# MoJEPA: Joint-Embedding Predictive Learning for Dynamic Graphs with Discrete Morse Theory

**Anonymous authors**
Paper under double-blind review

## Abstract

Self-supervised pretraining on dynamic graphs can reduce reliance on task-specific labels and support transfer across datasets. In joint-embedding predictive architectures (JEPAs), the learning task is jointly defined by what is masked and what context remains available for prediction. On dynamic graphs, these choices face two fundamental coupled challenges. First, masking interactions also removes relational evidence needed for prediction. Second, changing interactions limit the context available at each time, leaving useful relations observed at other times unavailable for prediction. To address these challenges, we design *MoJEPA* which combines partial interaction masking with a shared Morse reference forest. We partially mask interactions so that the model can predict missing evidence from the relations that remain visible. To enrich this predictive context, we introduce *Morse reference context*, which leverages discrete Morse theory from low-dimensional topology to reduce the relations observed across a window to a sparse rooted forest. This allows the relations observed at other times to support information exchange at the target time. A relation observed at one time can subsequently support the exchange of node states at another, even when the corresponding interaction is absent. This combination allows prediction to use context beyond the interactions present at an individual time. Our experiments demonstrate that MoJEPA leads on all 9 within-dataset tasks and 15 of 18 transfer directions, with relative gains over the strongest external baseline up to 13.13% and 12.27% for within-dataset prediction and transfer respectively.

## 1 Introduction

Time-varying interactions are often well described by dynamic graph abstractions, ranging from crypto-transactions in finance (Shamsi et al., 2022) to question-answering (Paranjape et al., 2017), and online discussions (Hessel et al., 2016). Capturing the evolution of these relationships supports the prediction of future interactions (You et al., 2022). Self-supervised pretraining leverages observed interactions to learn graph representations for downstream tasks, and therefore reduces dependence on task-specific labels (Chen et al., 2022; Alomrani et al., 2024; Bei et al., 2024). These representations can subsequently be adapted for downstream tasks such as link prediction, and recent results on multi-network pretraining demonstrate potential to transfer temporal graph models to unseen networks (Shamsi et al., 2025).

An emerging framework of Joint-embedding predictive architectures (JEPAs) (LeCun, 2022) learn by predicting the representation of a hidden target from visible context. Unlike reconstruction-based methods, the prediction target is an encoder-produced representation rather than raw input values. Masking and context therefore jointly define this learning task, where the mask determines which information must be inferred, and the context determines what information the predictor can use for inference. I-JEPA of Assran et al. (2023) demonstrates the importance of this combination by using sufficiently large target blocks and an informative, spatially distributed context to learn semantic image representations. JEPA has also been explored for graph-level representation learning (Skenderi et al., 2025) and dynamic-graph representation learning (Zhang et al., 2026). In graph settings, however, masking interactions also alters the relational structure through which nodes receive contextual information. As such, extending JEPA to dynamic graphs introduces two coupled design choices; namely, *which interactions to hide* and *how to construct the context that remains*.

**(1) How can masking preserve relational evidence needed for prediction?** In visual JEPA, target locations remain identifiable through spatial coordinates even when patch content is hidden (Figure 1(a), left). A graph node also retains its identity, but message-passing encoders obtain neighboring states through graph relations (Gilmer et al., 2017). Removing all incident interactions leaves no direct interaction-based context for that node



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.

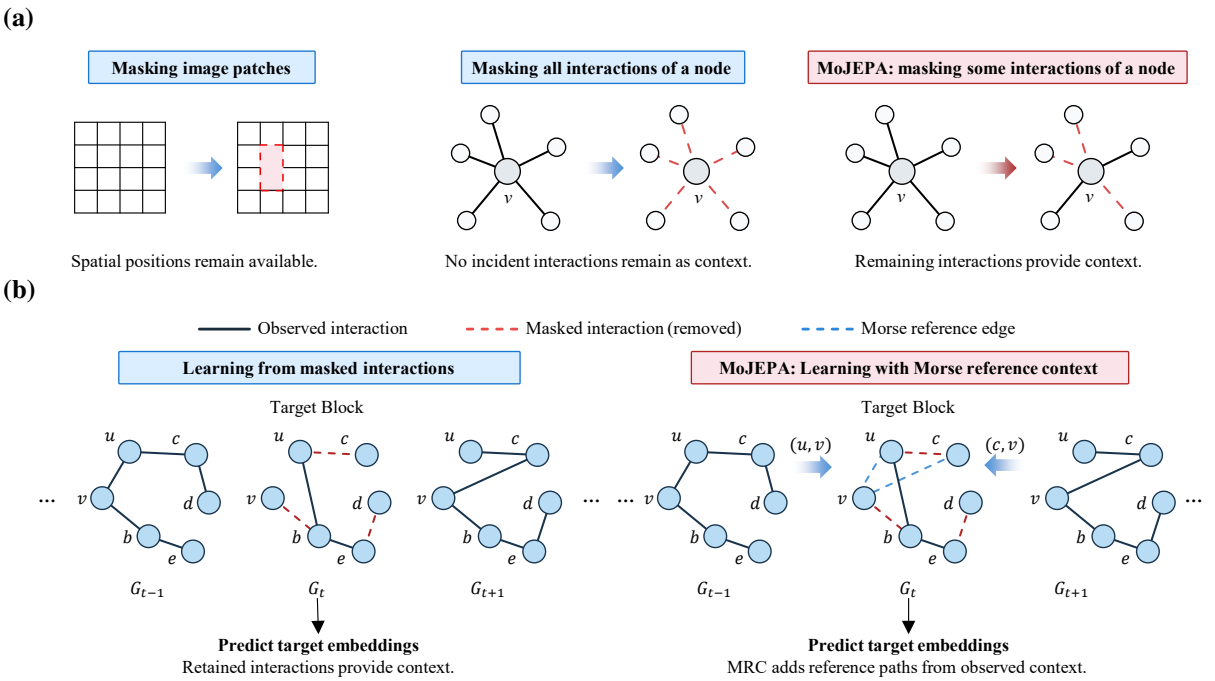



Figure 1: **Masking and context for dynamic-graph JEPA. (a) What remains after masking.** Image patches retain their spatial positions, whereas removing all of a node's incident interactions removes its direct relational context at that time. Partial masking retains interactions as predictive evidence. **(b) Enriching the context for masked prediction.** Both sides use the same observed window and interaction mask. *Morse reference context (MRC)* adds reference paths derived from snapshots outside the target block. The illustrated relations $(u, v)$ and $(c, v)$ provide paths to $v$ at time $t$. For clarity, one target snapshot and selected reference edges are shown. The reference forest is shared across snapshots.

at the target time. Retaining some interactions preserves access to neighboring states, informing target prediction. Masked graph reconstruction similarly uses retained edges to infer withheld structure (Li et al., 2023). So, a useful masking rule must hide interaction evidence while preserving relations through predictor-informing context.

