating factor $\exp\big(-\frac{\tau}{T}\boldsymbol{M}[i]\big)$ and integrating from t_i to τ gives, for $\tau \in [t_i, t_{i+1})$,

$$\boldsymbol{c}(\tau) = \exp\left(\frac{\tau - t_i}{T}\boldsymbol{M}[i]\right)\boldsymbol{c}(t_i) + \int_{t_i}^{\tau}\exp\left(\frac{\tau - s}{T}\boldsymbol{M}[i]\right)\frac{1}{T}\boldsymbol{x}_i\,ds. \tag{81}$$

Evaluating at $\tau = t_{i+1}$ and writing $\Delta_i = t_{i+1} - t_i$, the integral evaluates in closed form when $\boldsymbol{M}[i]$ is invertible:

$$\frac{1}{T}\int_{t_i}^{t_{i+1}}\exp\left(\frac{t_{i+1} - s}{T}\boldsymbol{M}[i]\right)ds\;\boldsymbol{x}_i = \boldsymbol{M}[i]^{-1}\left(\exp\left(\frac{\Delta_i}{T}\boldsymbol{M}[i]\right) - \boldsymbol{I}\right)\boldsymbol{x}_i, \tag{82}$$

which gives

$$\mathrm{vec}(\boldsymbol{M}[i+1]) = \bar{\boldsymbol{M}}_i \,\mathrm{vec}(\boldsymbol{M}[i]) + \boldsymbol{M}[i]^{-1}\big(\bar{\boldsymbol{M}}_k - \boldsymbol{I}\big)\boldsymbol{x}_i, \qquad \bar{\boldsymbol{M}}_i := \exp\!\left(\frac{\Delta_i}{T}\boldsymbol{M}[i]\right), \quad (83)$$

as claimed. The state matrix $\boldsymbol{M}[i+1] = \mathrm{unvec}(\mathrm{vec}(\boldsymbol{c}_{i+1})$. □

Given output of encoder Q, the computation complexity of equation 77 is $\mathcal{O}(N(t_i)^3 D^3)$, where $N(t_i) = |\mathbb{S}(t_i)|$, is the total number of unique simplices.

For general nonlinear Q, we derive an approximate discretization under a piecewise-constant assumption $Q_m(\boldsymbol{M}(\tau), \mathcal{D}(\tau)) \approx Q(\boldsymbol{M}(t_i), \mathcal{D}(t_i))$ for $\tau \in [t_i, t_{i+1})$ present it in Theorem 6.

Theorem 6 (ZOH-approximate discretization for nonlinear h)**.** *Let $\boldsymbol{M}(\tau)$ satisfy equation 11 on $[t_i, t_{i+1})$ with Q_m, Q_x possibly nonlinear in $\boldsymbol{M}, \boldsymbol{X}$. Adopt the piecewise-constant approximation along with ZOH on input as $Q_m(\boldsymbol{M}(\tau), \mathcal{D}(\tau)) \approx Q_m(\boldsymbol{M}[i], \mathcal{D}(t_i))$, $\boldsymbol{X}(\tau) = \boldsymbol{X}[i]$ and $\tau \in [t_i, t_{i+1})$. Define the frozen aggregates $\boldsymbol{U}_{m,i} := Q_m(\boldsymbol{M}[i], \mathcal{D}(t_i))$, $\boldsymbol{U}_{x,i} := Q_x(\boldsymbol{X}[i], \mathcal{D}(t_i))$, and $\Delta_i := t_{i+1} - t_i$. Then, under this approximation, the state at t_{i+1} is given by the recurrence*

$$\boldsymbol{M}[i+1] = \boldsymbol{M}[i]\bar{\boldsymbol{A}}^\top \;+\; \rho\boldsymbol{U}_{m,i}\bar{\boldsymbol{B}}^\top + \boldsymbol{U}_{x,i}\bar{\boldsymbol{B}}^\top,$$

where $\bar{\boldsymbol{A}} = \exp\big(-\frac{\Delta_i}{T}\boldsymbol{A}^\top\big)$, and $\bar{\boldsymbol{B}} = \boldsymbol{B}(\boldsymbol{A}^\top)^{-1}\big(\boldsymbol{I} - \exp\big(-\frac{\Delta_i}{T}\boldsymbol{A}^\top\big)\big)$.

