LEARNING ON DYNAMIC HYPERGRAPHS FROM INCIDENCE STREAMS

Anonymous authors
Paper under double-blind review

ABSTRACT

Hypergraphs model higher-order interactions, and in many systems their hyperedges persist: nodes join over time, carry attributes such as roles and may leave. Dynamic hypergraph learning addresses separate tasks on transient hyperedges, while persistent hyperedges are reduced to their star expansion for temporal graph networks. Neither line characterizes what an evaluation protocol reveals beyond the history or what a model must retain. A unified formulation of all tasks would make both questions precise. We represent a dynamic hypergraph as a stream of incidence records and factorize its likelihood, so that every task is a conditional of one factorization. Protocol leakage is then the conditional mutual information an exposure adds to the history, equal to the Bayes log-risk it removes. For persistent processes with exponential kernels and Markov roles (Dirichlet or known role kernels), a finite state of node, hyperedge, pair and incidence statistics suffices. First-hop readouts of the star expansion cannot represent its pair terms. The incidence state machine (ISM) maintains this state in one pass and reads it by attention pooling. On persistent streams from public datasets, the ISM performs best overall across join, role and departure prediction and outperforms memory-based and first-hop temporal graph networks on joining.

1 INTRODUCTION

Hypergraphs represent higher-order interactions by hyperedges that connect arbitrary sets of nodes (Feng et al., 2019; Chien et al., 2022). In many systems these hyperedges persist. On community question-answering platforms, a question thread accumulates an asker, answerers, commenters and editors over months, and the tag set of a question is revised as the discussion evolves. Each node joins such a hyperedge at a specific time, carries an attribute such as its role in the thread and may later leave. Dynamic hypergraphs are therefore the natural representation of these data, and question threads and tag sets are among their standard sources (Benson et al., 2018a). Since the probability that a node joins a group depends on its ties to the current members (Backstrom et al., 2006), the natural unit of prediction is the *incidence*, a node as it occurs in one hyperedge.

Nevertheless, learning on dynamic hypergraphs has developed as a collection of separate tasks on transient hyperedges (Figure 1a). Temporal hyperedge prediction forecasts which set of nodes forms the next event (Liu et al., 2022; Behrouz et al., 2023; Gracious & Dukkipati, 2023; Ko et al., 2025). Node-label prediction within static hyperedges assigns a label to each member (Choe et al., 2023; Zheng & Worring, 2025), building on hypergraph networks that pass messages between nodes and hyperedges (Feng et al., 2019; Chien et al., 2022; Wang et al., 2023). Hyperedges whose members arrive over time are instead flattened into their star expansion, a bipartite stream of node-hyperedge interactions. Temporal graph networks process this stream with node memories, temporal walks, interaction-sequence encoders or state-space models (Kumar et al., 2019; Rossi et al., 2020; Wang et al., 2021; Cong et al., 2023; Yu et al., 2023; Lu et al., 2024; Li et al., 2025; Raghuvanshi et al., 2026). Recent benchmarks with Stack Exchange data likewise reduce threads to pairwise links (Robinson et al., 2024; Zhang et al., 2024).

Both lines leave two questions open. The first is what an evaluation protocol reveals beyond the history. Temporal link prediction has learned that negative samplers and batching change its conclusions (Poursafaei et al., 2022; Lampert et al., 2026). Leakage taxonomies name features computed after the prediction time (Kaufman et al., 2012; Kapoor & Narayanan, 2023). Yet hypergraph bench-

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.

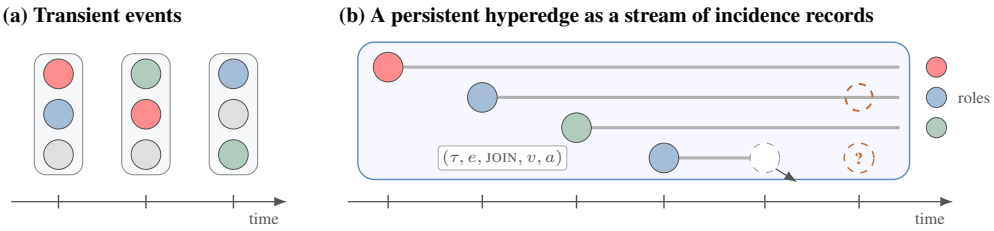

Figure 1: Dynamic hypergraphs as incidence streams. (a) Existing tasks treat each hyperedge as a transient event, a set of nodes observed at one time. (b) A persistent hyperedge is a stream of records: members join at different times with roles (colors) and may leave (arrow). A prediction (orange) concerns the next record given the records before it: who joins with which role, or who leaves.

marks ship node features computed on the whole hypergraph (Choe et al., 2023), and the size of such exposures has not been measured for dynamic hypergraphs. The second is what a model must retain from the history. Star-expansion models read first-hop neighborhoods or node memories. Expressivity results for temporal graph networks (Souza et al., 2022) concern temporal graphs rather than the pair terms that link a node to the current members of a group.

A single object in which both questions are precise would let one measure what any protocol leaks and derive what any model needs, for all tasks at once. We therefore treat a dynamic hypergraph as a stream of *incidence records*: a time, a hyperedge, an operation (join, leave or update), a node and an attribute (Figure 1b). The chain rule factors each record into where the next change happens, who joins or leaves, and which attribute the incidence carries. Every established task is one conditional of this factorization, distinguished only by the part of the record it reveals (Table 1).

The factorization makes both questions exact. Protocol leakage is the conditional mutual information between the target and whatever the protocol exposes beyond the strict history. It equals the Bayes log-risk the exposure removes, and same-timestamp updates and dump-time attributes are measurable instances. For persistent processes with exponential kernels and Markov roles (Dirichlet or known role kernels), a finite *incidence state* of node, hyperedge, pair and incidence statistics suffices for all three task conditionals. Its pair terms sit two hops away on the star expansion, beyond the reach of first-hop readouts. The *incidence state machine* (ISM) maintains this state with three streaming operators (scatter, filter and gather) and reads it with one network per stream whose heads correspond to the factors (Figure 2).

We build a benchmark of persistent streams from five Stack Exchange sites with three tasks: who joins a thread, which role a joining user takes, and which tags a question edit removes. Our central finding is that protocol choices that expose information outside the strict history can change scores by more than the choice of architecture. The benchmark's leakage audit shows that both attributes read from the data dump and same-day records applied in file order inflate the scores. Across the three tasks and five sites, the ISM performs best overall. On joining, the ISM is significantly ahead of GraphMixer on all five sites and far ahead of TGN, DyGFormer and DyG-Mamba. It stays within .005 mean reciprocal rank (MRR) of TPNet, whose time-decayed walk matrices are a neural counterpart of the pair state. On streams simulated from the process family with a strong pair term, the pair state brings every readout within 0.013 join MRR of the Bayes optimum.

Our contributions are as follows.

- We cast transient and persistent dynamic hypergraphs as incidence streams, in which every task is a conditional of one factorization and protocol leakage equals the Bayes log-risk an exposure removes.
- We introduce the incidence state machine, which maintains in one pass a finite state sufficient for persistent processes, including pair terms that first-hop and memory readouts of the star expansion cannot in general represent.
- We build a persistent-hypergraph benchmark with join, role and leave tasks and a leakage audit, on which the ISM performs best overall.

2 Related Work

Dynamic hypergraphs and relational event models. Temporal hyperedge prediction forecasts the next group (Ko et al., 2025), with fast and directional variants (Xu et al., 2025; Gracious et al., 2025), and hypergraph neural ODEs model disease progression in continuous time (Xiao et al., 2026). Edge-dependent vertex weights (Chitra & Raphael, 2019) and roles (Chodrow & Mellor, 2020; Choe et al., 2023) make the incidence the unit on static data. Network science studies group dynamics, the recurrence of node sets and generative models of time-varying groups (Iacopini et al., 2024; Benson et al., 2018b; Choo & Shin, 2022; Mancastroppa et al., 2025). Relational (hyper)event models fit event rates as log-linear functions of chosen past statistics (Butts, 2008; Lerner & Lomi, 2023; Lerner et al., 2025), such as subgroup repetition and persistence (Fabbrucci Barbagli et al., 2025; Poda et al., 2026). Incidence streams add the joins and leaves of persistent groups to these views, and the ISM reads the statistics that a log-linear process with exponential kernels requires (Section 4) without fixing the rate form.

Temporal graph networks. Memory, walk and sequence models learn from interaction streams (Rossi et al., 2020; Wang et al., 2021; Cong et al., 2023; Yu et al., 2023; Lu et al., 2024), and Souza et al. (2022) characterize the expressive power of memory and walk models. Heuristics often match them (Poursafaei et al., 2022; Cornell et al., 2025; Feldman et al., 2026). Recent models add state-space memories (Li et al., 2025; Raghuvanshi et al., 2026), kernelized edge attention (Waghmare et al., 2026) and attention-free token mixers (Zou et al., 2026), combine walks with memory (Farokhi et al., 2026), or drop memory and aggregation (Yi et al., 2025a). On a dynamic hypergraph all these models read the star expansion. Proposition 3 identifies the pair terms that their first-hop and memory readouts cannot represent, and our benchmark evaluates TGN, DyGFormer, GraphMixer, TPNet and DyG-Mamba under the strict protocol of Section 6.1.

Leakage and evaluation. Negative sampling, chronological protocols and sequential benchmarks reshaped temporal link prediction (Poursafaei et al., 2022; Huang et al., 2023; Gastinger et al., 2024; Yi et al., 2025b). Time-aware metrics capture clustered errors (Su & Wu, 2025), and fixed-size batches can leak same-timestamp records (Lampert et al., 2026). Definition 1 gives the exposures named by leakage taxonomies (Kaufman et al., 2012; Kapoor & Narayanan, 2023) one Bayes-risk measure, and our audit estimates it for dump-time attributes and time blocks on dynamic hypergraph data.

3 Dynamic Hypergraphs as Incidence Streams

Incidence records. Let $\mathcal{V}$ be a set of nodes and $\mathcal{E}$ a set of hyperedge identifiers. We observe a stream of *incidence records*

$$r_k = (\tau_k,\ e_k,\ o_k,\ v_k,\ a_k), \qquad k = 1, 2, \ldots, \tag{1}$$

ordered by time τ_k. Record k says that at time τ_k the incidence (v_k, e_k) undergoes the operation $o_k \in \{\text{JOIN}, \text{LEAVE}, \text{UPDATE}\}$ and carries the attribute a_k (a role, a score, or nothing). The membership of e at time τ is $U_e(\tau) = \{v : o_j \in \{\text{JOIN}, \text{UPDATE}\} \text{ for } j = \max\{i : \tau_i < \tau,\ (v_i, e_i) = (v, e)\}\}$, written $v \in e$ when τ is clear. A *transient* hyperedge (a meeting, a purchase) is a batch of JOINs that share one timestamp and never recur. A *persistent* hyperedge (a question thread, a tag set, a chat group) keeps its identifier while members join, change attributes and leave. Snapshot sequences are the special case whose records are the differences of consecutive snapshots.

Strict history and time blocks. The history before record k is the σ-algebra of all records with an earlier timestamp,

$$\mathcal{F}_{<k} = \sigma\big(\{r_j : \tau_j < \tau_k\}\big). \tag{2}$$

Records that share a timestamp form a *time block* and are mutually invisible. This choice matters: many sources round times to a day or a year. An implementation that applies the records of a block in file order lets later records see earlier ones (Corollary 1).

Factorization and tasks. Let $B_1, B_2, \ldots$ be the time blocks and $\mathcal{F}_{<B}$ the history before block B. We view the block times as the ground process of a marked point process whose mark is the set of

Table 1: Tasks as conditionals of (3). O_k is revealed at prediction time.

Task	Revealed O_k	Target	Hyperedges
Temporal hyperedge prediction	–	the member set of a new event	transient
T-JOIN (who joins)	e_k, $o_k = \text{JOIN}$	v_k	persistent
T-ROLE (role on joining)	e_k, $o_k = \text{JOIN}$, v_k	a_k	persistent
T-LEAVE (who leaves)	e_k, an edit occurs	the set of members with $o = \text{LEAVE}$	persistent

records in the block. For every hyperedge e changed in block B, the chain rule gives

$$p(e, \mathcal{R}_{B,e} \mid \tau_B, \mathcal{F}_{<B}) = \underbrace{p(e \text{ changes} \mid \tau_B, \mathcal{F}_{<B})}_{\text{where}} \underbrace{p\big(\{(o,v)\}_{r\in\mathcal{R}_{B,e}} \mid e, \tau_B, \mathcal{F}_{<B}\big)}_{\text{structural: who joins or leaves}} \times \underbrace{p\big(\{a\}_{r\in\mathcal{R}_{B,e}} \mid \{(o,v)\}, e, \tau_B, \mathcal{F}_{<B}\big)}_{\text{incidence: attributes}}, \tag{3}$$

where $\mathcal{R}_{B,e}$ is the set of records of block B on hyperedge e. Distinct hyperedges in one block are modeled as conditionally independent given $\mathcal{F}_{<B}$. A transient event is one such set with several JOINs. With fine timestamps a persistent stream has singleton blocks and (3) reduces to one record at a time. A *task* is a conditional of (3): it fixes which part of the block's mark is revealed at prediction time and which part is the target. With O_k the revealed part of the mark of query k, the information available to a deployed predictor is

$$\mathcal{F}_k^+ = \mathcal{F}_{<k} \vee \sigma(\tau_k, O_k), \tag{4}$$

and a predictor is *valid* if its output is $\mathcal{F}_k^+$-measurable. Table 1 lists temporal hyperedge prediction and the three persistent tasks we study; only O_k changes between them.