**(2) How can observed relations enrich the context for masked prediction?** The interactions retained at a target time provide only part of the relational evidence available in an observed window. For example, interactions involving $v$ remain visible in the surrounding context snapshots even when they are unavailable in the masked target snapshot (Figure 1(b), left). Snapshot-based encoders can carry node states forward through recurrent updates (You et al., 2022). When spatial message passing uses only the interactions in each snapshot, these updates do not themselves add paths for exchanging node states at the target time. The challenge is to use these observed relations to support state exchange at target times when the corresponding interactions are unavailable, without collapsing the window into a dense aggregate that overrides the masked view.

By leveraging the concepts from low-dimensional and combinatorial topology, we introduce **MoJEPA**, a joint-embedding predictive framework that couples partial interaction masking with Morse reference context (MRC). ***Our central idea is to use relations observed across a temporal window as context for prediction at individual times.*** We partially mask interactions within a contiguous target block and, using discrete Morse theory, construct a reference forest from snapshots outside that block. The student then uses retained interactions and the reference forest to predict complete-view node embeddings. Partial masking determines which interaction evidence is withheld, while MRC organizes the observed relations available to support target prediction.

The introduced notion of MRC then extends this context beyond the interactions present in any single snapshot. Aggregating every relation observed across a window would supply such context, but would swamp few interactions retained at each time and reintroduce what masking may withhold. Discrete Morse theory provides a topological reduction that allows a small set of routes to still reach every needed observed node. Namely, a scalar field on the nodes induces a gradient pairing of nodes with edges, and the paired edges form a rooted forest in

which every node has a unique descending path to a critical node. We use the connection between discrete Morse pairings and rooted forests (Forman, 2002; Chari & Joswig, 2005) to organize observed relations into sparse reference paths guided by a node scalar field. The forest is therefore determined by the field and a fixed tie-breaking rule rather than an arbitrary spanning-tree choice, and enables us compare fields and derive conditional bounds on changes in local gradient pointers in critical edge membership. That is, equivalently, it allows us to characterize how paths respond to perturbed observations. We share this forest across all snapshots and both encoding branches. **A relation observed at one time can then support the exchange of node states at another time, even when the corresponding interaction is absent.** In Figure 1(b), reference edges derived from the surrounding snapshots let $v$ receive the states of $u$ and $c$ at the target time. After pretraining, we adapt the encoder for downstream link prediction. For cross-dataset transfer, the learned encoder is reused while the reference context is constructed from the new dataset's own observations. Our contributions are summarized as follows:

1. **A JEPA pretraining task for dynamic graphs.** We formulate prediction of complete-view node embeddings from partially masked temporal blocks, coupling interaction masking with context construction for representation learning.
2. **Morse reference context for state exchange.** We harness discrete Morse theory from low-dimensional topology to organize observed relations into a sparse forest shared across snapshots and encoding branches, providing paths for exchanging node states when corresponding real interactions are absent.
3. **Within-dataset prediction and cross-dataset transfer.** We evaluate MoJEPA on nine dynamic-graph datasets from three domains and 18 within-domain transfer directions. For transfer, we reuse the source encoder and link head without further training and rebuild the reference context from the target graph's history. MoJEPA achieves the relative gains over the strongest external baseline up to 13.13% and 12.27% for within-dataset prediction and transfer respectively.

## 2 Related Work

**Dynamic graph pretraining.** Pretraining on interaction histories learns temporal and relational representations for downstream tasks (Alomrani et al., 2024; Bei et al., 2024), while training across temporal networks further supports transfer (Shamsi et al., 2025). We study pretraining on a single source graph and transfer both the encoder and link head with learned parameters fixed.

**Joint-embedding predictive learning.** JEPA predicts target embeddings from observed context, with advances in visual learning (LeCun, 2022; Assran et al., 2023; Bardes et al., 2024; Assran et al., 2025) and collapse prevention (Balestriero & LeCun, 2025; Wu et al., 2026b). Graph extensions explore subgraph and temporal prediction (Skenderi et al., 2025; Zhang et al., 2026). MoJEPA focuses on the relational context needed for prediction, combining partial interaction masking with reference paths shared across snapshots.

**Low-dimensional topology in graph learning.** Persistence-based descriptors from topological data analysis are typically used to characterize structural graph organization (Carlsson & Gabrielsson, 2020; Wei, 2024; Su et al., 2025). In contrast, here we use discrete Morse theory (Forman, 2002) and its correspondence between discrete gradients and rooted forests (Chari & Joswig, 2005; Contreras & Tawfeek, 2024) to organize historical relations into sparse paths for information exchange.

## 3 Preliminaries

**Discrete-Time Dynamic Graphs and Link Prediction.** A discrete-time dynamic graph is a sequence $\mathcal{G} = (G_1, \ldots, G_T)$, where each snapshot $G_t = (V_t, E_t, c_t)$ records directed interactions within a time interval. The set $V_t$ contains nodes participating in those interactions, and $E_t \subseteq V_t \times V_t$ contains their ordered endpoint pairs. The count $c_t(u, v)$ records the number of interactions from $u$ to $v$ and is zero when the pair is absent. Node identities remain consistent across snapshots, although the participating nodes and interactions can change over time. Dynamic link prediction ranks candidate directed links $(u, v)$ in snapshot $q$ using the completed history $\mathcal{H}_q = (G_{q-K}, \ldots, G_{q-1})$, where $K$ is the history length. A higher score $s(u, v, q)$ indicates stronger evidence for an interaction from $u$ to $v$. Within-dataset prediction evaluates held-out snapshots of the dataset used for training, whereas cross-dataset transfer evaluates a trained model on another dataset. The candidate sets and temporal splits are specified in Section 5.1.

**Discrete Morse Theory.** Discrete Morse theory describes the structure of a cell complex through gradient pairings and critical cells (Forman, 2002). On an undirected graph $U = (V, \mathcal{E})$, the cells are nodes and edges, of dimension

zero and one, respectively. A discrete gradient can be represented by an acyclic set $\mathcal{P}$ of incident node–edge pairs. Each cell participates in at most one pair, and a pair $(v, \{v, u\})$ directs a gradient step from $v$ to $u$. Acyclicity requires that these steps form no nontrivial closed path. Unpaired nodes and edges are called critical.

**Rooted forests.** The paired edges define $F = (V, \mathcal{E}_F)$, where $\mathcal{E}_F = \{e \in \mathcal{E} : (v, e) \in \mathcal{P} \text{ for some } v\}$. Acyclicity makes $F$ a forest spanning the node set, with one critical node as the root of each tree (Chari & Joswig, 2005; Contreras & Tawfeek, 2024). Conversely, a rooted forest defines such a pairing by pairing each non-root node with its edge toward the root. A connected component of $U$ may contain several trees; obtaining one tree per component requires a pairing with one root in each component. A node scalar field $f : V \to \mathbb{R}$ can guide this construction by pairing each node that has a lower-valued neighbor with an edge to one such neighbor. Values decrease along every gradient step, ensuring acyclicity. This field supplies node priorities, while a discrete Morse function assigns values to both nodes and edges subject to incidence constraints. Appx. B gives the formal definition and the correspondence with rooted forests.