Proof. Substituting the piecewise-constant approximation $Q_m(\boldsymbol{M}(\tau), \mathcal{D}(\tau)) \approx Q_m(\boldsymbol{M}[i], \mathcal{D}(\tau))$, $\boldsymbol{X}(\tau) = \boldsymbol{X}[i]$ for $\tau \in [t_i, t_{i+1})$ into the update equation gives the frozen aggregates $\boldsymbol{U}_{m,i} := Q_m(\boldsymbol{M}[i], \mathcal{D}(t_i))$ and $\boldsymbol{U}_{x,i} := Q_x(\boldsymbol{X}[i], \mathcal{D}(t_i))$, which are constant on the interval. Substituting into Eq. equation 11, with normalization constant T, yields

$$\frac{\mathrm{d}\boldsymbol{M}(\tau)}{\mathrm{d}\tau} = -\frac{1}{T}\boldsymbol{M}(\tau)\boldsymbol{A}^\top + \frac{1}{T}\boldsymbol{U}_{m,i}\boldsymbol{B}^\top + \frac{1}{T}\boldsymbol{U}_{x,i}\boldsymbol{B}^\top = -\frac{1}{T}\boldsymbol{M}(\tau)\boldsymbol{A}^\top + \frac{1}{T}\boldsymbol{W}_i, \quad (84)$$

where $\boldsymbol{W}_i := \boldsymbol{U}_{m,i}\boldsymbol{B}^\top + \boldsymbol{U}_{x,i}\boldsymbol{B}^\top$ is constant on $[t_i, t_{i+1})$. Unlike the linear case of Theorem 5, $\boldsymbol{M}(\tau)$ here appears only right-multiplied by $\boldsymbol{A}^\top$, so equation 84 is a one-sided linear matrix ODE with constant coefficients and admits a closed-form solution directly, without vectorization.

Applying the integrating factor $\exp\big(\frac{\tau}{T}\boldsymbol{A}^\top\big)$ on the right of equation 84:

$$\frac{\mathrm{d}}{\mathrm{d}\tau}\Big[\boldsymbol{M}(\tau)\exp\Big(\frac{\tau}{T}\boldsymbol{A}^\top\Big)\Big] = \frac{1}{T}\boldsymbol{W}_i\exp\Big(\frac{\tau}{T}\boldsymbol{A}^\top\Big). \quad (85)$$

Integrating from t_i to t_{i+1} and right-multiplying by $\exp\big(-\frac{t_{i+1}}{T}\boldsymbol{A}^\top\big)$:

$$\boldsymbol{M}(t_{i+1}) = \boldsymbol{M}(t_i)\exp\Big(-\frac{\Delta_i}{T}\boldsymbol{A}^\top\Big) + \frac{1}{T}\boldsymbol{W}_i\int_{t_i}^{t_{i+1}}\exp\Big(\frac{t_{i+1}-s}{T}\boldsymbol{A}^\top\Big)\,ds, \quad (86)$$

with $\Delta_i = t_{i+1} - t_i$. When $\boldsymbol{A}$ is invertible, the integral evaluates in closed form:

$$\frac{1}{T}\int_{t_i}^{t_{i+1}}\exp\Big(\frac{t_{i+1}-s}{T}\boldsymbol{A}^\top\Big)\,ds = (\boldsymbol{A}^\top)^{-1}\Big(\boldsymbol{I} - \exp\Big(-\frac{\Delta_i}{T}\boldsymbol{A}^\top\Big)\Big), \quad (87)$$