Nuisance marks. Data sources also store quantities generated *after* the labels they accompany, such as a question's final number of answers. We collect such quantities in a nuisance mark Z_k, which lies outside $\mathcal{F}_k^+$ yet is easily exposed through an input column or a static feature file.

Risk. For an evaluation set of queries, a predictor that outputs a distribution q_k over the target is scored by log-loss $-\log q_k(y_k)$, and after decoding by accuracy, macro-F1, average precision (AP) or reciprocal rank. For an information set $\mathcal{G}$ we write $R^\star(\mathcal{G})$ for the Bayes log-risk over $\mathcal{G}$-measurable predictors; $R^\star(\mathcal{F}^+) = H(y \mid \mathcal{F}^+)$ is the part of the target the history cannot explain.

4 What a Predictor May See, and What It Needs

We ask what a protocol adds to $\mathcal{F}^+$, at what cost, and which finite summary of $\mathcal{F}^+$ suffices.

4.1 Protocol leakage

Definition 1 (Protocol leakage)**.** *A protocol that makes $\mathcal{F}_k^+ \vee \sigma(Z_k)$ available for query k has leakage $\Lambda(Z) = \mathbb{E}_k\big[\, I\big(y_k; Z_k \;\big|\; \mathcal{F}_k^+\big)\big]$, where k is a uniformly drawn evaluation query and $I(\cdot;\cdot \mid \cdot)$ is conditional mutual information.*

Proposition 1 (Leakage is Bayes-risk inflation; standard)**.** *For any exposed quantity Z: (i) $R^\star(\mathcal{F}^+) - R^\star(\mathcal{F}^+ \vee \sigma(Z)) = \Lambda(Z)$; (ii) the Bayes accuracy rises by at most $\sqrt{2\Lambda(Z)}$ (Λ in nats); (iii) $\Lambda(Z) = 0$ if and only if y_k and Z_k are conditionally independent given $\mathcal{F}_k^+$, in which case exposing Z leaves the Bayes risk unchanged under every loss.*

The proof (Appendix A) uses the chain rule and Pinsker's inequality. Two exposures recur in practice.

Corollary 1 (Time blocks)**.** *If a streaming implementation applies records that share the timestamp of query k in file order, it exposes $Z_k = \{r_j : \tau_j = \tau_k,\ j < k\}$, with leakage equal to the information these same-time records carry about y_k beyond $\mathcal{F}_k^+$.*

Corollary 2 (Snapshot attributes). *Let $Z_k = g(\{r_j : \tau_j \le T\})$ be an attribute evaluated at a snapshot time $T > \tau_k$, such as a thread's final number of answers or a user's reputation at the date of a data dump. Then Z_k is in general not $\mathcal{F}_k^+$-measurable, and $\Lambda(Z) > 0$ whenever the part of g that depends on records after τ_k carries information about y_k beyond $\mathcal{F}_k^+$.*

Leakage is estimated by training the same readout with and without Z and taking the difference of test log-losses, $\widehat{\Lambda}(Z) = \widehat{R}(\mathcal{F}^+) - \widehat{R}(\mathcal{F}^+ \vee \sigma(Z))$, a plug-in estimate of conditional usable information (Xu et al., 2020; Hewitt et al., 2021). The difference equals $\Lambda(Z)$ plus the difference of the two readouts' excess risks, so its sign is not guaranteed. Section 6.2 reports it in bits.

4.2 A SUFFICIENT INCIDENCE STATE

Fix a finite set $\mathcal{T}$ of time constants and a finite attribute set $\mathcal{C}$. For a node v, a hyperedge e, a node pair $\{v, u\}$ and an incidence (v, e), the *incidence state* after the records of $\mathcal{F}_{<k}$ consists of:

- *node*: exponentially discounted counts $x^c_{v,\theta}$ of v's past records with attribute c, for every $\theta \in \mathcal{T}$, the last attribute c_v^{last} and the counts $\mathbf{A}_v(c, b)$ of its attribute transitions $c \to b$;
- *hyperedge*: the current members with their attributes, the creation and last-change times, and discounted counts of records on e;
- *pair*: the co-membership count $M_{v,u}$, its discounted counts $m_{v,u,\theta}$ for $\theta \in \mathcal{T}$, its last time, and the *pair-last* attribute $c_{v|u}^{\text{last}}$, v's attribute in the last hyperedge it shared with u;
- *incidence*: the tenure of (v, e) and its attribute history.

A record touches the state of its node, its hyperedge, its incidence, and the pairs it forms with the current members of the hyperedge. Each update costs $O(|\mathcal{T}|\,|e|)$. Every entry is $\mathcal{F}_{<k}$-measurable.

A family of persistent processes. Let $\mathcal{M}$ be the following process. A node v joins hyperedge e at time τ with intensity $\lambda_{v,e}(\tau) = \exp\big(\beta^\top \phi_{v,e}(\tau)\big)$, where $\phi_{v,e}(\tau)$ stacks exponentially discounted sums of v's past records, of e's past records and of v's past co-memberships with the current members of e, all with kernels $\exp(-t/\theta)$, $\theta \in \mathcal{T}$. Edits of e occur at times of a ground process that does not depend on the kernels below. At an edit, each current member v leaves ($y_v = 1$) independently with probability $\mathrm{sig}(\gamma^\top \psi_{v,e}(\tau))$, where sig is the logistic function. The vector $\psi_{v,e}$ is built likewise from the incidence terms of (v, e) and its discounted pair counts with the other current members. On joining, v takes attribute b with probability $\omega\,\pi_{v,e}(b) + (1-\omega)K_v(c_v^{\text{last}}, b)$, where K_v is node v's transition kernel and $\pi_{v,e}$ is the pair-last law, the law of the attribute v last held with a uniformly chosen current member of e it has met before; ω acts only if there is one. Either $\omega = 0$ and the rows of K_v are independent $\mathrm{Dirichlet}(\alpha\mathbf{1})$ draws, or $\omega > 0$ and K_v is known; β, γ, ω are fixed.

Proposition 2 (Sufficiency of the incidence state). *Under $\mathcal{M}$, for each task of Table 1 on persistent hyperedges, the Bayes predictive given $\mathcal{F}_k^+$ is determined by incidence states: (i) T-JOIN: $p(v_k = v \mid \mathcal{F}_k^+) \propto \lambda_{v,e}(\tau_k)$ over non-members v of e, where $\lambda_{v,e}$ is a function of the node, hyperedge and discounted pair states of (v, e); the Bayes ranking of candidates therefore depends on each candidate's own incidence state only; (ii) T-ROLE: for $\omega = 0$, $p(a_k = b \mid \mathcal{F}_k^+) = \big(\alpha + \mathbf{A}_v(c_v^{\text{last}}, b)\big)/\big(|\mathcal{C}|\alpha + \sum_{b'} \mathbf{A}_v(c_v^{\text{last}}, b')\big)$; for $\omega > 0$ it is $\omega\,\pi_{v,e}(b) + (1-\omega)K_v(c_v^{\text{last}}, b)$ if v has met a current member of e and $K_v(c_v^{\text{last}}, b)$ otherwise, which depends on the history only through c_v^{last} and the pair-last attributes of v with the current members; (iii) T-LEAVE: at an edit, the removed set has law $\prod_{v \in e} \mathrm{sig}(\gamma^\top \psi_{v,e})^{y_v} \big(1 - \mathrm{sig}(\gamma^\top \psi_{v,e})\big)^{1-y_v}$, a product of functions of each member's incidence and pair terms.*

Exponential kernels make the intensities Markov in the discounted counts (Oakes, 1975). Dirichlet–multinomial conjugacy makes the transition counts sufficient for the kernels; the factors of the likelihood that do not involve K_v drop out of its posterior (Appendix A). Proposition 2 follows from the construction of $\mathcal{M}$, which contains the effects we observe in practice. It fixes what an architecture must be able to represent: discounted node, hyperedge and pair counts at a few time constants, and per-partner attributes. Mixtures of exponential kernels are covered by adding their time constants to $\mathcal{T}$. The $\omega = 0.5$ columns of Table 11 test the regime $0 < \omega < 1$ with unknown K_v, which the proposition excludes (Appendix A).

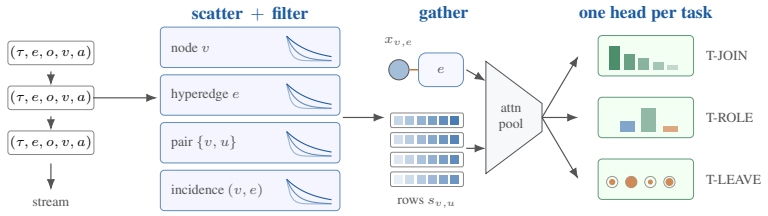

Figure 2: The incidence state machine. Each record scatters updates to the states of its node, its hyperedge, the pairs it forms with current members and its incidence. Every state is a bank of exponential filters with time constants from an hour to a year. A query gathers the query vector $x_{v,e}$ and one row per current member, pools the rows by attention, and applies the head of its factor.

4.3 What a readout of the star expansion can see

The star expansion turns an incidence stream into a bipartite temporal graph between nodes and hyperedges, on which temporal graph networks run unchanged. Proposition 2 locates terms that this reduction can hide: pair terms, which connect v to the current members of e through past hyperedges two hops away, and hyperedge-level terms such as the members' attributes.

Proposition 3 (Pair terms on the star expansion)**.** *Under $\mathcal{M}$ with $\omega > 0$, the Bayes predictive of T-ROLE depends on $\{c^{\text{last}}_{v|u} : u \in e\}$, and with a nonzero pair coefficient the Bayes ranking of T-JOIN depends on $\{m_{v,u,\theta} : u \in e\}$. (i)* First-hop readouts. *A readout whose input at query (v, e) is the time-stamped first-hop neighborhoods of v and of e on the star expansion, with edge attributes and without node-identity features, cannot represent these terms: there are histories with identical inputs and different pair-last attributes or pair counts. This covers the configurations we evaluate, which use zero node features; learned node-identity embeddings are memories and fall under (ii). (ii)* Memory readouts. *A readout that accesses the history at query (v, e) only through the memories of v, of e, of the current members of e and of at most L past hyperedges of v chosen without reference to the members of e (for example the most recent), each of dimension d and precision ν bits, cannot be exact on every history in which n nodes have each met the same n others with arbitrary attributes, once $n > (L+2)d\nu/\log_2 |\mathcal{C}|$. (iii) The incidence state stores $c^{\text{last}}_{v|u}$ and $m_{v,u,\theta}$ explicitly and is exact.*

Part (i) holds because whether a past hyperedge of v also contained u is second-hop information (Appendix A). It covers models that encode first-hop interaction sequences, such as DyGFormer and GraphMixer. Part (ii) is a counting bound that covers memory-based models such as TGN. Models that maintain time-decayed multi-hop walk statistics, such as TPNet, can approximate pair counts, and Table 3 shows TPNet close to the ISM on joining. Event-level constraints on transient events (exactly one leader) add a set-level term with a Bayes-optimal joint decoder (Appendix B).

5 The Incidence State Machine

We instantiate Propositions 2 and 3 as the *incidence state machine* (ISM, Figure 2). This streaming model keeps the state of Section 4 with three operators and reads it with one network per stream, whose heads correspond to the factors of (3).

Scatter. A record (τ, e, o, v, a) is applied after every query in its time block has been answered. For a join, each member u already in e with attribute a_u updates the pair state, including earlier joiners of the same block:

$$M_{v,u} \leftarrow M_{v,u} + 1, \quad m_{v,u,\theta} \leftarrow m_{v,u,\theta} \exp(-(\tau - t_{v,u})/\theta) + 1, \quad c^{\text{last}}_{v|u} \leftarrow a, \quad c^{\text{last}}_{u|v} \leftarrow a_u, \tag{5}$$

with $t_{v,u} \leftarrow \tau$ the time of their last contact. The record then enters the member set of e, the node state of v ($\mathbf{A}_v(c^{\text{last}}_v, a) \leftarrow \mathbf{A}_v(c^{\text{last}}_v, a) + 1$, then $c^{\text{last}}_v \leftarrow a$) and the incidence entry of (v, e).

Filter. Each count is a bank of one-pole filters: a channel with records at times $\tau_1 < \tau_2 < \cdots$ holds $x_\theta(\tau) = \sum_{j:\tau_j<\tau} \exp(-(\tau - \tau_j)/\theta)$, updated in constant time by $x_\theta \leftarrow x_\theta \exp(-\Delta\tau/\theta) + 1$ at each record, with $|\mathcal{T}| = 6$ time constants from one hour to one year. For each channel i, the

Table 2: Protocol audit with one GBDT readout. Top: snapshot leakage (Corollary 2), with and without dump-time attributes. Bottom: time blocks (Corollary 1) at day resolution, a day's records applied in chronological file order, same queries and candidates. $\widehat{\Lambda}$ in bits.