**Joint-Embedding Predictive Architecture.** JEPA learns by predicting target embeddings from an observed context (LeCun, 2022; Assran et al., 2023). Given views $x_{\mathrm{ctx}}$ and $x_{\mathrm{tgt}}$, student encoder $E_\theta$ and predictor $P_\phi$ estimate the output of a teacher encoder $E_{\bar{\theta}}$: $\widehat{z}_{\mathrm{tgt}} = P_\phi(E_\theta(x_{\mathrm{ctx}}), a)$ and $z_{\mathrm{tgt}} = E_{\bar{\theta}}(x_{\mathrm{tgt}})$. Here $a$ specifies the target to predict, e.g., through its position. The student and teacher are also called the context and target encoders, respectively. An embedding discrepancy updates the student and predictor, with gradients stopped through the teacher target. In I-JEPA (Assran et al., 2023) , the teacher parameters follow an exponential moving average (EMA) of the student parameters. The view construction and prediction objective used in MoJEPA are defined in Section 4.

# 4 METHOD

**MoJEPA** consists of two stages (see Figure 2): self-supervised pretraining and downstream link prediction. Pretraining combines partial target-block masking, Morse reference construction, and joint-embedding prediction. A student encoder processes the masked view, while a teacher encoder processes the complete view and supplies the target embeddings. The teacher parameters track an EMA of the student parameters, and both views share the same reference forest. After pretraining, the teacher initializes an encoder that is fine-tuned with a link head. We evaluate link prediction within the same dataset and transfer to a new dataset with the learned parameters fixed.

## 4.1 PARTIAL TARGET-BLOCK MASKING

Partial target-block masking removes sampled interaction records within a contiguous interval, and any retained records provide within-snapshot evidence for prediction. We refer to the pretraining dataset as the source dataset. Given a window $\mathcal{W} = (G_1, \ldots, G_S)$ from its training split, we uniformly sample a length $L \in \{1, \ldots, L_{\max}\}$ and start $a \in \{2, \ldots, S - L\}$, where $L_{\max} \leq S - 2$. The target block is $\mathcal{T} = \{a, \ldots, a + L - 1\}$ and its context is $\mathcal{C} = \{1, \ldots, S\} \setminus \mathcal{T}$. Both boundary snapshots remain observed. Pretraining therefore recovers representations within an observed window, using context on both sides of the target block.

Let $V_t$ denote the nodes participating in interactions at time $t$. For each $t \in \mathcal{T}$, every active node $v$ samples a fraction $\rho$ of its incident directed interaction records $I_t(v)$ without replacement. Each record contains an ordered endpoint pair and its interaction count. The union defines the removed records:

$$M_t = \bigcup_{v \in V_t} \mathrm{Sample}(I_t(v), \lceil \rho |I_t(v)| \rceil), \qquad G_t^{\mathrm{stu}} = G_t \setminus M_t, \qquad G_t^{\mathrm{tea}} = G_t. \tag{1}$$

At context positions, both branches retain $G_t$. At target positions, records sampled separately for each node are removed jointly, so $\rho$ controls per-node sampling rather than the overall removal rate. Appx. E.4 reports the actual removal rates for each dataset.

## 4.2 MORSE REFERENCE CONTEXT

The reference forest supplies a sparse set of context-derived communication routes across the window, which allow nodes exchange current states even when their real interaction is absent or masked. The student and teacher use this shared forest so that their views differ in interaction visibility while retaining the same reference routes.

**Reference construction.** Let $c_t(u, v)$ be the number of directed interactions from $u$ to $v$ in snapshot $G_t$. We aggregate these counts over $\mathcal{C}$ and symmetrize them as $W_{\mathcal{C}} = A_{\mathcal{C}} + A_{\mathcal{C}}^\top$, where $A_{\mathcal{C}}(u, v) = \sum_{t \in \mathcal{C}} c_t(u, v)$. The

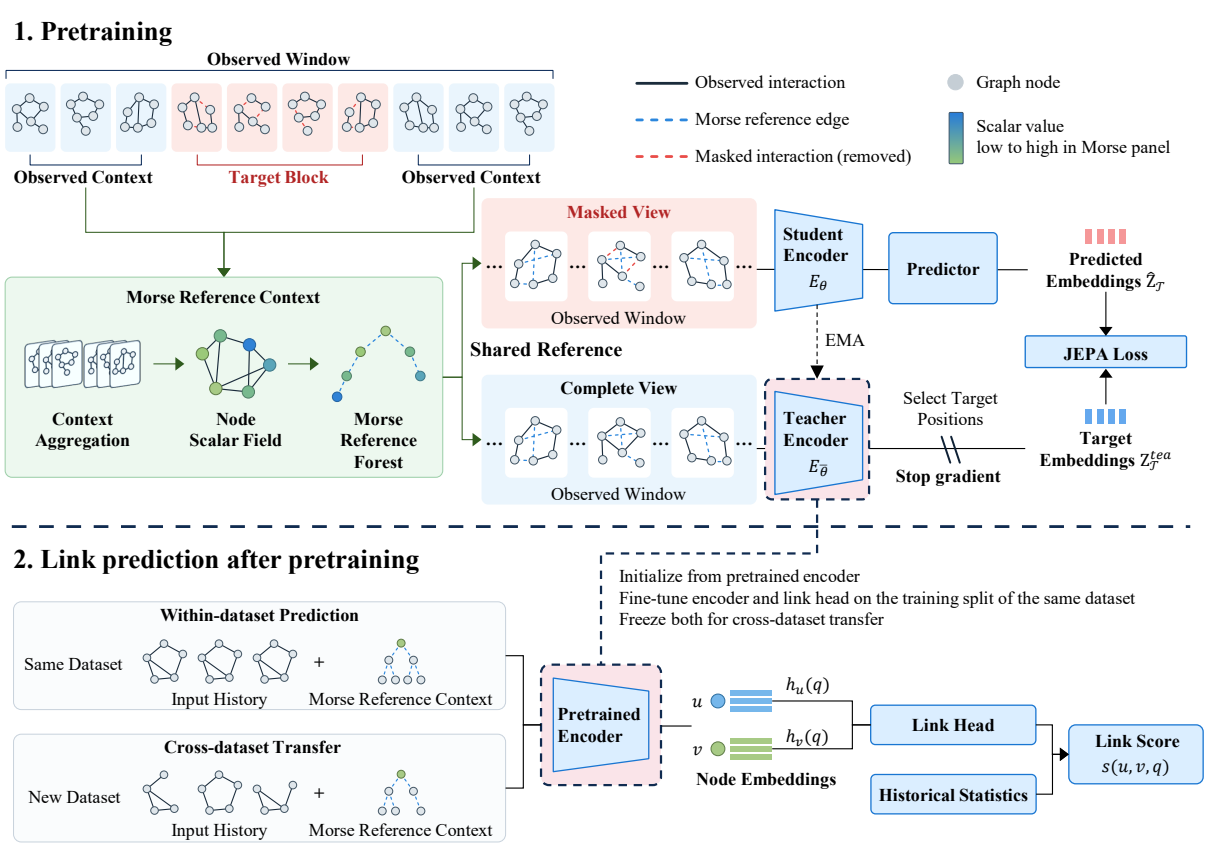



Figure 2: Overview of MoJEPA. Top: pretraining predicts complete-view target embeddings from a masked view under a shared Morse reference forest. Bottom: the pretrained encoder supports within-dataset link prediction and cross-dataset transfer.