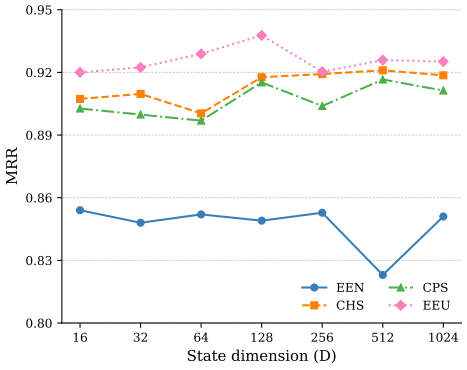
which gives

$$\boldsymbol{M}[i+1] = \boldsymbol{M}[i]\exp\Big(-\frac{\Delta_i}{T}\boldsymbol{A}^\top\Big) + \boldsymbol{W}_i(\boldsymbol{A}^\top)^{-1}\Big(\boldsymbol{I} - \exp\Big(-\frac{\Delta_i}{T}\boldsymbol{A}^\top\Big)\Big), \quad (88)$$

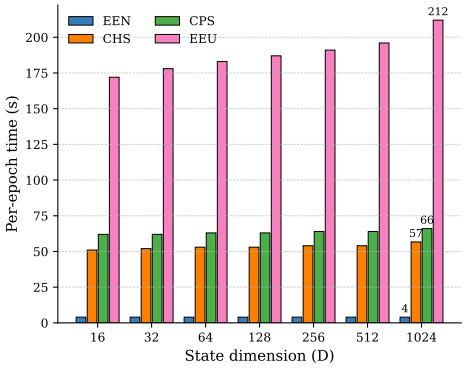
as claimed. □

Finally, when Q_m is linear in $\boldsymbol{M}$, i.e. $\boldsymbol{U}_{m,k} = \boldsymbol{P}_{c,k}\boldsymbol{M}[i]$ for some fixed $\boldsymbol{P}_{c,k}$, this recurrence agrees with Theorem 5 exactly in the special case $\rho = \mathbf{0}$; for general $\rho \neq \mathbf{0}$ the two differ because equation 84 here treats $Q_m(\boldsymbol{M}(\tau), \mathcal{D}(\tau))$ as frozen throughout $[t_i, t_{i+1})$, whereas Theorem 5 allows $\boldsymbol{M}(\tau)$ to evolve continuously inside Q_m over the same interval, which is exact only in the linear case.

Given the output of encoder Q the computation complexity approximate update in Theorem 6 is $\mathcal{O}(D^3 + N(t_i)D^2)$.



(a) MRR across increasing TS^{3}M state dimensions for higher-order link prediction.



(b) Per-epoch training time across state dimensions, demonstrating sublinear scaling as capacity expands from 16 to 1024.

Figure 9: Effect of increasing state dimension on TS^{3}M predictive performance and per-epoch training time.

G Simplicial Weisfeiler–Lehman (SWL) Test

We now describe the SWL color refinement procedure for a static simplicial complex.

Let $\mathcal{K}$ be a SC and let c^i_σ denote the color assigned to a simplex $\sigma \in \mathcal{K}$ at iteration i. The Simplicial Weisfeiler–Lehman (SWL) procedure initializes all simplices with the same color and iteratively refines the colors by aggregating structured adjacency information across dimensions.

For each $\tau \in \mathbb{N}_\uparrow(\sigma)$, let $\delta = \sigma \cup \tau$ denote the corresponding coface for upper adjacency. For each $\tau \in \mathbb{N}_\downarrow(\sigma)$, let $\delta = \sigma \cap \tau$ denote the common face for lower adjacency.

The color refinement rule is then given by

$$c^{i+1}_\sigma = \mathrm{HASH}\Big(c^i_\sigma,\ \{\!\{(c^i_\tau,\ c^i_\delta) : \tau, \delta \in \mathbb{N}_\uparrow(\sigma)\}\!\},\ \{\!\{(c^i_\tau,\ c^i_\delta) : \tau, \delta \in \mathbb{N}_\downarrow(\sigma)\}\!\}\Big), \tag{89}$$

where $\{\!\{\cdot\}\!\}$ denotes a multi-set and $\mathrm{HASH}(\cdot)$ is injective over its arguments.