	CS	DataSci	SoftEng	Stats	AI
T-ROLE $\widehat{\Lambda}$ (bits), thread counts	0.407	0.556	0.219	0.344	0.462
T-ROLE macro-F1 gain	+0.149	+0.192	+0.071	+0.117	+0.162
T-JOIN MRR gain, user reputation, up-votes, views	+0.004	+0.005	+0.002	0.000	+0.008
T-JOIN MRR gain, same-day records in file order	+0.013	+0.038	+0.017	+0.011	+0.027
T-ROLE $\widehat{\Lambda}$ (bits), same-day records in file order	0.021	0.052	0.016	0.026	0.018

readout replaces the standardized vector $\tilde{x}_i \in \mathbb{R}^{|\mathcal{T}|}$ of the values $\log(1 + x_{i,\theta})$ by $W_i^\top \tilde{x}_i$, where $W_i \in \mathbb{R}^{|\mathcal{T}|\times|\mathcal{T}|}$ is learned and initialized to the identity.

Gather. A query reveals O_k (Table 1) and names a query incidence: the candidate (v, e) for T-JOIN, the joining incidence for T-ROLE, a current incidence for T-LEAVE. The ISM forms the query vector $x_{v,e}$ from three groups of state. The node group holds v's record count, the time since its last record, its attribute frequencies, its last attribute, its filter bank and its transition row $\mathbf{A}_v(c_v^{\text{last}}, \cdot)$. The hyperedge group holds the size, age, time since the last change, attribute counts and filter bank of e. The pair group aggregates over the current members: how many of them v has met and their fraction, the sum and maximum of $M_{v,u}$, the most recent contact, the counts of each pair-last attribute $c_{v|u}^{\text{last}}$ and $\sum_u m_{v,u,\theta}$. Each current member u (the 16 most recent) adds a row $s_{v,u}$ with u's attribute in e, $M_{v,u}$ and the time since $t_{v,u}$, both pair-last attributes, u's activity and $m_{v,u,\theta}$. Counts enter as $\log(1 + \cdot)$. Appendix C.3 lists all features and maps each term of Proposition 2 to them.

Readout. Two-layer MLPs of width 128 encode $z = \text{MLP}_x(x_{v,e})$ and $h_u = \text{MLP}_s(s_{v,u})$. The member rows are pooled by attention and by their mean,

$$\zeta_u = \text{softmax}_u\big(h_u^\top \ell_{\text{Q}}(z)/\sqrt{128}\big), \qquad \xi_{v,e} = \Big[\, z;\ \textstyle\sum_u \zeta_u h_u;\ \bar{h} \,\Big], \tag{6}$$

where ℓ_{Q} is affine and $\bar{h}$ is the mean of the h_u. Affine heads $\ell_{\text{J}}, \ell_{\text{R}}, \ell_{\text{L}}$ give $q(v_k = w) \propto \exp \ell_{\text{J}}(\xi_{w,e})$ over $w \in C_k \cup \{v_k\}$ with negatives C_k (Section 6.1), $q(a_k) = \text{softmax}(\ell_{\text{R}}(\xi_{v,e}))$ and $q(y_v = 1) = \text{sig}(\ell_{\text{L}}(\xi_{v,e}))$.

Training. On the thread stream one network minimizes $\mathcal{L}_{\text{J}} + \mathcal{L}_{\text{R}}$, and on the tag stream a network of the same architecture minimizes $\mathcal{L}_{\text{L}}$, with

$$\mathcal{L}_{\text{J}} = -\frac{1}{|\mathcal{Q}_{\text{J}}|} \sum_{k\in\mathcal{Q}_{\text{J}}} \log q(v_k), \qquad \mathcal{L}_{\text{R}} = -\frac{1}{|\mathcal{Q}_{\text{R}}|} \sum_{k\in\mathcal{Q}_{\text{R}}} \log q(a_k), \tag{7}$$

and $\mathcal{L}_{\text{L}}$ the mean binary cross-entropy of $q(y_v)$ over the current incidences at the training edits; $\mathcal{Q}$ denotes the training queries of a task. Each optimizer step sums the losses of one mini-batch per task. The epoch and the learning rate are selected by the mean validation score of the tasks, and the T-ROLE and T-LEAVE logits are divided by a temperature fitted on validation log-loss. Features are standardized on the training rows and clipped. The state and all query inputs are computed in a single pass that updates $O(|\mathcal{T}|\,|e|)$ entries per record, so training never replays the stream.

6 Experiments

6.1 A benchmark of persistent incidence streams

Data. We build persistent streams from five Stack Exchange sites (Computer Science, Data Science, Software Engineering, Cross Validated/Stats, Artificial Intelligence; April 2024 dumps). Two hypergraphs come from each dump, persistent versions of the thread and tag hypergraphs of Benson et al. (2018a), which keep one time per hyperedge. In the *thread stream* a thread is a hyperedge. A

Table 3: Persistent incidence streams: test results, mean over three seeds (std in Table 6). T-JOIN ranks the joiner against 49 users of the same familiarity stratum. †Chosen per site and task on validation. Bold: best per column; underlined: second best.

	T-JOIN: MRR					T-ROLE: macro-F1					T-LEAVE: AP				
Method	CS	DataSci	SoftEng	Stats	AI	CS	DataSci	SoftEng	Stats	AI	CS	DataSci	SoftEng	Stats	AI
Best heuristic†	.840	.840	.782	.847	.809	.522	.615	.583	.650	.602	.407	.586	.363	.523	.325
TGN (Rossi et al., 2020)	.785	.771	.706	.798	.722	.546	.662	.639	**.684**	.615	.323	.572	.420	.542	.279
DyGFormer (Yu et al., 2023)	.665	.638	.463	.683	.632	.537	.654	.627	.677	.621	.363	.613	.459	.572	.357
GraphMixer (Cong et al., 2023)	.866	.876	.795	.874	.825	.546	.645	.627	.678	.610	.353	.604	.452	.569	.339
TPNet (Lu et al., 2024)	.873	.883	**.816**	.880	.850	.553	.650	.639	.682	.599	.346	.526	.473	.567	**.368**
DyG-Mamba (Li et al., 2025)	.758	.762	.727	.813	.663	.557	**.663**	.636	.682	**.632**	.381	.596	**.479**	.589	.358
ISM (ours)	**.877**	**.887**	.815	**.883**	**.853**	**.558**	.650	**.641**	.683	.628	**.455**	**.636**	.473	**.621**	.361

user joins it with their first activity, as asker, answerer, commenter or editor (from the record type). In the *tag stream* a question is a hyperedge whose members are its tags, and tag edits give genuine LEAVE records. Sizes and splits are in Table 5 (Appendix C.1).

Protocol. Each thread stream keeps its most recent 600k join records (all records for smaller sites) and is split chronologically 70/15/15 by record. Tag edits are split at the same times. Records that share a timestamp are applied after all their queries are answered (Corollary 1). Let $U = U_{e_k}(\tau_k)$ and let $\mathcal{A}_k$ be the users with a record in the window before τ_k. T-JOIN queries joins with $U \neq \emptyset$ and $v_k \in \mathcal{A}_k$. It ranks v_k against 49 negatives C_k (nine in training) drawn uniformly without replacement from $N_k = \{w \in \mathcal{A}_k \setminus U : w \neq v_k,\ f(w) = f(v_k)\}$. Here the *familiarity stratum* $f(w) = \mathbb{I}[\exists u \in U : M_{w,u} > 0]$ indicates whether w has shared a thread with a current member. Queries for which 49 negatives cannot be drawn are dropped (familiar-stratum yield 0.78–0.99, Table 5). The reciprocal rank is $1/(1 + |\{w \in C_k : \eta_w > \eta_{v_k}\}| + \frac{1}{2}|\{w \in C_k : \eta_w = \eta_{v_k}\}|)$ for scores η, and MRR is its mean. Stratification removes the easiest contrast, a familiar joiner against strangers; uniform negatives raise every model's MRR (Table 8). T-ROLE predicts a_k for joins with $U \neq \emptyset$ (answerer, commenter or editor; the asker creates the thread and is determined by O_k). T-LEAVE predicts, at each tag edit of e at time τ, which tags in $U_e(\tau)$ are removed. No dump-time attribute is an input feature (for dump-time selections see Appendix C.1); we call this the strict protocol.

Models. *Heuristics*: recency, 30-day activity and co-occurrence for T-JOIN; the user's role frequencies, alone or mixed with pair-last roles, for T-ROLE; the tag's removal rate and its pair-conditioned rate for T-LEAVE. Validation selects one per site and task. *Star expansion*: TGN (Rossi et al., 2020), DyGFormer and GraphMixer (Yu et al., 2023; Cong et al., 2023), TPNet (Lu et al., 2024) and DyG-Mamba (Li et al., 2025) with their published default configurations, pretrained for link prediction. On threads they run on the bipartite user-thread stream with past roles as edge features and a role head; on tags, on the question-tag stream with edit operations as edge features and a removal head. Validation picks the training negatives of TGN and DyGFormer (DyGLib's random or our stratified ones) and TPNet's learning rate on the thread tasks. Candidates are scored from the model state before their evaluation batch of at most 50 records. *ISM*: one network per stream. Gradient-boosted trees (GBDT) on the same state serve as a second readout of it (Tables 4 and 6). Learned readouts get at most four configurations selected on validation and three seeds. Appendix C gives all implementation details, including seeds, and Appendix D.1 verifies that no baseline sees a record at or after the query time.

6.2 PROTOCOL AUDIT

Table 2 measures Corollaries 1 and 2 with one GBDT readout. Adding a thread's dump-time numbers of answers, comments and views, and its score, lowers the T-ROLE log-loss by $\widehat{\Lambda} = 0.22$–0.56 bits. It raises macro-F1 by .071–.192, more than any difference between models on T-ROLE in Table 3. The final counts summarize the very joins being predicted. RelBench's Stack Exchange loader drops these columns as time leakage (Robinson et al., 2024); benchmarks of node labels within hyperedges ship node centralities computed on the whole hypergraph, which would be snapshot attributes under a chronological protocol (Choe et al., 2023). Time blocks matter once timestamps are coarse, as observed for temporal link prediction (Lampert et al., 2026). Corollary 1 casts the effect as leakage. Rounded to days, at least 99.8% of the records of every site share a block. On the same

queries (Appendix C.2), features that apply a day's earlier records in file order raise T-JOIN MRR by .011–.038 ($\widehat{\Lambda} = 0.016$–$0.052$ bits on T-ROLE), more than the spread among the three strongest models of Table 3 on three sites. A query then sees who was active earlier that day.

6.3 Reading the incidence state

Table 3 gives the main comparison. The ISM performs best overall. It has the highest score in 9 of the 15 columns and one of the two highest in 14, and no other model is best in more than three. On T-JOIN the star-expansion models fall into three tiers. TPNet, which maintains time-decayed walk matrices, comes closest, within .005 MRR of the ISM on every site (Table 9). TPNet's projections are updated after each evaluation batch and the ISM's state after each time block; Appendix D.1 aligns the two schedules in both directions. The ISM leads GraphMixer, a first-hop sequence encoder, by .009–.028, significantly on every site (paired bootstrap over queries and test weeks, Table 9), and GraphMixer falls below even the node-state GBDT on four sites. The leads over TGN (.085–.131), DyG-Mamba (.070–.190) and DyGFormer (.200–.352) exceed what pair terms explain. All three score far higher against uniform negatives (Table 8), and a two-layer TGN narrows its gap by up to 0.02 MRR on the two smallest sites (Appendix D.1). On T-ROLE the ISM is best or second best on four of five sites (log-losses in Table 7). On T-LEAVE the ISM is best or second best on every site. It exceeds the best heuristic by .036–.110 AP on every site and all star-expansion models on three sites.

Table 4: Readout ladder on the thread tasks (site means; per site: Table 10).

State read	T-JOIN	T-ROLE
Node state	.855	.605
+ hyperedge state	.856	.622
+ pair aggregates	.859	.623
+ member rows (full GBDT)	.861	.625
ISM without member rows	.863	.630
ISM (ours)	.863	.632

6.4 What the readout needs

Table 4 reads growing parts of the state with the same GBDT. The node state alone reaches most of the T-JOIN score. The hyperedge state adds most for T-ROLE (whether a thread already has answers), and pair aggregates and member rows add further on T-JOIN. Averaged over the sites, the ISM reads the full state better than GBDT on both thread tasks. The difference is significant on T-ROLE on Computer Science and Software Engineering (up to .021 macro-F1) and on T-JOIN on two sites (Table 9). On the tag streams (Table 10), pair aggregates raise T-LEAVE AP on every site, and GBDT on the full state is within .006 AP of the ISM on four sites and ahead of it on Artificial Intelligence. Removing the ISM's member rows lowers T-LEAVE AP on four of five sites.

6.5 Sufficiency on simulated streams

Because the Bayes risk of real data is unknown, we simulate a log-count variant of $\mathcal{M}$ (200k joins, 3,000 users; Appendix D.2) and run the same builder, protocol and readouts (Table 11). With a strong pair term in the join intensity, the node state reaches MRR 0.29 against a Bayes optimum of 0.49–0.50. Adding the pair state brings every readout within 0.013 of it, as Proposition 2(i) predicts. With a weak pair term the pair state adds little and all readouts stay within 0.014 of Bayes. For T-ROLE with $\omega = 0.5$, the pair state brings the node state from 0.08–0.10 to within 0.003–0.011 nats of the plug-in oracle (Appendix D.2). On transient streams with an event-level constraint, the sufficient statistics likewise come closest to the Bayes risk (Table 12). Building the state takes at most 4 min per site on one CPU core, and an ISM for both thread tasks trains in at most 61 s on one GPU. The star-expansion models replay the stream in every epoch (1–70 min per run).