positive off-diagonal entries of $W_\mathcal{C}$ define the edges of $U_\mathcal{C} = (V_\mathcal{C}, \mathcal{E}_\mathcal{C})$, where $V_\mathcal{C} = \bigcup_{t\in\mathcal{C}} V_t$. The entire target block is excluded. The forest construction accepts a node scalar field $f_\mathcal{C}$ derived from the observed context, so different choices can guide the pairing. We use the expected random-walk hitting time to the node $h$ with the largest weighted degree in each connected component. This choice gives every other node a neighbor with a lower value, allowing the forest to connect the component to a single root. The field satisfies

$$f_\mathcal{C}(h) = 0, \qquad f_\mathcal{C}(v) = 1 + \sum_u \frac{W_\mathcal{C}(v,u)}{\sum_w W_\mathcal{C}(v,w)} f_\mathcal{C}(u), \quad v \neq h, \tag{2}$$

and ensures that the reference forest connects every observed component using a minimal number of edges.

**Proposition 4.1** (Hitting time yields one root per connected component)**.** *Let $U = (V, \mathcal{E})$ be a finite undirected graph with positive edge weights and nonnegative self-loop weights. Choose one hub per connected component and let $H$ denote the expected random-walk hitting time to that hub. Resolve ties by fixed node identifiers, preserving every strict inequality in $H$. Then the minimum-neighbor construction produces exactly one rooted spanning tree per component. In particular, $|\mathcal{E}_F| = |V| - \beta_0(U)$. (Proof is in Appx. C.)*

Hence, the reference forest preserves component connectivity and supplies $2(|V| - \beta_0(U))$ directed reference arcs per snapshot. Self-loop counts remain in the walk and singleton components have field value zero. For forest selection, ties are resolved using fixed node identifiers, with every strict ordering of the raw field preserved. Following the rooted-forest construction in Section 3, each node with a lower-valued neighbor is paired with an edge to such a neighbor, chosen to minimize the mean of the two endpoint field values. Strict descent makes the resulting node–edge matching acyclic, so it defines a discrete gradient. Its paired edges form $F_\mathcal{C}$, with one tree per component for the hitting-time field. (See Appx. B for correspondence between the discrete gradient and the

forest, Appx. C on the stability of this construction under score perturbations on a fixed graph and graph edits with fixed node scores, and Appx. E.3 on the average reference-forest construction time.) Each forest is computed once and cached for reuse whenever the same historical context and construction configuration recur.

We measure changes in the construction using $d_{\mathrm{Sk}}$, which counts changed vertex pointers and changes in critical-edge membership. Extended pointers distinguish roots from absent vertices. For a graph $U$ and comparison scores $g$, let $m_v^U(g)$ be the gap between the two smallest scores on the closed neighborhood $N_U[v]$, taking $m_v^U(g) = +\infty$ for isolated vertices. Define the neighborhoods not certified stable under $g \to g'$ by

$$\mathcal{H}_U(g, g') = \left\{ v \in V : \max_{u \in N_U[v]} |g'(u) - g(u)| \geq \frac{m_v^U(g)}{2} \right\}.$$

**Theorem 4.2** (Combined graph–score stability)**.** *Let $U = (V, \mathcal{E})$ and $U' = (V', \mathcal{E}')$ be finite simple graphs, and let $g, g'$ be injective comparison scores on a common vertex catalog containing $V \cup V'$. Under the minimum-neighbor construction,*

$$d_{\mathrm{Sk}}(\mathcal{S}(U, g), \mathcal{S}(U', g')) \leq |V \triangle V'| + 2|\mathcal{E} \triangle \mathcal{E}'| + 3 \min\{|\mathcal{H}_U(g, g')|, |\mathcal{H}_{U'}(g, g')|\}.$$

*(Proof is in Appx. C.)*

The bound separates support edits from score-induced changes. When hitting times are recomputed, stability depends on the resulting score changes relative to local margins. Our construction allows to compare scalar fields and derive conditional bounds on changes in local gradient pointers and critical-edge membership under perturbed observations.

**Reference augmentation.** To distinguish context-derived routes from current interactions, we assign them separate relation types. Each forest edge becomes two directed reference arcs in every snapshot. Let $\phi_{uv,t}$ denote the transformed attributes of a real interaction. We encode the two relation types as

$$\psi_{uv,t}^{\mathrm{real}} = [\phi_{uv,t}; 0], \qquad \psi_{uv}^{\mathrm{ref}} = [\mathbf{0}; 1], \qquad \widetilde{G}_t^b = G_t^b \oplus F_{\mathcal{C}}, \quad b \in \{\mathrm{stu}, \mathrm{tea}\}. \tag{3}$$

Windows $\widetilde{\mathcal{W}}^{\mathrm{stu}}$ and $\widetilde{\mathcal{W}}^{\mathrm{tea}}$ are the masked view and complete view in Figure 2. Both contain all window positions and share $F_{\mathcal{C}}$. The operator $\oplus$ retains both records when their endpoints coincide. Reference arcs carry messages even through currently inactive nodes, but do not contribute to real-interaction statistics, activity, or masking.

### 4.3 Reference-Conditioned Joint-Embedding Prediction

**Spatiotemporal encoding.** Spatial aggregation exchanges states along real and reference relations, while temporal encoding integrates each node's observations across the window. For each view, we compute node features from its retained real interactions, including incoming and outgoing interaction counts, neighbor counts, and an activity indicator. The encoder projects these features into hidden embeddings and applies directed graph isomorphism layers with edge features (GINE) that aggregate incoming and outgoing messages separately over real and reference arcs. A bidirectional gated recurrent unit (GRU) applied to each node then integrates the spatial embeddings with sinusoidal position encodings. Feature transformations are specified in Appx. E.1.