The procedure terminates once the partition of simplices induced by color stabilizes. Two simplicial complexes are declared non-isomorphic if, at any iteration, the histograms of simplex colors differ at convergence. This formulation is expressively equivalent to the original SWL test proposed in Bodnar et al. (2021), since boundary and co-boundary information can be derived from the paired upper and lower adjacency relations.

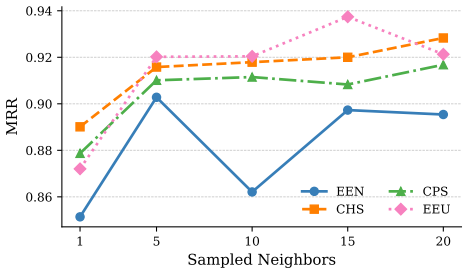
Definition 5 (WL Equivalence)**.** *Two simplices $\sigma, \sigma' \in \mathcal{K}$ are said to be* WL equivalent*, denoted $\sigma \sim_{\mathrm{WL}} \sigma'$, if and only if*

$$c_\sigma = c_{\sigma'}. \tag{90}$$

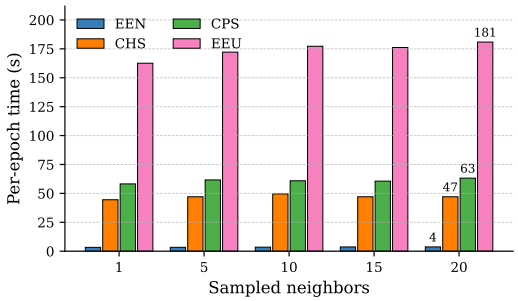
Analogously, two simplicial complexes $\mathcal{K}$ and $\mathcal{K}'$ are WL equivalent*, denoted $\mathcal{K} \sim_{\mathrm{WL}} \mathcal{K}'$, if and only if there exists a bijection between the simplices of the two complexes that preserves the partition induced by WL equivalence.*

H Ablation Studies

In this section, we conduct an ablation study of TS^{3}M to evaluate its predictive performance across four datasets: `CHS`, `CPS`, `EEN`, and EEU. Specifically, we analyze how varying the state dimension D, the number of sampled neighbors, and the memory coefficient α affects both the model's predictive accuracy and its per-epoch training time.



(a) MRR across increasing number of sampled neighbors for higher-order link prediction.



(b) Per-epoch training time across number of sampled neighbor.

Figure 10: Effect of increasing number of sampled neighbors on predictive performance and per-epoch training time.

State Dimension. We systematically evaluate the sensitivity of model performance and computational efficiency to varying state dimensions ($D \in \{16, 32, 64, 128, 256, 512, 1024\}$). Figure 9a illustrates the predictive performance measured by MRR against five negative samples. We observe a consistent performance gain up to $d = 128$, beyond which MRR degrades, likely due to over-parameterization and overfitting on sparse temporal interactions. Concurrently, Figure 9b demonstrates that training runtime scales sub-linearly with respect to state dimension. Expanding the state dimension by a factor of $64\times$ (from 16 to 1024) yields an average increase of only 10.22% in per-epoch execution time (ranging from 0.00% on Email-Enron to 23.26% on Email-Eu). These findings suggest that scaling model capacity offers favorable trade-offs between predictive accuracy and runtime overhead, with $d = 128$ serving as the optimal hyperparameter choice across all evaluated benchmarks.

Neighbor Sampling. For efficient scaling of large TSC, the equation 13 only updates the state of the simplices in local sampled neighborhood of the simplex observed in the event instead of the entire TSC. This sampling not only effects the accuracy of this update step but also the computation time required to calculate the update when new events are observed. In Figure 10 we illustrate the effects of increasing sampled neighbors on model predictive performance Figure 10a and per-epoch time Figure10b. Both the plots indicate a tradeoff where model performance generally increased by increasing number of sampled neighbors, while also increasing the per-epoch time. For increasing the neighbors sampled from $1 \mapsto 20$, we got an average 9.2% increase in per-epoch time for training the model.