7 Conclusion

One factorization of incidence streams covers transient and persistent groups and states what a protocol leaks and what a model needs. On persistent benchmarks, dump-time attributes change role scores on every site by more than the choice of architecture, and same-day updates in file order raise join scores on every site. The incidence state machine maintains a finite incidence state in one pass and performs best overall on the three tasks with one network per stream. The persistent benchmark with its leakage audit gives dynamic hypergraph learning a common testbed.

AI USE STATEMENT

As guided by the conference AI disclosure policy, in this work, we used generative AI tools to assist in the writing of proofs, design or provide feedback on research methodology or experiments, implement methods, assist with translation, clean and reformat datasets and formulate mathematical claims. We have not used generative AI tools to generate synthetic datasets, help develop theoretical models or conceptual frameworks, provide critical ingredients for proving mathematical claims, interpret results, propose or refine hypotheses. Additionally, we also used generative AI tools to search related works, help authors create or modify scientific figures or images, help authors build experiment pipelines, help improve the writing quality and clarity of the human-drafted manuscript, edit a research paper to improve readability, sourcing/searching for information and check for potential plagiarism, brainstorming, identify relevant literature, format references. All work is first drafted by human authors and subsequently reviewed and edited by human authors, with the help of generative AI tools. And all AI-assisted work has been reviewed and verified by human authors. We take responsibility for the final content of this work, including text, claims or artifacts produced with the aid of generative AI.

ETHICS STATEMENT

All experiments use public data dumps released under a Creative Commons license. We use pseudonymous user identifiers only as node identifiers, publish no user-level outputs, and collected no new data. Predicting who joins a discussion could be misused for profiling; our benchmark releases aggregate metrics and the code to recompute them.

REPRODUCIBILITY STATEMENT

The stream, the strict history and the valid-predictor condition are defined in Section 3, leakage and sufficiency in Section 4 with proofs in Appendix A. Appendix C lists the data construction, the query and negative-sampling rules, the features, all hyperparameter grids, seeds and the selection rule. Anonymized code that builds the streams from the public dumps, runs every model and regenerates every table is provided in the supplementary material.

REFERENCES

Lars Backstrom, Dan Huttenlocher, Jon Kleinberg, and Xiangyang Lan. Group formation in large social networks: Membership, growth, and evolution. In *Proceedings of the 12th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 44–54, 2006. doi: 10.1145/1150402.1150412.

Ali Behrouz, Farnoosh Hashemi, Sadaf Sadeghian, and Margo Seltzer. CAT-Walk: Inductive hypergraph learning via set walks. In *Advances in Neural Information Processing Systems*, volume 36, pp. 32636–32671, 2023.

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*, 115 (48):E11221–E11230, 2018a. doi: 10.1073/pnas.1800683115.

Austin R. Benson, Ravi Kumar, and Andrew Tomkins. Sequences of sets. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pp. 1148–1157, 2018b. doi: 10.1145/3219819.3220100.

Carter T. Butts. A relational event framework for social action. *Sociological Methodology*, 38(1): 155–200, 2008. doi: 10.1111/j.1467-9531.2008.00203.x.

Eli Chien, Chao Pan, Jianhao Peng, and Olgica Milenkovic. You are AllSet: A multiset function framework for hypergraph neural networks. In *International Conference on Learning Representations*, 2022.

Uthsav Chitra and Benjamin Raphael. Random walks on hypergraphs with edge-dependent vertex weights. In *International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pp. 1172–1181, 2019.

Philip Chodrow and Andrew Mellor. Annotated hypergraphs: Models and applications. *Applied Network Science*, 5(1):9, 2020. doi: 10.1007/s41109-020-0252-y.

Minyoung Choe, Sunwoo Kim, Jaemin Yoo, and Kijung Shin. Classification of edge-dependent labels of nodes in hypergraphs. In *Proceedings of the 29th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pp. 298–309, 2023. doi: 10.1145/3580305.3599274.

Hyunjin Choo and Kijung Shin. On the persistence of higher-order interactions in real-world hypergraphs. In *Proceedings of the 2022 SIAM International Conference on Data Mining*, pp. 163–171, 2022. doi: 10.1137/1.9781611977172.19.

Weilin Cong, Si Zhang, Jian Kang, Baichuan Yuan, Hao Wu, Xin Zhou, Hanghang Tong, and Mehrdad Mahdavi. Do we really need complicated model architectures for temporal networks? In *International Conference on Learning Representations*, 2023. arXiv:2302.11636.

Filip Cornell, Oleg Smirnov, Gabriela Zarzar Gandler, and Lele Cao. On the power of heuristics in temporal graphs. In *Proceedings on "I Can't Believe It's Not Better: Challenges in Applied Deep Learning" at ICLR 2025 Workshops*, volume 296 of *Proceedings of Machine Learning Research*, pp. 37–46, 2025.

Amin Gino Fabbrucci Barbagli, Domenico De Stefano, Francesco Santelli, and Susanna Zaccarin. Unveiling collaboration persistence and interactions among Italian academic statisticians through relational hyperevent models. *Statistical Methods & Applications*, 34(5):1149–1167, 2025. doi: 10.1007/s10260-025-00809-w.

Soheila Farokhi, Xiaojun Qi, and Hamid Karimi. TAWRMAC: A novel dynamic graph representation learning method. In *Proceedings of the ACM Web Conference 2026*, pp. 684–695, 2026. doi: 10.1145/3774904.3792163.

Or Feldman, Krishna Sri Ipsit Mantri, Moshe Eliasof, and Chaim Baskin. Revisiting node affinity prediction in temporal graphs. In *International Conference on Learning Representations*, 2026. arXiv:2510.06940.

Yifan Feng, Haoxuan You, Zizhao Zhang, Rongrong Ji, and Yue Gao. Hypergraph neural networks. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pp. 3558–3565, 2019. doi: 10.1609/aaai.v33i01.33013558.

Julia Gastinger, Shenyang Huang, Mikhail Galkin, Erfan Loghmani, Ali Parviz, Farimah Poursafaei, Jacob Danovitch, Emanuele Rossi, Ioannis Koutis, Heiner Stuckenschmidt, Reihaneh Rabbany, and Guillaume Rabusseau. TGB 2.0: A benchmark for learning on temporal knowledge graphs and heterogeneous graphs. In *Advances in Neural Information Processing Systems (Datasets and Benchmarks Track)*, volume 37, pp. 140199–140229, 2024.

Tony Gracious and Ambedkar Dukkipati. Dynamic representation learning with temporal point processes for higher-order interaction forecasting. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 37, pp. 7748–7756, 2023. doi: 10.1609/aaai.v37i6.25939.

Tony Gracious, Arman Gupta, and Ambedkar Dukkipati. Neural temporal point processes for forecasting directional relations in evolving hypergraphs. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 39, pp. 16888–16896, 2025. doi: 10.1609/aaai.v39i16.33856.

John Hewitt, Kawin Ethayarajh, Percy Liang, and Christopher Manning. Conditional probing: Measuring usable information beyond a baseline. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pp. 1626–1639, 2021. doi: 10.18653/v1/2021.emnlp-main.122.

Shenyang Huang, Farimah Poursafaei, Jacob Danovitch, Matthias Fey, Weihua Hu, Emanuele Rossi, Jure Leskovec, Michael Bronstein, Guillaume Rabusseau, and Reihaneh Rabbany. Temporal Graph Benchmark for machine learning on temporal graphs. In *Advances in Neural Information Processing Systems (Datasets and Benchmarks Track)*, volume 36, pp. 2056–2073, 2023.

Iacopo Iacopini, Márton Karsai, and Alain Barrat. The temporal dynamics of group interactions in higher-order social networks. *Nature Communications*, 15(1):7391, 2024. doi: 10.1038/s41467-024-50918-5.

Sayash Kapoor and Arvind Narayanan. Leakage and the reproducibility crisis in machine-learning-based science. *Patterns*, 4(9):100804, 2023. doi: 10.1016/j.patter.2023.100804.

Shachar Kaufman, Saharon Rosset, Claudia Perlich, and Ori Stitelman. Leakage in data mining: Formulation, detection, and avoidance. *ACM Transactions on Knowledge Discovery from Data*, 6(4):15:1–15:21, 2012. doi: 10.1145/2382577.2382579.

Yunyong Ko, Da Eun Lee, Song Kyung Yu, and Sang-Wook Kim. Learning short-term and long-term patterns of high-order dynamics in real-world networks. In *Proceedings of the 34th ACM International Conference on Information and Knowledge Management*, pp. 4900–4904, 2025. doi: 10.1145/3746252.3760902.

Srijan Kumar, Xikun Zhang, and Jure Leskovec. Predicting dynamic embedding trajectory in temporal interaction networks. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pp. 1269–1278, 2019. doi: 10.1145/3292500.3330895.

Moritz Lampert, Christopher Blöcker, and Ingo Scholtes. From link prediction to forecasting: Addressing challenges in batch-based temporal graph learning. *Transactions on Machine Learning Research*, 2026. arXiv:2406.04897.

Jürgen Lerner and Alessandro Lomi. Relational hyperevent models for polyadic interaction networks. *Journal of the Royal Statistical Society Series A: Statistics in Society*, 186(3):577–600, 2023. doi: 10.1093/jrsssa/qnac012.

Jürgen Lerner, Marian-Gabriel Hâncean, and Matjaž Perc. Modeling temporal hypergraphs. *Journal of Complex Networks*, 13(6):cnaf054, 2025. doi: 10.1093/comnet/cnaf054.

Dongyuan Li, Shiyin Tan, Ying Zhang, Ming Jin, Shirui Pan, Manabu Okumura, and Renhe Jiang. DyG-Mamba: Continuous state space modeling on dynamic graphs. In *Advances in Neural Information Processing Systems*, volume 38, 2025. doi: 10.52202/085713-4299.

Yunyu Liu, Jianzhu Ma, and Pan Li. Neural predicting higher-order patterns in temporal networks. In *Proceedings of the ACM Web Conference 2022*, pp. 1340–1351, 2022. doi: 10.1145/3485447.3512181.

Xiaodong Lu, Leilei Sun, Tongyu Zhu, and Weifeng Lv. Improving temporal link prediction via temporal walk matrix projection. In *Advances in Neural Information Processing Systems*, volume 37, pp. 141153–141182, 2024.

Marco Mancastroppa, Giulia Cencetti, and Alain Barrat. Emerging Activity Temporal Hypergraph: A model for generating realistic time-varying hypergraphs. *Physical Review E*, 112(5):054305, 2025. doi: 10.1103/8rmn-mdgz.

David Oakes. The Markovian self-exciting process. *Journal of Applied Probability*, 12(1):69–77, 1975. doi: 10.2307/3212408.

Ashwin Paranjape, Austin R. Benson, and Jure Leskovec. Motifs in temporal networks. In *Proceedings of the Tenth ACM International Conference on Web Search and Data Mining*, pp. 601–610, 2017. doi: 10.1145/3018661.3018731.

Veronica Poda, Veronica Vinciotti, and Ernst C. Wit. Hyperevent network modelling of partially observed gossip data. *Journal of the Royal Statistical Society Series C: Applied Statistics*, 2026. doi: 10.1093/jrsssc/qlag041.

Farimah Poursafaei, Shenyang Huang, Kellin Pelrine, and Reihaneh Rabbany. Towards better evaluation for dynamic link prediction. In *Advances in Neural Information Processing Systems (Datasets and Benchmarks Track)*, volume 35, pp. 32928–32941, 2022.

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. In *International Conference on Machine Learning*, 2026. arXiv:2606.04672.

Joshua Robinson, Rishabh Ranjan, Weihua Hu, Kexin Huang, Jiaqi Han, Alejandro Dobles, Matthias Fey, Jan E. Lenssen, Yiwen Yuan, Zecheng Zhang, Xinwei He, and Jure Leskovec. RelBench: A benchmark for deep learning on relational databases. In *Advances in Neural Information Processing Systems (Datasets and Benchmarks Track)*, volume 37, pp. 21330–21341, 2024.

Emanuele Rossi, Ben Chamberlain, Fabrizio Frasca, Davide Eynard, Federico Monti, and Michael Bronstein. Temporal graph networks for deep learning on dynamic graphs. In *ICML 2020 Workshop on Graph Representation Learning and Beyond*, 2020. arXiv:2006.10637.

Amauri H. Souza, Diego Mesquita, Samuel Kaski, and Vikas Garg. Provably expressive temporal graph networks. In *Advances in Neural Information Processing Systems*, volume 35, pp. 32257–32269, 2022.

Junwei Su and Shan Wu. Temporal-aware evaluation and learning for temporal graph neural networks. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 39, pp. 20628–20636, 2025. doi: 10.1609/aaai.v39i19.34273.

Govind Waghmare, Srini Rohan Gujulla Leel, Nikhil Tumbde, Sumedh B G, Sonia Gupta, and Srikanta Bedathur. Kernelized edge attention: Addressing semantic attention blurring in temporal graph neural networks. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 40, pp. 26081–26089, 2026. doi: 10.1609/aaai.v40i31.39810.