**Embedding prediction.** For each node at a target position, the predictor estimates the embedding that the teacher computes for that node at the same position from the complete view. The student and EMA teacher produce $Z^{\mathrm{stu}} = E_\theta(\widetilde{\mathcal{W}}^{\mathrm{stu}})$ and $Z^{\mathrm{tea}} = E_{\bar{\theta}}(\widetilde{\mathcal{W}}^{\mathrm{tea}})$. For each target position $t$, we combine a learned query with the position encoding to obtain the target-position query $q_t$. The query cross-attends to the student embeddings of node $v$ across the entire window. The predictor produces $\widehat{z}_{v,t} = P_\phi(Z_{v,:}^{\mathrm{stu}}, q_t)$. We compute the loss for nodes observed in both the context and the target snapshot, giving $V_t^+ = V_{\mathcal{C}} \cap V_t$. The valid target positions are $\mathcal{T}_+ = \{t \in \mathcal{T} : |V_t^+| > 0\}$. The collections of predicted and target embeddings are denoted by $\widehat{Z}_{\mathcal{T}}$ and $Z_{\mathcal{T}}^{\mathrm{tea}}$ in Figure 2. For windows with $\mathcal{T}_+ \neq \varnothing$, the objective is

$$\mathcal{L}_{\mathrm{JEPA}} = \frac{1}{|\mathcal{T}_+|} \sum_{t \in \mathcal{T}_+} \frac{1}{|V_t^+|} \sum_{v \in V_t^+} \left\| \mathrm{norm}(\widehat{z}_{v,t}) - \mathrm{sg}\big[\mathrm{norm}(Z_{v,t}^{\mathrm{tea}})\big] \right\|_2^2. \tag{4}$$

Here $\mathrm{norm}$ denotes stabilized $\ell_2$ normalization and $\mathrm{sg}$ stops gradients through the teacher target. Gradients update the student and predictor, while $\bar{\theta} \leftarrow \tau\bar{\theta} + (1-\tau)\theta$ updates the teacher. When $\mathcal{T}_+ = \varnothing$, the window contributes no gradient and the student and predictor parameters remain unchanged. The iteration still counts toward the pretraining budget, and the teacher receives its EMA update. The window is not resampled.

### 4.4 Link Prediction and Cross-Dataset Transfer

**Supervised fine-tuning.** We initialize the downstream encoder with the final EMA teacher and jointly fine-tune it with a link head on the training split of the source dataset. To rank candidate links in snapshot $q$, we rebuild the forest from the completed history $\mathcal{H}_q = (G_{q-K}, \ldots, G_{q-1})$ and augment each history frame. We use the encoder output for node $v$ at the final history position as its embedding $h_v(q)$. An MLP applied to the ordered concatenation $[h_u(q); h_v(q)]$ produces the directed link logit $g(u, v, q)$. Pretraining masks and the predictor are discarded. The encoder and link head are jointly fine-tuned on source-training snapshots using binary cross-entropy, averaged over candidate destinations for each query node and then over query nodes. Candidate construction follows the new-link task in Appx. F.4. We select the checkpoint by the neural model's MRR on the source validation set.

**Scoring and transfer.** The link score combines the link head with historical statistics on observed interactions:

$$s(u, v, q) = \langle \pi_u(q), \pi_v(q) \rangle + \eta \, a(u, v, q) \, \sigma(g(u, v, q)). \tag{5}$$

Here $\pi_v(q)$ summarizes the historical neighborhood of $v$, $a(u, v, q)$ weights the neural score by endpoint recency, and $\sigma$ is the sigmoid function. These statistics use real observations before $q$. (see Appx. E.1 for more details). We select $\eta$ on source validation data after fixing the neural checkpoint.

For within-dataset prediction, we evaluate the adapted model on the held-out split of the same dataset. For cross-dataset transfer, we apply it to a new dataset while fixing the source encoder, link head, feature scaling, and $\eta$. At each target prediction time, the target's own completed observations supply its history forest, structural vectors, and recency weights. Source training and validation precede the target evaluation interval.

## 5 Experiments

We evaluate within-dataset prediction and cross-dataset transfer, compare node scalar fields for reference construction, and examine observation robustness and training budgets.

### 5.1 Experimental Setup

**Data and tasks.** We use nine daily directed graphs from three domains: Token (Bytom/BY, Decentraland/DE, Golem/GO) (Shamsi et al., 2022), QA (MathOverflow/MO, AskUbuntu/AU, SuperUser/SU) (Paranjape et al., 2017), and Reddit (gaming/Ga, politics/Po, movies/Mv) (Hessel et al., 2016). Within-dataset prediction uses chronological training, validation, and test splits. Transfer covers all 18 ordered pairs within domains, with source training and validation preceding target testing on a common calendar. (See Apps. F.4– F.5 for details on the splits and candidate rules and Appx. D on graph summaries.) **Comparisons and evaluation.** Our baselines include EGCH (Pareja et al., 2020), ROLAND (You et al., 2022), SiST-GNN (Roy & Dasgupta, 2026), WinGNN (Zhu et al., 2023), HawGCN (Qi et al., 2025), UnderGS (Wu et al., 2026a), EAGLE (Li et al., 2025b), TNCN (Zhang et al., 2025), SG-JEPA (Zhang et al., 2026), CPDG (Bei et al., 2024), and TMetaNet (Li et al., 2025a) (see Appx. F.2 for detail on settings). MoJEPA and SG-JEPA (Zhang et al., 2026) use 6,000 pretraining iterations and at most 2,000 fine-tuning updates, with checkpoints selected on source validation data. We report MRR ($\times 100$), as mean $\pm$ sample standard deviation over three seeds. **Bold** marks the highest complete mean in each column. $^{*}$, $^{**}$, $^{***}$ indicate significant, statistically significant and highly statistical significant improvement assessed over one-tailed hypothesis (Holm-adjusted) ($p$-value $< 0.10$, $p$-value $< 0.05$, $p$-value $< 0.01$), respectively.

### 5.2 Main Results

Table 1 group within-dataset prediction and transfer by domain, allowing performance on each source graph to be read alongside its two outgoing transfers.

**Prediction across domains.** MoJEPA achieves the highest mean MRR on all nine within-dataset tasks, with an average relative gain of 7.34% over the strongest external baseline in each task. CPDG is the strongest external baseline on Token, EAGLE on QA, and SG-JEPA on Reddit. MoJEPA outperforms all three in their strongest domains and exceeds both pretrained baselines, SG-JEPA and CPDG, across all three domains. We next examine whether this advantage carries over to another dataset.

**Transfer with fixed source models.** MoJEPA retains a prediction advantage when the source encoder, link head, and calibration are applied to a different graph without target-side training. It leads all transfer directions on Token

Table 1: Within-dataset prediction and cross-dataset transfer on Token, QA, and Reddit. In each domain, the first three result columns report within-dataset prediction and the remaining six report transfer.