Memory coefficient α. The memory coefficient ρ introduces a temporal bias into the encoder input Q. Specifically, it ensures that even when raw input features are identical across two simplices, their distinct memory representations can differentiate them—thereby enhancing model expressivity without increasing the depth of encoder Q. As shown in Figure 11, incorporating memory via $\rho > 0$ yields consistent gains in predictive performance (MRR) across three out of the four evaluated datasets (CHS, CPS, and EEU), whereas performance on EEN peaks at $\rho = 0$.

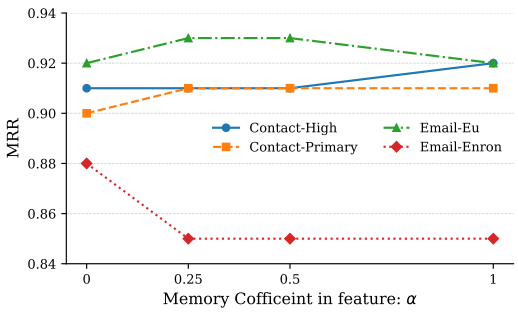


Figure 11: MRR for higher-order link prediction across varying memory coefficient values (α).

I Baseline and Datasets

In this section, we describe the baseline models and datasets used in the experiments, as well as the implementation and hardware details for the setup.

Table 2: Hyperparameter configurations used in the experiments.

LR	Batch Size	Epochs	Early Stopping	Train Seeds	Test Seed
1e-4	64	30	5	0, 1, 2, 3, 4	0

I.1 Experiment Setup

For all experiments, we follow the setup used by (Behrouz et al., 2023); Table 2 details the specific values used. All experiments were conducted on a single server equipped with an Intel Xeon Gold 5220R CPU (@ 2.20 GHz), 256 GB RAM, and dual NVIDIA RTX A6000 GPUs (48 GB VRAM each).

The hyperparameter settings used across datasets for model evaluation are detailed below. Table 3 specifies dataset-dependent parameters including boundary-simplex sampling size (N_{sub}), co-boundary simplex sampling size (N_{cof}), and scaling factors to scale down raw-timestamps values, alongside global model architecture dimensions for TS^{3}M. While no upper and lower adjacent samples were used during the experiments.

Table 3: Hyperparameter configurations for dataset sampling and model architecture.

Dataset	Times-Scale-down	N_{sub}	N_{cof}	Description
CB	1	26	30	
EEN	1	20	20	
EEU	1	26	20	
MQT	$10^3 \times 60$	25	21	
NDC	$10^3 \times 60$	25	21	
CHS	1	6	45	
CPS	1	6	17	
Shared Model Architecture Parameters				
State Dimension ($\text{dim}_{\text{state}}$)	128			
Time Embedding Dimension (dim_{time})	100			
Latent Dimension ($\text{dim}_{\text{latent}}$)	128			

I.2 Baseline

Baseline Descriptions Below are the details for the baseline models evaluated in our experiments:

1. **Cat-Walk** (Behrouz et al., 2023): An inductive temporal hypergraph learning framework that extracts higher-order dynamic patterns using temporal set walks (`SetWalk`), set-based anonymization, and an MLP-Mixer encoder. **FastHeP** (Xu et al., 2025): An efficient inductive temporal hypergraph learning framework decouples multi-hop structure learning and uses hybrid message passing.
2. **CHESHIRE** (Chen et al., 2023): A spectral hyperedge prediction framework. It initializes node embeddings using the hypergraph incidence matrix and treats hyperedges as cliques, using Chebyshev spectral graph convolutions to learn localized spatial features over the expansion.
3. **HyperSAGNN** (Zhang et al., 2019): A deep hypergraph model combining random-walk node initialization with a Spectral Aggregated Graph Neural Network (SAGNN) and mean pooling to derive hyperedge representations.
4. **NHP** (Yadati et al., 2020): Neural Hyperlink Predictor extends HyperSAGNN by generating initial embeddings via `Node2Vec` on the clique-expanded graph and applying a max-min pooling mechanism to adaptively weight node features.
5. **HPLSF** (Xu et al., 2013): A probabilistic model that leverages network homophily to construct hyperedge representations via latent feature learning and entropy-based optimization.