Peihao Wang, Shenghao Yang, Yunyu Liu, Zhangyang Wang, and Pan Li. Equivariant hypergraph diffusion neural operators. In *International Conference on Learning Representations*, 2023.

Yanbang Wang, Yen-Yu Chang, Yunyu Liu, Jure Leskovec, and Pan Li. Inductive representation learning in temporal networks via causal anonymous walks. In *International Conference on Learning Representations*, 2021.

Tingsong Xiao, Yao An Lee, Zelin Xu, Yupu Zhang, Zibo Liu, Yu Huang, Jiang Bian, Jingchuan Guo, and Zhe Jiang. Temporally detailed hypergraph neural ODE for disease progression modeling. In *International Conference on Learning Representations*, 2026. arXiv:2510.17211.

Yilun Xu, Shengjia Zhao, Jiaming Song, Russell Stewart, and Stefano Ermon. A theory of usable information under computational constraints. In *International Conference on Learning Representations*, 2020.

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*, pp. 1727–1738, 2025. doi: 10.1145/3690624.3709327.

Lu Yi, Runlin Lei, Fengran Mo, Yanping Zheng, Zhewei Wei, and Yuhang Ye. Future link prediction without memory or aggregation. In *Advances in Neural Information Processing Systems*, volume 38, 2025a. doi: 10.52202/085713-0084.

Lu Yi, Jie Peng, Yanping Zheng, Fengran Mo, Zhewei Wei, Yuhang Ye, Yue Zixuan, and Zengfeng Huang. TGB-Seq benchmark: Challenging temporal GNNs with complex sequential dynamics. In *International Conference on Learning Representations*, 2025b. arXiv:2502.02975.

Le Yu, Leilei Sun, Bowen Du, and Weifeng Lv. Towards better dynamic graph learning: New architecture and unified library. In *Advances in Neural Information Processing Systems*, volume 36, pp. 67686–67700, 2023.

Sha Yuan, Yu Zhang, Jie Tang, Wendy Hall, and Juan Bautista Cabotà. Expert finding in community question answering: A review. *Artificial Intelligence Review*, 53(2):843–874, 2020. doi: 10.1007/s10462-018-09680-6.

Jiasheng Zhang, Jialin Chen, Menglin Yang, Aosong Feng, Shuang Liang, Jie Shao, and Rex Ying. DTGB: A comprehensive benchmark for dynamic text-attributed graphs. In *Advances in Neural Information Processing Systems (Datasets and Benchmarks Track)*, volume 37, pp. 91405–91429, 2024.

Yijia Zheng and Marcel Worring. Modeling edge-specific node features through co-representation neural hypergraph diffusion. In *Proceedings of the 34th ACM International Conference on Information and Knowledge Management*, pp. 4444–4454, 2025. doi: 10.1145/3746252.3761094.

Tao Zou, Chengfeng Wu, Tianxi Liao, Junchen Ye, and Bowen Du. Global-Lens Transformers: Adaptive token mixing for dynamic link prediction. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 40, pp. 16575–16583, 2026. doi: 10.1609/aaai.v40i19.38698.

A PROOFS

Throughout, k is a uniformly drawn evaluation query, $y = y_k$, $\mathcal{G} = \mathcal{F}_k^+$ and $\mathcal{G}' = \mathcal{G} \vee \sigma(Z_k)$. Conditional quantities are regular conditional distributions of y given the respective σ-algebra; $\mathcal{C}$ is finite.

Proposition 1. (i) For log-loss, the Bayes predictor over $\mathcal{G}$-measurable predictors is $p(\cdot \mid \mathcal{G})$ and its risk is $H(y \mid \mathcal{G})$; likewise for $\mathcal{G}'$. The chain rule gives $H(y \mid \mathcal{G}) - H(y \mid \mathcal{G}') = I(y; Z_k \mid \mathcal{G})$, and averaging over k gives $R^\star(\mathcal{G}) - R^\star(\mathcal{G}') = \Lambda(Z)$.

(ii) Let $P = p(\cdot \mid \mathcal{G}')$, $Q = p(\cdot \mid \mathcal{G})$ and $\hat{y}_Q = \arg\max_c Q(c)$, which is $\mathcal{G}$-measurable. By the tower property, $A^\star(\mathcal{G}) = \mathbb{E}[Q(\hat{y}_Q)] = \mathbb{E}[P(\hat{y}_Q)]$ and $A^\star(\mathcal{G}') = \mathbb{E}[\max_c P(c)]$. Write $\hat{y}_P = \arg\max_c P(c)$. Pointwise,

$$P(\hat{y}_P) - P(\hat{y}_Q) = \big[P(\hat{y}_P) - Q(\hat{y}_P)\big] + \big[Q(\hat{y}_P) - Q(\hat{y}_Q)\big] + \big[Q(\hat{y}_Q) - P(\hat{y}_Q)\big] \le 2\,\mathrm{TV}(P, Q),$$

because the middle bracket is non-positive and each outer bracket is at most $\mathrm{TV}(P, Q)$. Pinsker's inequality gives $\mathrm{TV}(P, Q) \le \sqrt{\mathrm{KL}(P\|Q)/2}$. Jensen's inequality and $\mathbb{E}[\mathrm{KL}(P\|Q)] = I(y; Z_k \mid \mathcal{G})$ then give

$$A^\star(\mathcal{G}') - A^\star(\mathcal{G}) \le 2\,\mathbb{E}\sqrt{\mathrm{KL}(P\|Q)/2} \le \sqrt{2\,\mathbb{E}\,\mathrm{KL}(P\|Q)} = \sqrt{2\Lambda(Z)}.$$

Non-negativity holds because $\mathcal{G} \subseteq \mathcal{G}'$.

(iii) $\Lambda(Z) = 0$ if and only if $\mathrm{KL}(P\|Q) = 0$ almost surely, that is, $P = Q$ almost surely. This is conditional independence of y and Z_k given $\mathcal{G}$. Then every $\mathcal{G}'$-measurable decision can be replaced by the $\mathcal{G}$-measurable decision that minimizes the same conditional expected loss under $Q = P$, so the Bayes risk is the same for every loss.

Corollaries 1 and 2. Both are Definition 1 with the stated Z_k. In each case Z_k depends on records with $\tau_j \ge \tau_k$, which are not in $\mathcal{F}_{<k}$, and is therefore not $\mathcal{G}$-measurable in general; Proposition 1(i)–(iii) apply unchanged.

Proposition 2. Write the likelihood of the records in $\mathcal{F}_k^+$ with the factorization (3). Under $\mathcal{M}$, the intensity of a join of v into e is $\exp(\beta^\top \phi_{v,e}(\tau))$, where each coordinate of $\phi_{v,e}(\tau)$ is a sum $\sum_{j:\tau_j<\tau} g(r_j)\exp(-(\tau - \tau_j)/\theta)$ over past records that touch v, e or a pair $\{v, u\}$ with u a current member of e. Such a sum satisfies $x(\tau) = x(\tau')\exp(-(\tau - \tau')/\theta)$ between records and $x \leftarrow x + g(r)$ at a record, so it is exactly a discounted count maintained by the incidence state (node, hyperedge, or the pair counts $m_{v,u,\theta}$) (Oakes, 1975). (i) Conditional on a join into e at τ_k, the joiner is v with probability proportional to its intensity among non-members, $p(v_k = v \mid \mathcal{F}_k^+) = \lambda_{v,e}(\tau_k)/\sum_{u \notin e} \lambda_{u,e}(\tau_k)$. The numerator is a function of the node, hyperedge and discounted pair states of (v, e). The denominator is common to all candidates, so the Bayes ranking and every ratio of predictive probabilities depend on the candidates' own states only. (iii) Given that an edit of e occurs at τ_k (revealed by O_k), the members leave independently with probabilities $\mathrm{sig}(\gamma^\top \psi_{v,e}(\tau_k))$ by the definition of $\mathcal{M}$. Each $\psi_{v,e}$ is a vector of discounted incidence and pair counts, hence a function of the state. The edit times do not depend on the other factors and cancel from the conditional. Our benchmark queries only edits that change the tag set, which couples the removals, so on real data the product law of (iii) is a working approximation. (ii) For $\omega = 0$, the attribute factor contributes $\prod_s K_{v_s}(c^{\mathrm{last}}_{v_s,s}, a_s)$ over past joins s. As a function of the row $K_v(c, \cdot)$ it is proportional to $\prod_b K_v(c, b)^{\mathbf{A}_v(c,b)}$, so with the $\mathrm{Dirichlet}(\alpha\mathbf{1})$ prior the posterior of the row is $\mathrm{Dirichlet}(\alpha\mathbf{1} + \mathbf{A}_v(c, \cdot))$, independently across nodes and rows, and the predictive is its mean. The

factors that do not involve the kernels (the ground process and the join and leave intensities) drop out of the posterior. For $\omega > 0$ with known kernels there is nothing to infer: given $\mathcal{F}_k^+$, the attribute is drawn from $\omega\,\pi_{v,e} + (1-\omega)K_v(c_v^{\text{last}}, \cdot)$ if v has met a current member of e and from $K_v(c_v^{\text{last}}, \cdot)$ otherwise. Here $\pi_{v,e}$ is the uniform mixture of the pair-last attributes $c_{v|u}^{\text{last}}$ over the current members u of e that v has met. (With unknown kernels and $0 < \omega < 1$ the kernel posterior is a mixture over which component generated each past join; the state is then no longer finite-dimensional, which is why the proposition separates the two cases.)

Proposition 3. The dependence on $\{c_{v|u}^{\text{last}}\}$ is Proposition 2(ii). (i) Take nodes v, u, w and two histories. In history A, v joins hyperedge e_1 as answerer at time t_1 after u created it, and joins e_2 as commenter at t_2 after w created it. History B is the same with u and w exchanged. At the query, u creates a new hyperedge e and v joins it. The time-stamped first-hop neighborhood of v on the star expansion is $\{(e_1, \text{answerer}, t_1), (e_2, \text{commenter}, t_2)\}$ in both histories, and that of e is $\{(u, \text{asker}, \tau_e)\}$ in both. Without node-identity features the two inputs are identical. Co-occurrence encodings between the two neighborhoods are empty, since the neighbors of v are hyperedges and those of e are nodes. The pair-last attribute is answerer in A and commenter in B, so under $\mathcal{M}$ with any $\omega > 0$ the Bayes predictives differ (we use $\omega = 1$ for concreteness) and one of the two is predicted wrongly. For T-JOIN, let v share e_1 with u in history A and with w in history B, all else equal. Then $m_{v,u,\theta}$ and hence $\lambda_{v,e}$ for the new hyperedge created by u differ between the histories whenever the pair coefficient is nonzero. The first-hop inputs of v and e are the same in both histories. (ii) Consider $2n$ nodes $v_1, \ldots, v_n, u_1, \ldots, u_n$ and a history in which, for every pair (i, j), v_i joins a hyperedge e_{ij} created by u_j with attribute $c_{ij} \in \mathcal{C}$, followed by the same activity for all configurations. A query in which u_j creates a new hyperedge and v_i joins it has Bayes predictive concentrated on c_{ij} ($\omega = 1$). The readout accesses the memories of v_i, of the new hyperedge and of its only current member u_j (the hyperedge's memory is computed from u_j's), and of at most L past hyperedges of v_i. Every c_{ij} must therefore be recoverable from the n node memories of the v_i, the n node memories of the u_j and the nL hyperedge memories accessible to the v_i, which hold at most $n(L+2)d\nu$ bits. The configurations (c_{ij}) carry $n^2 \log_2 |\mathcal{C}|$ bits, so for $n > (L+2)d\nu / \log_2 |\mathcal{C}|$ two configurations produce identical memories and one of them is predicted wrongly. (iii) The incidence state stores $c_{v_i|u_j}^{\text{last}}$ for every pair.

B Event-Level Constraints

Proposition 4 (Event-level constraints)**.** *If the attributes of a transient event satisfy* $p(Y \mid \mathcal{F}^+) \propto \prod_v q_v(y_v)\, \mathbb{I}[\,|\{v : y_v = c_0\}| = 1\,]$ *with per-incidence predictives* q_v *that are functions of each member's state* b_v*, then (a) the Bayes marginal of each member depends on the states of all members; (b) the joint MAP gives* c_0 *to* $\arg\max_w q_w(c_0) / \max_{c \neq c_0} q_w(c)$ *and every other member its best attribute other than* c_0*; (c) a readout of one incidence's own state loses at least* $2\,\mathbb{E}[(\rho_v - \mathbb{E}[\rho_v \mid b_v])^2]$ *nats of log-loss, where* ρ_v *is the Bayes probability that* v *takes* c_0*.*

Proof. By assumption, $p(Y \mid \mathcal{F}^+) = N^{-1} \prod_u q_u(y_u)\, \mathbb{I}[\,|\{u : y_u = c_0\}| = 1\,]$ with normalizer N. Summing out the other members,

$$p(y_v = c_0 \mid \mathcal{F}^+) = \frac{q_v(c_0) \prod_{u \neq v} \big(1 - q_u(c_0)\big)}{\sum_w q_w(c_0) \prod_{u \neq w} \big(1 - q_u(c_0)\big)} = \frac{r_v}{\sum_w r_w}, \qquad r_w = \frac{q_w(c_0)}{1 - q_w(c_0)},$$

which depends on every member's q_u unless the odds coincide. We assume $q_w(c_0) < 1$. An assignment with leader w has probability proportional to $q_w(c_0) \prod_{u \neq w} m_u = \big[q_w(c_0)/m_w\big] \prod_u m_u$ with $m_u = \max_{c \neq c_0} q_u(c)$, which is maximized by $w^\star = \arg\max_w q_w(c_0)/m_w$ with the other members at their best non-c_0 attribute. For the loss bound, with b_v the query incidence's own state, let $\rho_v = p(y_v = c_0 \mid \mathcal{F}^+)$ and $\bar{\rho}_v = \mathbb{E}[\rho_v \mid b_v]$. The chain rule gives $R^\star(\sigma(b_v)) - R^\star(\mathcal{F}^+) = \mathbb{E}\, I(y; \mathcal{F}^+ \mid b_v)$; by data processing this is at least $I(\mathbb{I}[y = c_0]; \mathcal{F}^+ \mid b_v) = \mathbb{E}\,\mathrm{KL}\big(\mathrm{Bern}(\rho_v) \| \mathrm{Bern}(\bar{\rho}_v)\big)$, and Pinsker's inequality for Bernoulli laws gives $\mathrm{KL} \geq 2(\rho_v - \bar{\rho}_v)^2$.