| **Token** | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| Method | BY | DE | GO | BY→DE | BY→GO | DE→BY | DE→GO | GO→BY | GO→DE |
| EGCH | 66.35±0.29 | 46.39±2.54 | 47.42±1.07 | 46.21±2.03 | 45.75±1.88 | 40.01±27.12 | 28.39±17.16 | 62.97±2.30 | 46.04±1.20 |
| ROLAND | 71.49±1.28 | 49.16±11.87 | 40.42±8.66 | 34.20±22.73 | 31.24±21.75 | 26.67±26.53 | 21.84±18.17 | 58.10±2.37 | 44.25±1.42 |
| SiST-GNN | 65.36±2.24 | 46.64±2.47 | 34.36±1.80 | 62.09±0.19 | 63.43±0.33 | 73.63±0.53 | 61.36±0.94 | 72.38±1.82 | 60.47±1.59 |
| WinGNN | 50.99±0.25 | 18.92±17.38 | 17.78±15.64 | 15.47±11.39 | 15.56±11.79 | 32.82±20.76 | 25.85±14.82 | 24.42±26.60 | 20.02±19.10 |
| HawGCN | 66.78±0.21 | 50.75±0.38 | 57.06±0.22 | 51.40±0.00 | 54.04±0.00 | 62.00±0.00 | 54.01±0.00 | 62.00±0.00 | 51.42±0.01 |
| UnderGS | 57.20±4.58 | 43.50±1.09 | 40.33±9.70 | 44.50±8.13 | 43.25±9.81 | 50.27±2.98 | 42.23±1.37 | 53.22±6.51 | 43.64±6.45 |
| EAGLE | 75.93±0.16 | 64.87±0.05 | 64.40±0.57 | 62.90±0.16 | 62.47±0.25 | 74.53±0.17 | 61.30±0.53 | 72.95±0.17 | 61.25±0.25 |
| TNCN | 68.87±1.34 | 60.67±2.34 | 61.25±1.00 | 56.32±1.30 | 59.61±1.19 | 63.53±1.33 | 62.99±1.48 | 66.97±0.21 | 55.83±1.19 |
| SG-JEPA | 77.25±1.03 | 65.35±0.97 | 64.79±0.23 | 65.08±0.27 | 61.50±0.69 | 73.17±0.79 | 57.96±2.03 | 74.19±1.17 | 63.16±1.38 |
| CPDG | 79.21±0.39 | 68.85±0.32 | 65.93±0.59 | 67.07±0.79 | 66.43±0.58 | 78.18±0.14 | 65.06±0.29 | 77.29±0.46 | 66.16±1.12 |
| TMetaNet | 59.73±0.64 | 45.28±0.39 | 56.36±0.78 | 49.68±0.58 | 48.63±1.02 | 50.35±0.11 | 52.36±0.76 | 55.33±0.23 | 40.26±0.77 |
| MoJEPA | **84.37±0.12**$^{***}$ | **75.45±0.76**$^{**}$ | **74.59±0.57**$^{***}$ | **74.43±0.16**$^{**}$ | **74.46±0.51**$^{***}$ | **81.47±0.35**$^{**}$ | **73.05±0.41**$^{***}$ | **81.13±0.62**$^{**}$ | **72.10±0.80**$^{**}$ |

| **QA** | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| Method | MO | AU | SU | MO→AU | MO→SU | AU→MO | AU→SU | SU→MO | SU→AU |
| EGCH | 32.49±0.19 | 43.27±0.57 | 41.12±1.00 | 43.59±0.62 | 41.82±0.88 | 32.90±0.32 | 41.76±0.03 | 33.13±0.62 | 43.54±0.11 |
| ROLAND | 27.15±2.81 | 28.27±7.32 | 29.28±5.74 | 34.88±5.02 | 31.55±5.77 | 25.64±6.50 | 28.16±8.30 | 21.41±8.33 | 23.53±12.00 |
| SiST-GNN | 38.45±0.22 | 47.46±0.54 | 44.37±0.54 | 49.06±1.03 | 46.15±0.79 | 34.67±0.77 | 46.57±0.43 | 35.59±0.36 | 49.34±0.71 |
| WinGNN | 28.83±0.43 | 34.42±0.18 | 34.54±0.08 | 33.77±0.08 | 34.06±0.04 | 27.00±0.71 | 34.03±0.28 | 28.40±0.09 | 34.26±0.08 |
| HawGCN | 28.24±1.22 | 42.03±0.58 | 37.57±0.42 | 41.46±0.96 | 37.57±0.26 | 26.87±0.30 | 34.84±0.62 | 26.61±0.44 | 39.12±0.81 |
| UnderGS | 30.74±0.28 | 43.48±0.69 | 40.07±1.23 | 40.50±1.70 | 38.36±1.24 | 30.87±0.21 | 40.15±0.65 | 30.96±0.62 | 43.27±1.14 |
| EAGLE | 50.87±0.26 | 54.67±0.33 | 54.22±0.26 | 55.94±0.21 | 56.11±0.24 | 50.48±0.33 | 54.93±0.78 | 50.60±0.40 | 54.75±0.73 |
| TNCN | 50.59±0.26 | 51.46±0.24 | 50.57±0.65 | 46.02±3.40 | 45.69±3.52 | 45.87±0.55 | 49.70±1.44 | 45.04±3.19 | 50.11±2.22 |
| SG-JEPA | 36.76±0.36 | 50.51±0.09 | 47.62±0.07 | 49.22±0.45 | 45.70±0.41 | 35.37±0.15 | 46.00±0.20 | 36.22±0.18 | 50.50±0.34 |
| CPDG | 38.35±0.53 | 52.23±0.10 | 49.89±0.45 | 50.36±0.55 | 47.66±0.60 | 37.41±0.35 | 49.10±0.26 | 38.35±0.10 | 52.50±0.15 |
| TMetaNet | 32.28±0.49 | 40.36±1.52 | 39.87±0.22 | 37.12±0.73 | 34.11±0.59 | 30.74±0.98 | 36.15±0.84 | 29.16±0.88 | 29.99±0.13 |
| MoJEPA | **52.87±0.12**$^{**}$ | **59.91±0.10**$^{***}$ | **59.39±0.07**$^{***}$ | **58.51±0.19**$^{***}$ | **58.81±0.08**$^{**}$ | **51.06±0.19** | **58.97±0.17**$^{**}$ | **52.10±0.30**$^{**}$ | **60.01±0.15**$^{**}$ |

| **Reddit** | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| Method | Ga | Po | Mv | Ga→Po | Ga→Mv | Po→Ga | Po→Mv | Mv→Ga | Mv→Po |
| EGCH | 21.17±21.26 | 21.75±22.54 | 47.50±0.56 | 33.67±21.61 | 34.91±22.55 | 45.74±0.50 | 48.33±0.78 | 32.60±20.54 | 33.36±21.35 |
| ROLAND | 34.64±7.81 | 39.15±6.38 | 46.06±5.76 | 28.14±20.88 | 28.94±22.50 | 27.73±13.59 | 29.89±14.38 | 29.65±13.57 | 29.74±15.34 |
| SiST-GNN | 48.48±0.31 | 51.30±0.11 | 52.12±1.18 | 53.25±0.28 | **56.58±0.10** | **52.65±0.17** | **57.49±0.19** | 51.59±0.36 | 52.54±0.29 |
| WinGNN | 41.72±0.30 | 38.77±3.17 | 41.37±4.36 | 38.26±3.83 | 40.59±4.00 | 41.58±1.26 | 43.79±1.33 | 40.83±0.55 | 40.52±0.77 |
| HawGCN | 39.62±0.01 | 41.53±0.02 | 41.56±0.19 | 41.57±0.00 | 42.36±0.00 | 37.24±0.01 | 41.71±0.01 | 39.47±0.00 | 42.78±0.00 |
| UnderGS | 39.70±0.47 | 47.88±0.20 | 44.98±0.65 | 45.18±0.70 | 45.63±0.78 | 40.50±0.49 | 47.19±0.60 | 38.66±0.44 | 44.44±0.33 |
| EAGLE | 49.54±0.20 | 51.80±0.11 | 53.94±0.08 | 48.55±0.63 | 52.15±0.28 | 49.36±0.15 | 54.19±0.19 | 49.15±0.18 | 51.58±0.11 |
| TNCN | 44.45±0.28 | 49.82±0.39 | 49.90±0.20 | 47.14±0.38 | 48.72±0.84 | 40.99±2.58 | 47.20±2.12 | 43.99±0.75 | 48.30±0.84 |
| SG-JEPA | 50.91±0.23 | 52.89±0.06 | 54.73±0.13 | 52.40±0.40 | 54.34±0.44 | 49.07±0.63 | 53.54±0.27 | 49.72±0.60 | 52.09±0.51 |
| CPDG | 49.86±0.12 | 51.74±0.17 | 53.95±0.11 | 50.53±0.17 | 53.85±0.25 | 48.18±0.63 | 53.45±0.29 | 48.97±0.53 | 50.81±0.12 |
| TMetaNet | 38.90±0.71 | 42.37±0.73 | 47.94±0.77 | 38.78±0.49 | 42.63±0.28 | 37.18±0.44 | 33.57±0.38 | 41.55±0.87 | 36.13±0.56 |
| MoJEPA | **53.36±0.06**$^{**}$ | **55.28±0.03**$^{***}$ | **57.17±0.06**$^{***}$ | **54.24±0.17**$^{**}$ | 56.17±0.22 | 52.60±0.06 | 56.73±0.13 | **52.70±0.06**$^{*}$ | **54.39±0.10**$^{**}$ |