6. **HPRA** (Kumar et al., 2020): A shallow heuristic method based on resource allocation. It computes hypergraph resource allocation indices across node pairs in a candidate hyperedge based on shared local neighborhoods.
7. **CE-CAW**: Applies Causal Anonymous Walks (CAW) (Wang et al., 2022) to the clique expansion (CE) of the hypergraph to capture temporal dynamic patterns inductively.
8. **CE-EVOLVEGCN**: A discrete-time dynamic graph baseline that applies EVOLVEGCN (Pareja et al., 2020) over the hypergraph's clique expansion, using RNNs to evolve GCN parameters across time snapshots.
9. **CE-GCN**: Standard Graph Convolutional Networks (GCN) (Wu et al., 2019) applied to the clique expansion to learn node representations, followed by an MLP classifier for link prediction.
10. **CE-DYGFORMER**: DyGFormer (Yu et al., 2023), a transformer based memory dynamic graph model, designed to capture long range temporal dependencies. Applied to CE of hypergraph.
11. **HEURISTIC**: Various unsupervised co-occurrence and face counting methods Benson et al. (2018) used to score higher-order link formation in SC.

I.3 DATASET

In this section, we briefly describe the datasets used for model evaluation and present their statistics in Table 4. Moreover, we illustrate the distribution of higher-order events (k-simplices) observed in the event streams in Figure 12. Notably, the CHS and CPS datasets contain some of the highest proportions of pairwise interactions, accounting for approximately $\sim 92\%$ and $\sim 91\%$, respectively, of all interactions observed in their corresponding streams. This may explain why CE-graph models perform well on these datasets, as reflected in their HLP performance. In contrast, MQT, NDC, and CB contain approximately $\sim 50\%$, $\sim 58\%$, and $\sim 28\%$ dyadic interactions, respectively, placing them among the lowest-performing datasets for CE-graph models.

1. **`Contact High School`** (CHS) (Benson et al., 2018; Mastrandrea et al., 2015): A temporal contact network capturing face-to-face proximity interactions among students and teachers in a high school setting, recorded via wearable sensors at 20-second intervals.
2. **`Contact Primary School`** (CPS) (Benson et al., 2018; Stehlé et al., 2011): A proximity contact network collected in a primary school, structured similarly to `CHS` with nodes representing students and teachers.
3. **`Congress Bill`** (CB) (Benson et al., 2018; Fowler, 2006): A political co-sponsorship network where nodes represent US Congress members and hyperedges represent legislative bills introduced in the House and Senate, timestamped by introduction date.
4. **`Email Enron`** (EEN) (Benson et al., 2018): A communication network where nodes represent email addresses and hyperedges group the sender and all recipients of an email, recorded with millisecond resolution.
5. **`Email EU`** (EEU) (Benson et al., 2018; Yin et al., 2017): An email communication network from a European research institution, where hyperedges connect senders and recipients for each email at 1-second timestamp resolution.
6. **`Math StackExchange`** (MQT) (Benson et al., 2018): A QA forum dataset where nodes represent community tags and hyperedges correspond to the set of tags applied to individual questions on `math.stackexchange.com`.
7. **`NDC Classes`** (NDC) (Benson et al., 2018): A medical dataset where nodes represent drug class labels and hyperedges group class categories corresponding to individual drugs, timestamped by initial market entry date (in days).

Table 4: Dataset statistics.