C Benchmark Construction and Model Details

This appendix covers the real-data benchmark; the synthetic streams are specified in Appendix D.2.

Table 5: Streams built from Stack Exchange dumps (2024-04). Thread streams: joins with roles; tag streams: tag additions and removals on questions. Splits are chronological (70/15/15 of thread records; tag edits split at the same times).

	CS	DataSci	SoftEng	Stats	AI
Join records	172,121	105,635	463,178	600,000	36,871
Users / threads	31,772 / 46,072	27,705 / 32,485	63,083 / 62,673	97,468 / 179,773	8,978 / 10,957
T-JOIN test queries	9,034	8,552	9,923	9,580	2,954
T-ROLE test queries	18,319	10,654	58,694	65,464	3,864
Familiar-stratum yield (test)	0.85	0.81	0.99	0.92	0.78
Questions / tags (tag stream)	48,388 / 837	36,774 / 761	63,422 / 2,261	183,895 / 2,211	12,378 / 1,173
Tag additions / removals	133,410 / 8,781	115,532 / 2,464	186,671 / 8,399	597,197 / 19,751	45,279 / 3,839
T-LEAVE test edits / rows	1,005 / 2,469	692 / 2,004	1,442 / 3,764	8,341 / 22,871	385 / 1,051
Removal rate (test rows)	0.18	0.18	0.14	0.13	0.16

C.1 DATA

Thread streams. From `Posts.xml`, `Comments.xml` and `PostHistory.xml` of each dump, a question thread is a hyperedge. A user joins it with their first activity: asking (question creation), answering (answer creation), commenting (on the question or one of its answers), or editing a post they do not own (history types 4–6). Later activity of the same user in the same thread adds no record. Deleted and anonymous users and the community bot are dropped, as are threads with fewer than two members. Both are dump-time selections of the record stream in the sense of Corollary 2: a thread's final number of members and a user's deletion are known only at the dump. Deleted users cannot be restored from the dump. For the thread filter, we rebuilt all five streams with every thread kept, on the same time windows and split times. GBDT's T-JOIN MRR and T-ROLE macro-F1 then change by at most 0.006 on every site, with mixed signs; this figure includes the resampling of T-JOIN queries. Times are POSIX seconds; records are ordered by time and then thread identifier, and records with equal times form time blocks. The role is taken from the record type. A handful of joins labeled asker occur after a thread exists (migrated questions); they are removed from T-ROLE (at most 16 per site; all but seven, on Stats, in training splits).

Tag streams. History type 3 (initial tags) adds each tag at question creation. Types 6 (tag edit) and 9 (rollback) are compared with the question's previous tag set, and removed and added tags become LEAVE and JOIN records. Edits that change nothing are ignored. Questions whose first tag record precedes the window start are dropped so that every in-window removal refers to an observed addition. At a tag-changing edit of question e at time τ, the T-LEAVE queries are all tags $v \in U_e(\tau)$, with target $y_v = 1$ if the edit removes v and $y_v = 0$ otherwise. If a split has more than $300{,}000$ (training) or $200{,}000$ (validation, test) query rows, its edit events are taken in a uniformly random order (seed 2026) until the next one would exceed that cap.

Relation to existing Stack Exchange data. The thread and tag hypergraphs of Benson et al. (2018a) contain one set per question, timestamped at its posting: the askers and answerers of threads whose question and answers were all posted within 24 hours, and the question's tags. Our streams add commenters and editors, a join time and a role for every member, and tag edits as JOIN and LEAVE records. Static role labels on threads exist for Stack Overflow and MathOverflow (Chodrow & Mellor, 2020) and, with random splits, for two other Stack Exchange sites (Choe et al., 2023). Timed user interactions appear as directed user graphs without thread identifiers (Paranjape et al., 2017), as bipartite user-question streams (Zhang et al., 2024), and as the relational database of the Cross Validated site in RelBench (Robinson et al., 2024). RelBench ships the raw post history but defines no hyperedge, role or tag-edit task. Question routing predicts the answerers of new questions (Yuan et al., 2020); T-JOIN instead ranks the users who join an existing thread in any role.

Windows and splits. Each thread stream keeps its most recent $600{,}000$ join records, starting at a time-block boundary. Computer Science, Data Science, Software Engineering and Artificial Intelligence have fewer records and are kept whole. The window of Stats starts in 2014. The split points are the times of the records at the 70% and 85% positions; the tag stream uses the same window and split times.

C.2 QUERIES AND NEGATIVES

With $U = U_{e_k}(\tau_k)$ and $\mathcal{A}_k$ as in Section 6.1, a join is eligible for T-JOIN if $U \neq \emptyset$ and $v_k \in \mathcal{A}_k$. Each eligible join is kept with probability $\min(1, n_{\max}/n)$, where n is the number of eligible joins in its split and $n_{\max}$ is 30,000 (training), 5,000 (validation) or 10,000 (test); the seed is 2026. The negatives come from $N_k = \{w \in \mathcal{A}_k \setminus U : w \neq v_k,\ f(w) = f(v_k)\}$ with $f(w) = \mathbb{I}[\exists u \in U : M_{w,u} > 0]$. For familiar joiners ($f(v_k) = 1$), N_k is the union of the co-membership neighborhoods $\{w : M_{w,u} > 0\}$ of the current members u, minus U and v_k, and the negatives are sampled from it uniformly without replacement. For unfamiliar joiners, users are drawn uniformly from $\mathcal{A}_k$, and a draw is rejected if it is v_k, a current member, already drawn or familiar. This rejection sampler makes at most 200 draws per required negative. Queries for which fewer than the required number of negatives are found are dropped; the yield is 1.00 for unfamiliar and 0.78–0.99 for familiar joiners (Table 5). Uniform negatives are drawn in the same way from $\mathcal{A}_k \setminus (U \cup \{v_k\})$ without the stratum condition. Training queries use nine stratified negatives. T-ROLE uses every join with $U \neq \emptyset$ in validation and test, and keeps each training join with probability $\min(1, 200{,}000/n)$. Ties are handled as in the MRR definition of Section 6.1: a tied negative counts one half, so a constant score receives rank 25.5, the expected rank of the joiner under a uniformly random order.

Snapshot audit. The top block of Table 2 trains the full-state GBDT with and without dump-time attributes on the same queries, with seeds 7, 11 and 13. For T-ROLE the added features are $\log(1+\cdot)$ of the question's final numbers of answers, comments and views and its score; for T-JOIN they are $\log(1+\cdot)$ of the candidate's reputation, up-votes and profile views at the dump. $\widehat{\Lambda}$ is the difference of the two test log-losses in bits. The user attributes change T-JOIN MRR by at most .008.

Time-block audit. The audit of Table 2 rounds every timestamp down to the day and builds each stream once. Queries, strata and negatives come from the state that applies a day's records after the day's queries, as in the benchmark. The leaky features come from a second state that applies every record right after its own query. Both variants therefore share every query and candidate. The difference of their test log-losses is the estimate $\widehat{\Lambda}$ of the leakage of Corollary 1, and the difference of their MRR is the change of score it causes. Rebuilding the whole stream in file order would also change the query set, because joins into threads created earlier on the same day become eligible. With this confound, the T-JOIN MRR changes by .047–.084 across the five sites.

C.3 INCIDENCE STATE AND FEATURES

Six time constants are used throughout: 1 hour, 6 hours, 1 day, 7 days, 30 days and 365 days. For thread streams, the query incidence $x_{v,e}$ has 74 entries:

- node (39): log count and log hours since the last record; role frequencies (Laplace, pseudo-count 0.25); last role (one-hot with "none"); discounted counts per role and time constant; the transition-count row $\mathbf{A}_v(c_v^{\text{last}}, \cdot)$ and its total.
- hyperedge (14): log size, log age and log time since the last join; log counts of members per role; whether the asker is present; discounted join counts per time constant.
- pair aggregates (21): number and fraction of current members met before; log sum and maximum of co-membership counts; log hours since the most recent co-membership; counts of the pair-last roles over current members; co-membership count with the asker and the pair-last role with the asker; the discounted co-membership counts $m_{v,u,\theta}$ summed over current members, per time constant.

Each member row $s_{v,u}$ has 29 entries: u's role in e; $\log(1 + M_{v,u})$ and the log age of the pair; $c_{v|u}^{\text{last}}$ and $c_{u|v}^{\text{last}}$ (one-hot with "none"); u's log count and discounted activity per time constant; $m_{v,u,\theta}$ per time constant. For tag streams, x has 34 entries. They cover tag usage and removal counts and rates with discounted versions, question age, and the number of tags and edits. They also cover the tag's tenure on the question, whether it was initial or added by an edit, and pair statistics with the other current tags (co-occurrence and pair-conditioned removal rates). Each member row has 24 entries describing another current tag and its pair statistics with the query tag. Absent thread-stream ages are set to a sentinel ($\log(1 + 10^6)$ hours).

From the state of Proposition 2 to the features. Each term that Proposition 2 requires has a counterpart in $x_{v,e}$ or $s_{v,u}$. (i) T-JOIN: the node and hyperedge terms of $\lambda_{v,e}$ are discounted counts, which enter per role and time constant. The pair term, the discounted co-membership counts summed over the current members, enters exactly as the pair aggregate $\sum_u m_{v,u,\theta}$. (ii) T-ROLE: c_v^{last} and the row $\mathbf{A}_v(c_v^{\text{last}}, \cdot)$ enter exactly, and the pair-last law $\pi_{v,e}$ is the normalized histogram of pair-last roles over the current members met before, whose counts enter exactly. (iii) T-LEAVE: the incidence and pair terms of $\psi_{v,e}$ enter through the tag, tenure and pair statistics. The $\log(1+\cdot)$ transform and the standardization are invertible. Four approximations remain. Member rows are truncated to the 16 most recent members, which removes per-partner detail but not the aggregates. Standardized values are clipped to $[-10, 10]$, and the tag-pair counts are not discounted. The readout has finite capacity and must learn the map from these features to the Bayes predictive; Table 11 measures the resulting gap on streams from a variant of $\mathcal{M}$.

Memory. The pair state holds 0.67M node pairs for Stats and 2.3M for Software Engineering. Building the Stats stream with all query features takes 173 s on one CPU core and peaks at 10.2 GiB of memory in Python. The slowest build (Software Engineering) takes 225 s. One ISM run for both thread tasks, with both learning rates, 12 epochs and inference, takes at most 61 s on one GPU. Storing the pair state exactly is what lets the state escape the memory bound of Proposition 3(ii); its size grows with the number of node pairs that have met.

C.4 Models and training

Heuristics. Let candidate w have n_w records, the last one Δ_w hours ago, and role counts $n_{w,c}$. The T-JOIN scores are $-\log(1+\Delta_w)$ (recency), $\sum_c \log(1+x^c_{w,30\,\text{d}})$ over the four roles (30-day activity) and $\log(1+\sum_{u\in U} M_{w,u}) + 10^{-3}\log(1+n_w)$ (co-occurrence with a count tie-breaker). The T-ROLE predictions are the smoothed role frequencies $\hat{\pi}_v(c) \propto n_{v,c} + 0.25$ over the three join roles, and $\frac{1}{2}\hat{\pi}_v(c) + \frac{1}{2}(P_c + 0.5)/\sum_{c'}(P_{c'} + 0.5)$. Here P_c is the number of current members u with $M_{v,u} > 0$ and $c^{\text{last}}_{v|u} = c$. For T-LEAVE, let E_v count the tag-changing edits at which tag v was present and R_v those that removed it. Let $E_{v,u}$ and $R_{v|u}$ be the same counts restricted to edits at which tag u was also present. The T-LEAVE scores are the removal rate $r_v = (R_v+1)/(E_v+2)$ and the pair-conditioned rate $\max_u (R_{v|u}+2r_v)/(E_{v,u}+2)$ over the other current tags u (r_v if there are none). The heuristic per site and task is the one with the best validation score.

GBDT. Histogram gradient boosting (scikit-learn), up to 400 iterations with early stopping on 10% of the training rows, grid over learning rate $\{0.05, 0.1\}$ and maximum leaf nodes $\{31, 63\}$, selected on validation MRR, macro-F1 or AP. Member rows enter as their masked mean and maximum. Seeds 7, 11, 13.