and QA and remains competitive on Reddit, where SiST-GNN performs better on some source–target pairs. Across the 18 transfer directions, MoJEPA achieves an average relative gain of 4.96% over the strongest external baseline in each direction. During transfer, reference context is reconstructed from the target graph's observed history while the learned parameters remain fixed. The source-trained encoder and scoring functions therefore remain effective when applied to a new graph within the same domain.

**Ablation Studies.** MoJEPA supports different node scalar fields for constructing Morse reference forests. Table 2 compares five such configurations with No forest, which uses the same JEPA task and encoder without reference arcs. No partial masking means removing all observed interactions from the selected target blocks while keeping the context blocks unchanged. Hitting time is the configuration used in the main experiments. All conditions use the same observed context, training budget, and downstream evaluation. We average within-dataset and transfer results separately within each domain. Appx. E.2 defines the fields and pairing rules.

**Effect of reference context.** On Token, every tested forest configuration improves on No forest in both within-dataset prediction and transfer. The gains across all five scalar fields show that shared historical reference context benefits both prediction and transfer in this domain. The gains on QA are smaller, while Reddit shows no consistent improvement over No forest.

**Choice of scalar field.** The scalar field also affects performance. On Token, hitting time obtains the highest mean in both settings, followed by hub distance, while the strength and net-flow fields yield smaller gains. Their relative ordering is largely preserved in transfer, so the benefit of a reference construction extends to prediction on another graph. The fields are less separated on QA, and hitting time does not lead on Reddit. Hitting time is the

strongest tested choice on Token, but no scalar field leads across all domains. MoJEPA can use any of these fields to construct its reference context.

Table 2: Reference-context ablation. W: within-dataset; T: transfer. MRR ($\times 100$), mean $\pm$ sample standard deviation. All conditions use 6,000 pretraining iterations and at most 2,000 fine-tuning updates.

| Scalar field / control | Token W | Token T | QA W | QA T | Reddit W | Reddit T |
|---|---|---|---|---|---|---|
| No partial masking | $72.08 \pm 0.34$ | $70.26 \pm 0.51$ | $55.39 \pm 0.08$ | $54.94 \pm 0.13$ | $54.04 \pm 0.14$ | $53.48 \pm 0.09$ |
| No forest | $76.26 \pm 0.51$ | $73.78 \pm 0.41$ | $57.22 \pm 0.13$ | $56.40 \pm 0.22$ | $55.40 \pm 0.03$ | $\mathbf{54.65 \pm 0.04}$ |
| Hub distance | $77.80 \pm 0.33$ | $75.89 \pm 0.28$ | $57.37 \pm 0.07$ | $56.54 \pm 0.11$ | $55.31 \pm 0.04$ | $54.41 \pm 0.13$ |
| Negative strength | $77.69 \pm 0.54$ | $75.51 \pm 0.18$ | $\mathbf{57.39 \pm 0.04}$ | $56.49 \pm 0.12$ | $55.32 \pm 0.05$ | $54.41 \pm 0.13$ |
| Negative net flow | $77.04 \pm 0.24$ | $74.30 \pm 0.12$ | $57.31 \pm 0.07$ | $56.49 \pm 0.07$ | $55.24 \pm 0.02$ | $54.38 \pm 0.08$ |
| Positive strength | $77.25 \pm 0.18$ | $74.26 \pm 0.34$ | $57.28 \pm 0.03$ | $56.40 \pm 0.13$ | $\mathbf{55.41 \pm 0.01}$ | $54.63 \pm 0.04$ |
| Hitting time | $\mathbf{78.14 \pm 0.47}$ | $\mathbf{76.11 \pm 0.29}^{**}$ | $57.39 \pm 0.08$ | $\mathbf{56.58 \pm 0.07}^{*}$ | $55.27 \pm 0.03$ | $54.47 \pm 0.05$ |

**Robustness.** We evaluate MoJEPA against its next best competitors SG-JEPA, and EAGLE under relation deletion, spurious relation addition, and one-day delays in the observed history. Corruption rates are 0, 5, 10, 20, 30, and 50%. Models and calibration are fixed after clean-data training, while reference forests and other history-derived inputs are rebuilt from corrupted observations. Queries and labels remain unchanged. Figure 3 reports domain-averaged MRR on Reddit. (See Appx. F.7 for additional detail.) We find that MoJEPA maintains the highest domain-averaged MRR among the three methods across all three perturbations and all tested rates on Reddit. This advantage persists when reference forests and historical inputs are rebuilt from corrupted observations. Delay produces the largest decline for MoJEPA, indicating that the timing of observed interactions matters for prediction. However, higher MRR does not imply a uniformly smaller decline from clean performance.

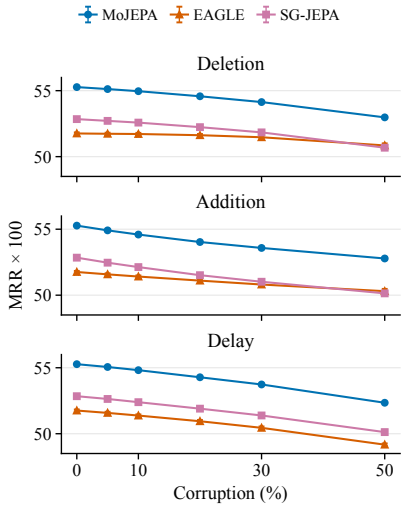



Figure 3: Observation robustness on Reddit under deletion, addition, and delay.