Dataset	NDC Class	High School	Primary School	Congress Bill	Email Enron	Email Eu	Question Tags M
$\lvert\mathcal{V}\rvert$	1,161	327	242	1,718	143	998	1,629
$\lvert\mathcal{E}\rvert$	49,724	172,035	106,879	260,851	10,883	234,760	822,059
#Timestamps	5,891	7,375	3,100	5,936	10,788	232,816	822,054
Task	HeP	HeP & NC	HeP & NC	HeP	HeP	HeP	HeP

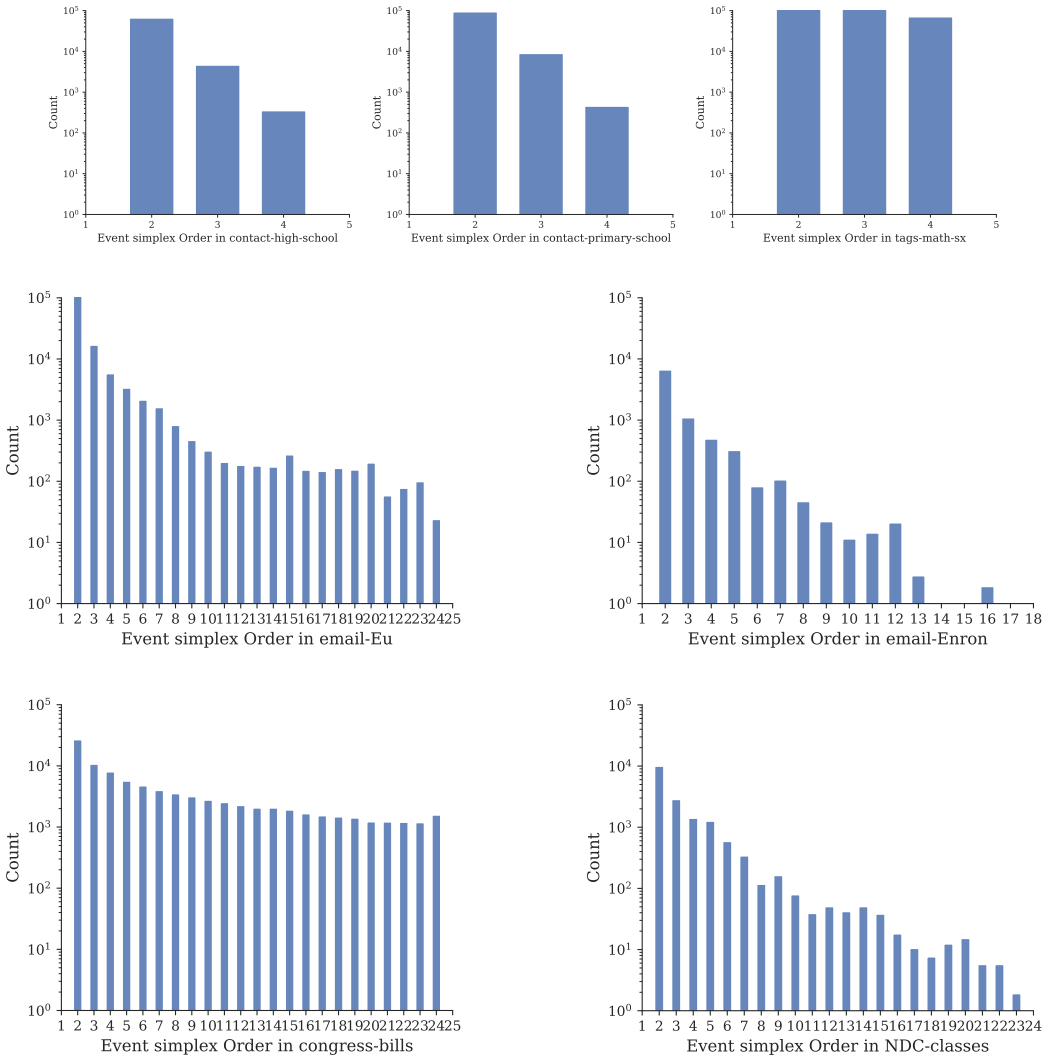


Figure 12: Count of k-simplices observed in the entire dataset stream. Different datasets depict varied distribution, where CHS, and CPS with highest proportion of pairwise interactions.