ISM. Hidden width 128, GELU MLPs with two layers for x and for member rows, single-head dot-product attention pooling. Learned 6×6 mixing matrices over time constants, initialized to the identity, for every channel that has one value per time constant. AdamW with weight decay 10^{-4}, learning rate in $\{10^{-3}, 3\cdot 10^{-4}\}$, 12 epochs, batches of 512 candidate groups (T-JOIN) and 4,096 rows (T-ROLE, T-LEAVE). The epoch and learning rate with the best mean validation score over the stream's tasks are kept. T-JOIN and T-ROLE share one network per site; T-LEAVE has its own. Features are standardized with the mean and standard deviation of all training rows of the stream's tasks and clipped to $[-10, 10]$. A temperature for each categorical or Bernoulli head is fitted on validation log-loss. Seeds 7, 11, 13.

Star expansion. TGN, DyGFormer and GraphMixer from DyGLib (Yu et al., 2023), TPNet (Lu et al., 2024) and DyG-Mamba (Li et al., 2025) run with their default hyperparameters (for DyG-Mamba, the authors' configuration for Wikipedia). The exception is TPNet's learning rate on the thread tasks, selected per site and task by the three-seed validation mean from $\{3\cdot 10^{-5}, 10^{-4}, 3\cdot 10^{-4}\}$. A seed-7 run screens each value, and values that beat the default on validation receive seeds 11 and 13. This selects $3\cdot 10^{-4}$ for T-ROLE on Computer Science and T-JOIN on Artificial Intelligence, and $3\cdot 10^{-5}$ for T-ROLE on Software Engineering and Artificial Intelligence; all other cells keep 10^{-4}. Users and threads are nodes of a bipartite temporal graph with zero node features, and each join is an interaction. Its edge feature is the one-hot role, used only for later interactions;

the query interaction's own feature is never visible when it is predicted. Link prediction is trained with our stratified training negatives (a softmax over each query's candidates, random negatives for the other interactions). TGN and DyGFormer are also trained with DyGLib's random negatives; GraphMixer, TPNet and DyG-Mamba use the stratified negatives only. The variant, like the epoch, is selected on validation (MRR on our stratified candidates for T-JOIN). Evaluation batches hold at most 50 records and never split a time block, and candidates are scored with the model state before their batch. T-ROLE adds a three-class head on the concatenated user and thread embeddings and fine-tunes end to end, selected on validation macro-F1. On the tag streams the models run on a question-tag graph with a removal head. Each model is run with seeds 7, 11 and 13. The median wall-clock time per thread-stream run with stratified training negatives is 9 to 13 minutes on one GPU for the other four models and 39 minutes for DyG-Mamba. Over all thread-stream runs, including those with random training negatives, it ranges over 1–70 min. Further details are in Appendix D.1.

D Additional Results

Table 6: Persistent incidence streams: test results, mean over three seeds ± std (omitted when below .0005); GBDT on the incidence state, a second readout of the ISM's state, is added for completeness. Star-expansion models run on the bipartite user-thread and question-tag streams. T-JOIN ranks the joiner against 49 users of the same familiarity stratum. †Chosen per site and task on validation. Bold: best per column; underlined: second best.

	T-JOIN: MRR					T-ROLE: macro-F1					T-LEAVE: AP				
Method	CS	DataSci	SoftEng	Stats	AI	CS	DataSci	SoftEng	Stats	AI	CS	DataSci	SoftEng	Stats	AI
Best heuristic†	.840	.840	.782	.847	.809	.522	.615	.583	.650	.602	.407	.586	.363	.523	.325
TGN (Rossi et al., 2020)	.785±.018	.771±.008	.706±.017	.798±.011	.722±.005	.546±.005	.662±.002	.639±.001	**.684±.001**	.615±.006	.323±.012	.572±.004	.420±.002	.542±.002	.279±.011
DyGFormer (Yu et al., 2023)	.665±.004	.638±.006	.463±.010	.683±.003	.632±.007	.537±.003	.654±.003	.627	.677±.001	.621±.003	.363±.013	.613±.011	.459±.004	.572±.004	.357±.004
GraphMixer (Cong et al., 2023)	.866±.001	.876±.001	.795±.005	.874±.002	.825±.005	.546±.005	.645±.002	.627±.004	.678	.610±.006	.353±.007	.604±.017	.452±.016	.569±.002	.339±.006
TPNet (Lu et al., 2024)	.873±.001	.883	**.816**	.880±.001	.850±.002	.553±.002	.650±.003	.639	.682±.001	.599±.005	.346±.003	.526±.036	.473±.003	.567±.002	.368±.008
DyG-Mamba (Li et al., 2025)	.758±.019	.762±.005	.727±.004	.813±.002	.663±.013	.557±.003	**.663±.001**	.636±.001	.682±.001	**.632±.002**	.381±.004	.596±.024	**.479±.002**	.589±.002	.358±.005
GBDT on the incidence state	.875±.002	**.887±.001**	.811±.002	.882	.851±.001	.537±.005	.650±.002	.635	**.684±.001**	.622±.002	.449±.002	.632±.003	.477±.001	.616	**.413±.001**
ISM (ours)	**.877±.001**	**.887**	.815	**.883±.001**	**.853**	**.558±.002**	.650±.001	**.641±.001**	.683±.003	.628±.004	**.455**	**.636±.004**	.473±.002	**.621±.001**	.361±.004

Table 6 repeats Table 3 with standard deviations and adds GBDT on the incidence state. Table 7 reports T-ROLE log-loss and accuracy, Hits@10, T-JOIN by familiarity stratum and T-LEAVE AUC. Table 9 gives paired bootstrap intervals for ISM versus GBDT, TPNet and GraphMixer, with 2,000 resamples for T-JOIN, 500 for T-ROLE and T-LEAVE, and 95% percentile intervals. It also marks the differences that stay significant when whole weeks of the test split are resampled, which respects the dependence between queries within a calendar week. Table 10 gives the state-composition ladder per site, including T-LEAVE. For completeness, we list the remaining observations. On the smallest site (Artificial Intelligence), GBDT is ahead of the ISM on T-LEAVE; the interval excludes zero when edit events are resampled and contains it when weeks are resampled (Table 9). On Data Science, DyG-Mamba has the highest T-ROLE macro-F1, with TGN second. On T-ROLE, DyG-Mamba is .004 ahead of the ISM on Artificial Intelligence, and TGN and GBDT are .001 ahead of it on Stats. On Software Engineering, TPNet has the highest T-JOIN MRR (Table 3). On T-LEAVE, DyG-Mamba has the highest AP on Software Engineering, where TPNet is level with the ISM. On Artificial Intelligence, TPNet is ahead of the ISM, and GBDT on the same state is ahead of all star-expansion models (Table 6). With uniform negatives (Table 8) the ISM stays within .005 MRR of TPNet on every site, and GBDT on the same state falls .012–.045 behind TPNet. Learned mixing across time constants changes the ISM's scores by at most .002, and training one ISM per thread task instead of one per stream changes its thread-task scores by at most .002 (Table 10). Building the thread streams without the transition counts $\mathbf{A}_v$ of Proposition 2 changes the T-JOIN MRR and T-ROLE macro-F1 of the ISM and GBDT by at most .005.

D.1 Star-expansion baselines

All five models run from their public code with the published configurations for the Wikipedia dataset. We use DyGLib at commit 3aacc36b, TPNet at commit 7dd0cf4 and DyG-Mamba at commit ce54319e, with mamba-ssm 2.2.6 compiled for our GPUs. Only the data interface, the losses and the evaluation loop are ours. DyGFormer and DyG-Mamba pad every neighbor sequence to its maximum length, so that no score depends on the other queries in its batch. We checked in the code that the neighbor samplers return only strictly earlier interactions, that TGN memories and TPNet

Table 7: Additional test metrics per site (mean over three seeds). Familiar joiners have shared a thread with a current member before the query.

	CS	DataSci	SoftEng	Stats	AI
T-ROLE log-loss (nats)					
Best heuristic	.885	.790	.789	.729	.898
TGN	.793±.003	.703±.006	.682±.005	.681±.003	.758±.007
DyGFormer	.784±.008	.704±.012	.678±.001	.676±.001	.758±.002
GraphMixer	.802±.003	.775±.004	.712±.009	.690±.004	.803±.009
TPNet	.786±.003	.700±.002	.677±.001	.670	.787
DyG-Mamba	.770±.002	.687±.001	.671	.664±.001	.736±.002
GBDT	.776±.003	.697±.003	.672±.001	.654±.001	.759±.005
ISM	.768±.001	.693	.671±.001	.664±.002	.741±.002
T-ROLE accuracy					
GBDT	.620±.004	.670±.001	.692	.707±.001	.627±.003
ISM	.627±.002	.668±.002	.693±.001	.703±.001	.633±.003
T-JOIN Hits@10					
GBDT	.976±.001	.970	.952±.003	.980	.965±.001
ISM	.978	.971	.957	.981±.001	.965±.001
ISM w/o member rows	.978	.971±.001	.958±.001	.981±.001	.964±.001
ISM w/o mixing	.978	.971±.001	.957	.981±.001	.964±.002
T-JOIN MRR, familiar joiners					
GBDT	.828±.001	.810±.001	.799±.002	.848±.001	.791±.003
ISM	.832±.001	.809±.002	.803	.849±.001	.789±.001
ISM w/o member rows	.831±.001	.809±.001	.803±.001	.850±.001	.790±.003
T-JOIN MRR, unfamiliar joiners					
GBDT	.927±.002	.912	.843±.002	.929	.887±.001
ISM	.928	.912	.845±.001	.928±.001	.892±.001
ISM w/o member rows	.928±.001	.912	.845±.001	.928	.891±.001
T-LEAVE AUC					
GBDT	.746±.003	.799±.010	.747±.002	.833	.766±.003
ISM	.743±.001	.811±.003	.754±.003	.839±.003	.717±.002
ISM w/o member rows	.738±.001	.794±.002	.750±.003	.844±.001	.707±.001

projections are updated only after a batch, and that no batch splits a time block. Every run also repeats its first fifteen test batches with all role features at or after the query time overwritten. The role probabilities change by at most $2 \cdot 10^{-6}$. The join scores change by at most 10^{-4} in logit for the other four models, consistent with floating-point rounding, and by up to $5.9 \cdot 10^{-4}$ for DyG-Mamba. For DyG-Mamba, scoring the same weights in a different order alone changes the join scores by up to $2.5 \cdot 10^{-4}$. Re-scoring its saved checkpoints with the scoring order held fixed, overwriting the features changes no join score. A positive control that makes the samplers inclusive changes the join scores by up to 37 logits and drives the test MRR of all five models to at least 0.998 (Artificial Intelligence, seed 7). DyGLib casts neighbor times to 32 bits, which over a multi-year span shifts them by up to 16 seconds. Neighbor selection is done in 64 bits, so this only perturbs the time encoding. For TPNet, whose $\log(1 + \Delta t)$ turns a negative difference into NaN, we keep the times in 64 bits. No run reached the wall-clock caps of two hours (T-JOIN) and one hour (T-ROLE). For the evaluated TGN (memory dimension $d = 172$, $\nu = 32$, $L = 10$ sampled neighbors, three join roles), the threshold of Proposition 3(ii) is $n > (L + 2)d\nu / \log_2 |\mathcal{C}| \approx 4.2 \cdot 10^4$. A two-layer TGN, with all else unchanged, was run on the two smallest sites with both training-negative variants and three seeds. From one to two layers, T-JOIN MRR changes from .722 to .722 (stratified) and from .695 to .727 (random) on Artificial Intelligence, and from .771 to .787 and from .760 to .791 on Data Science. With the variant selected on validation, the two-layer TGN reaches .727 and .791, against .722 and .771 in Table 3; T-ROLE macro-F1 changes by at most .008.

Tag streams. For T-LEAVE the same five models run on a bipartite temporal graph of questions and tags. Every tag record (an addition at creation, an addition by an edit, or a removal) is an interaction between the question and the tag, with the one-hot operation as its edge feature. Node

Table 8: T-JOIN MRR with uniform versus familiarity-stratified negatives: a change of difficulty, not leakage.

	CS	DataSci	SoftEng	Stats	AI
Best heuristic, uniform	0.901	0.873	0.857	0.931	0.862
Best heuristic, stratified	0.840	0.840	0.782	0.847	0.809
TGN, uniform	0.878	0.830	0.835	0.918	0.775
TGN, stratified	0.785	0.771	0.706	0.798	0.722
DyGFormer, uniform	0.812	0.720	0.687	0.864	0.712
DyGFormer, stratified	0.665	0.638	0.463	0.683	0.632
GraphMixer, uniform	0.927	0.908	0.868	0.944	0.867
GraphMixer, stratified	0.866	0.876	0.795	0.874	0.825
TPNet, uniform	0.934	0.915	0.894	0.950	0.887
TPNet, stratified	0.873	0.883	0.816	0.880	0.850
DyG-Mamba, uniform	0.875	0.827	0.848	0.926	0.740
DyG-Mamba, stratified	0.758	0.762	0.727	0.813	0.663
GBDT, uniform	0.900	0.903	0.849	0.918	0.866
GBDT, stratified	0.875	0.887	0.811	0.882	0.851
ISM, uniform	0.938	0.917	0.890	0.950	0.891
ISM, stratified	0.877	0.887	0.815	0.883	0.853

Table 9: Paired bootstrap over test queries (T-LEAVE: over edit events), per-query reciprocal ranks (T-JOIN) and predicted probabilities (T-ROLE, T-LEAVE) averaged over the three seeds: mean difference and 95% interval of ISM minus GBDT (first three rows) and of ISM minus the two strongest star-expansion models on T-JOIN. For T-ROLE and T-LEAVE the metric is that of the seed-averaged predictor, so differences can deviate from those of Table 6. *Interval excludes zero before rounding; *†also when whole test weeks are resampled instead of queries.

	CS	DataSci	SoftEng	Stats	AI
T-JOIN MRR	$+0.003^{*\dagger}$ [+0.001, +0.005]	0.000 [−0.002, +0.002]	$+0.004^{*\dagger}$ [+0.002, +0.006]	0.000 [−0.002, +0.002]	+0.002 [−0.002, +0.006]
T-ROLE macro-F1	$+0.021^{*\dagger}$ [+0.015, +0.026]	0.000 [−0.007, +0.006]	$+0.006^{*\dagger}$ [+0.004, +0.009]	0.000 [−0.002, +0.002]	+0.001 [−0.010, +0.011]
T-LEAVE AP	+0.001 [−0.017, +0.022]	0.000 [−0.021, +0.024]	−0.008 [−0.021, +0.006]	$+0.005^{*}$ [0.000, +0.009]	-0.062^{*} [−0.104, −0.019]
T-JOIN MRR, ISM − TPNet	$+0.004^{*\dagger}$ [+0.002, +0.006]	$+0.005^{*\dagger}$ [+0.002, +0.007]	−0.001 [−0.003, +0.001]	+0.002 [0.000, +0.004]	$+0.004^{*}$ [0.000, +0.007]
T-JOIN MRR, ISM − GraphMixer	$+0.011^{*\dagger}$ [+0.009, +0.014]	$+0.012^{*\dagger}$ [+0.009, +0.014]	$+0.020^{*\dagger}$ [+0.017, +0.023]	$+0.009^{*\dagger}$ [+0.006, +0.011]	$+0.029^{*\dagger}$ [+0.024, +0.034]

features are zero. Each model is first trained for link prediction on the training interactions with DyGLib's random negatives. It is then fine-tuned end to end with a Bernoulli removal head on the concatenated tag and question embeddings, by binary cross-entropy on the training rows of T-LEAVE. All five keep their default configurations, including TPNet's learning rate. The epochs of both stages are selected on validation, by link-prediction AP and then by T-LEAVE AP. A query row, a current tag of a question at an edit time τ, is scored from the model state before the evaluation batch that holds τ. The records at τ, including the removals being predicted, are applied after scoring. Test AP is pooled over the same rows as for the ISM and GBDT. No run reached the wall-clock caps of two hours (link prediction) and one hour (T-LEAVE). The analogous probe, which overwrites the operation features at or after the query time, changes no removal probability by more than 10^{-6} in any run. On the same weights, an inclusive sampler changes every probed score, and in full runs of TGN, DyGFormer, GraphMixer and TPNet with the inclusive sampler on Artificial Intelligence the test AP reaches at least 0.997.

Update schedules. The benchmark lets every model use all records before the query's time block. The ISM's state is updated after every time block. TGN memories and TPNet projections are updated after each evaluation batch of at most 50 records, while their neighbor samplers see every earlier record. We aligned the schedules in both directions. First (preregistered), we rebuilt the ISM and GBDT features from a state updated only at the end of the same batches as DyGLib, with identical queries and candidates. T-JOIN MRR then drops by $.016$–$.043$ for the ISM and by $.019$–$.054$ for GBDT, which puts the ISM $.014$–$.039$ below TPNet, and the ISM's T-ROLE macro-F1 drops by $.010$–$.026$. The most recent activity is therefore highly predictive, and it is carried by the state rather than by the readout. This alignment is stricter for the ISM than for TPNet, whose neighbor lists stay current. Second (exploratory, seed 7), we evaluated TPNet with an update after every time block, without retraining its 200-record training batches. Its T-JOIN MRR improves on no site (changes of

Table 10: State-composition ladder per site: GBDT on growing state groups and ISM ablations; mean ± std over three seeds (std omitted when below .0005).

	CS	DataSci	SoftEng	Stats	AI
T-JOIN MRR					
Node state	.868±.002	.882	.800	.874±.001	.849±.001
+ hyperedge state	.870±.001	.883±.001	.806	.873±.001	.850
+ pair aggregates	.875±.001	.886±.001	.808±.001	.881±.001	.847±.001
+ member rows (GBDT on the full state)	.875±.002	.887±.001	.811±.002	.882	.851±.001
ISM without member rows	.877	.887	.814±.001	.883	.853±.001
ISM without time-constant mixing	.877	.887±.001	.814	.883±.001	.852±.001
ISM trained per task	.878	.887	.815±.001	.884	.854±.001
ISM (ours)	.877±.001	.887	.815	.883±.001	.853
T-ROLE macro-F1					
Node state	.518±.002	.640±.005	.602±.002	.674±.001	.591±.004
+ hyperedge state	.537±.001	.641±.002	.632±.001	.684	.617±.003
+ pair aggregates	.539±.002	.643±.001	.635±.002	.684±.001	.615±.002
+ member rows (GBDT on the full state)	.537±.005	.650±.002	.635	.684±.001	.622±.002
ISM without member rows	.555±.001	.651±.004	.639±.001	.681±.003	.627±.004
ISM without time-constant mixing	.557±.001	.652±.002	.641±.001	.683±.003	.626±.005
ISM trained per task	.556±.004	.650±.003	.641±.002	.682±.001	.627±.001
ISM (ours)	.558±.002	.650±.001	.641±.001	.683±.003	.628±.004
T-LEAVE AP					
Node state	.376±.006	.596±.006	.437±.006	.538±.010	.319±.010
+ hyperedge state	.410±.009	.639±.006	.479±.007	.599±.002	.386±.009
+ incidence terms	.415±.008	.636±.007	.478±.004	.601±.001	.391±.016
+ pair aggregates	.445±.008	.639±.003	.488±.002	.614±.003	.396±.003
+ member rows (GBDT on the full state)	.449±.002	.632±.003	.477±.001	.616	.413±.001
ISM without member rows	.444±.002	.624±.001	.471±.004	.622±.001	.345±.003
ISM without time-constant mixing	.455	.637±.004	.472±.002	.621±.001	.361±.004
ISM (ours)	.455	.636±.004	.473±.002	.621±.001	.361±.004

Table 11: Streams simulated from a variant of $\mathcal{M}$: T-ROLE log-loss (LL, nats) and T-JOIN MRR. Plug-in oracle: Bayes with estimated K_v and β (Appendix D.2). Below the rule: GBDT on growing state groups and ISM (three-seed means); one stream per setting.

	ω=0, strong pair		ω=0, weak pair		ω=0.5, strong pair		ω=0.5, weak pair	
Readout	LL↓	MRR↑	LL↓	MRR↑	LL↓	MRR↑	LL↓	MRR↑
Bayes (exact)	0.718	0.500	0.712	0.132	0.662	0.493	0.664	0.129
Plug-in oracle	0.718	0.500	0.712	0.132	0.740	0.493	0.733	0.130
Node state	0.725	0.288	0.719	0.120	0.819	0.291	0.836	0.120
+ hyperedge	0.724	0.294	0.719	0.118	0.817	0.296	0.837	0.119
+ pair	0.723	0.487	0.719	0.120	0.744	0.483	0.742	0.119
GBDT, full state	0.724	0.492	0.721	0.119	0.745	0.487	0.744	0.121
ISM (ours)	0.723	0.494	0.718	0.125	0.743	0.489	0.743	0.124

at most .006, none positive), and the ISM (three seeds) is ahead of it by up to .008 on four sites and level on Software Engineering. Neither alignment is symmetric: the first leaves TPNet's neighbor lists current while the ISM's whole state is stale, and the second changes TPNet's schedule only at evaluation. We therefore claim parity with TPNet on T-JOIN, not a margin. GraphMixer has no memory and reads all earlier records through its sampler, so its gap to the state readouts does not depend on the schedule. TGN's gap exceeds the whole effect of the schedule on the ISM.

D.2 Synthetic streams

Persistent streams from $\mathcal{M}$. Threads are created by a Poisson process, the asker drawn in proportion to one plus her 30-day discounted activity. Joins arrive by an independent Poisson process. The thread is drawn by recency of its activity. The joiner is drawn among all non-members with probability proportional to $\exp(\beta_n \phi_{\text{node}} + \beta_e \phi_{\text{edge}} + \beta_p \phi_{\text{pair}})$, where each ϕ is $\log(1 + \cdot)$ of the

Table 12: Synthetic streams with exact Bayes risk: test log-loss in nats (mean over three generator seeds). ω: probability of repeating the role last held with a current partner; flip: role-mixing rate; κ: strength of the exactly-one-leader constraint. The sufficient statistics come closest to the Bayes risk in every setting.

Readout	ω=0.0, flip=0.05	ω=0.5, flip=0.05	ω=0.0, flip=0.3	ω=0.5, flip=0.3	ω=0.3, flip=0.1, κ=2	ω=0.3, flip=0.1, κ=8
Bayes (exact)	0.192	0.283	0.655	0.624	0.781	0.497
History prior	0.986	0.805	1.079	1.051	1.078	0.934
GBDT, bank	0.249	0.409	0.693	0.762	0.852	0.637
GBDT, bank + pair-last	0.235	0.320	0.689	0.649	0.805	0.597
GBDT, sufficient statistics	0.208	0.301	0.672	0.643	0.800	0.519

discounted counts of the user's records, of the thread's records, and of the user's co-memberships with the current members. These counts use time constants of 1, 7 and 30 days and the builder's update rules. The strong setting uses $\beta_p = (5, 2.5, 0)$ and the weak setting $(0.15, 0.1, 0.05)$; β_n has a negative 30-day component, without which strong pair effects concentrate activity on a few users. Roles follow the ω-mixture of Section 4 with five kernel rows (last role none, asker or one of three join roles), each drawn from $\mathrm{Dirichlet}(0.5 \cdot \mathbf{1})$. For $\omega = 0$ the oracle uses the Dirichlet posterior mean, for $\omega = 0.5$ the known kernels. The oracle's T-JOIN scores are the log-intensities of the candidates, recomputed by replaying the stream and checked against the generation-time values. Since negatives are drawn uniformly within the joiner's stratum, their softmax is the exact posterior within the candidate set. The plug-in oracle uses the Dirichlet posterior mean of the kernels in every setting. It replaces (β_n, β_p) by their conditional maximum-likelihood estimates on the training T-JOIN queries, a softmax over each candidate set in which the edge term cancels. Its join MRR is within 0.001 of the Bayes optimum in all four settings. Each stream has 200k join records, 3,000 users and about 40k threads over 400 days, with the 70/15/15 split and the stratified 49-negative protocol of the real data. Each stream is generated once (seed 1), and the GBDT and ISM readouts use seeds 7, 11 and 13. The node state includes the transition counts $\mathbf{A}_v$.

Transient streams. The generator produces transient streams. Membership follows recency-weighted copying of past sets and preferential attachment, with new nodes at rate 0.05. Roles follow each node's Markov chain with mixing rate *flip*. With probability ω a member instead repeats the role it held the last time it met one of its current partners. With constraint strength $\kappa > 0$, exactly one member per event takes the leader role c_0. The leader is v with probability proportional to $\exp(\kappa s_v)$, where $s_v = (1 + n_v^{\mathrm{lead}})/(|\mathcal{C}| + n_v)$ is v's smoothed rate of past leader roles over its n_v past events. The other members draw from their chains restricted to the other roles. The constraint itself is exact for every κ; κ sets how strongly the leader choice depends on s_v. The joint law has the form assumed in Proposition 4, with $q_v(c_0) \propto \exp(\kappa s_v)$ and $q_v(c) \propto K_v(c)/(1 - K_v(c_0))$ for $c \neq c_0$, each normalized per member. Every conditional that the generator samples from is a function of the observable strict history, so the Bayes predictive is computed exactly. Each stream has $20{,}000$ events and three roles, split chronologically 60/20/20. Under the exactly-one-leader constraint of strength $\kappa = 8$, a GBDT restricted to each member's own statistics and the event size (not a row of Table 12) loses 0.125 nats against the exact Bayes predictor. This loss lies above the separation bound of Proposition 4(c), which evaluates to 0.064 nats with the same statistics as b_v. With the co-members' leader rates the loss falls to 0.022 nats. At $\kappa = 2$ the leader odds barely vary across members and the bound is 0.001 nats. Table 12 compares four readouts. The history prior uses the member's Laplace-smoothed role frequencies as the predictive. The bank readout applies GBDT to 29 statistics of the member's history, its current partners and the event. Its pair-last variant adds the histogram of the roles the member last held with each current partner and a has-memory flag. The sufficient-statistics readout applies GBDT to the last role, that histogram, the number of partners met, the member's leader rate s_v with summaries of its co-members' rates, the event size and the number of past events. GBDT uses learning rate 0.1 and 100, 200 or 400 iterations selected on validation macro-F1, with the generator seed (0, 1, 2) as model seed. The full feature lists are in the supplementary code.