**Context-Graph Structure.** The aggregated context graphs differ in connectivity and cycle rank (Appx. H). For the hitting-time construction, the reference forest contains $|V_{\mathcal{C}}| - \beta_0(U_{\mathcal{C}})$ edges and omits $\beta_1(U_{\mathcal{C}})$ support edges, preserving connected components but not cycle homology. Across seven-day training histories matching the downstream context construction, the median retained-edge fractions are 88–94% on Token, 66–73% on QA, and 44–62% on Reddit. These summaries characterize downstream-style contexts; pretraining uses the observed positions remaining after an internal target block is excluded. Scalar fields producing additional roots yield smaller, more fragmented forests, while fields selecting equally many edges can still provide different reference routes. The reference-context ablation shows the largest benefit on Token, but we note that the structural summaries alone do not establish why the performance effects differ across domains.

Appx. G reports ablations of pretraining and score fusion, observation robustness on Token and QA under deletion, addition, and delay, and training-budget experiments across all three domains.

## 6 Discussion, Limitations and Future Directions

We have developed MoJEPA, a framework for learning transferable representations across dynamic graphs through joint-embedding prediction. Leveraging machinery of low-dimensional and combinatorial topology, MoJEPA integrates partial interaction masking with a shared Morse reference forest, allowing historical relations to support prediction when those interactions are absent at the target time. Our experiments on 9 diverse datasets have demonstrated superior MoJEPA performance both for within-dataset link prediction and transfer tasks. As the first step toward bringing the machinery of low-dimensional topology to shared latent representations and JEPA, MoJEPA is yet limited to the explored topological regimes. Given its high premise, in the future we advance MoJEPA to a wider range of regimes and higher-dimensional skeletons.

### AI USE STATEMENT

We used OpenAI Codex to assist with code implementation and manuscript drafting. The authors are responsible for verifying AI-assisted content and for the correctness and originality of the final text, code, claims, and analyses.

### REPRODUCIBILITY STATEMENT

Section 4 describes the model and learning procedure. Apps. B and C provide the theoretical background, assumptions, and proofs. Appx. E details the encoder, historical statistics, and scalar fields used to construct reference forests. Appx. F documents graph construction, temporal splits, implementation Details, baseline configurations, candidate sampling, evaluation aggregation, and observation corruption. Appx. G reports additional robustness and training-budget experiments. Together, these descriptions specify the method and evaluation procedures used in this study.

### ETHICS STATEMENT

We use existing interaction datasets to evaluate link prediction and report aggregate results without seeking to identify individuals or infer sensitive personal attributes. Use of these data should respect their original access conditions, applicable licenses, and privacy constraints.

## REFERENCES

Mohammad Ali Alomrani, Mahdi Biparva, Yingxue Zhang, and Mark Coates. DyG2Vec: Efficient representation learning for dynamic graphs. *Trans. Mach. Learn. Res.*, 2024, 2024.

Mahmoud Assran, Quentin Duval, Ishan Misra, Piotr Bojanowski, Pascal Vincent, Michael Rabbat, Yann LeCun, and Nicolas Ballas. Self-supervised learning from images with a joint-embedding predictive architecture. In *2023 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2023.

Mido Assran, Adrien Bardes, David Fan, Quentin Garrido, Russell Howes, Mojtaba, Komeili, Matthew Muckley, Ammar Rizvi, Claire Roberts, Koustuv Sinha, Artem Zholus, Sergio Arnaud, Abha Gejji, Ada Martin, Francois Robert Hogan, Daniel Dugas, Piotr Bojanowski, Vasil Khalidov, Patrick Labatut, Francisco Massa, Marc Szafraniec, Kapil Krishnakumar, Yong Li, Xiaodong Ma, Sarath Chandar, Franziska Meier, Yann LeCun, Michael Rabbat, and Nicolas Ballas. V-jepa 2: Self-supervised video models enable understanding, prediction and planning, 2025. URL https://arxiv.org/abs/2506.09985.

Randall Balestriero and Yann LeCun. LeJEPA: Provable and scalable self-supervised learning without the heuristics. *arXiv preprint arXiv:2511.08544*, 2025.

Adrien Bardes, Quentin Garrido, Jean Ponce, Xinlei Chen, Michael Rabbat, Yann LeCun, Mahmoud Assran, and Nicolas Ballas. Revisiting feature prediction for learning visual representations from video, 2024.

Ulrich Bauer. *Persistence in discrete Morse theory*. PhD thesis, Georg-August-Universität Göttingen, July 2011.

Yuanchen Bei, Hao Xu, Sheng Zhou, Huixuan Chi, Haishuai Wang, Mengdi Zhang, Zhao Li, and Jiajun Bu. CPDG: a contrastive pre-training method for dynamic graph neural networks. In *2024 IEEE 40th international conference on data engineering (ICDE)*, 2024.

Gunnar Carlsson and Rickard Brüel Gabrielsson. Topological approaches to deep learning. In *Topological Data Analysis*, pp. 119–146. Springer, 2020.

Gunnar Carlsson, Vin De Silva, and Dmitriy Morozov. Zigzag persistent homology and real-valued functions. In *Proceedings of the Symposium on Computational Geometry*, 2009.

Michael J Catanzaro, Justin M Curry, Brittany Terese Fasy, Jānis Lazovskis, Greg Malen, Hans Riess, Bei Wang, and Matthew Zabka. Moduli spaces of morse functions for persistence. *Journal of Applied and Computational Topology*, 4(3):353–385, 2020.

Manoj K Chari and Michael Joswig. Complexes of discrete morse functions. *Discrete mathematics*, 302(1-3): 39–51, 2005.

Ke-Jia Chen, Jiajun Zhang, Linpu Jiang, Yunyun Wang, and Yuxuan Dai. Pre-training on dynamic graph neural networks. *Neurocomputing*, 500:679–687, 2022.

David Cohen-Steiner, Herbert Edelsbrunner, and John Harer. Stability of persistence diagrams. *Discrete & Computational Geometry*, 37(1):103–120, 2007.

Ivan Contreras and Andrew Tawfeek. On discrete gradient vector fields and Laplacians of simplicial complexes. *Annals of Combinatorics*, 28(1):67–91, 2024.

Herbert Edelsbrunner and John Harer. *Computational Topology: An Introduction*. American Mathematical Soc., 2010.

Robin Forman. A user's guide to discrete Morse theory. *Séminaire Lotharingien de Combinatoire*, 48:B48c–35, 2002.

Justin Gilmer, Samuel S Schoenholz, Patrick F Riley, Oriol Vinyals, and George E Dahl. Neural message passing for quantum chemistry. In *International Conference on Machine Learning*, pp. 1263–1272, 2017.

Allen Hatcher. *Algebraic Topology*. Cambridge University Press, 2002.

Jack Hessel, Chenhao Tan, and Lillian Lee. Science, askscience, and badscience: On the coexistence of highly related communities. In *AAAI conference on Web and Social Media*, 2016.

Zhenyu Hou, Xiao Liu, Yukuo Cen, Yuxiao Dong, Hongxia Yang, Chunjie Wang, and Jie Tang. GraphMAE: Self-supervised masked graph autoencoders, 2022.

Zhenyu Hou, Yufei He, Yukuo Cen, Xiao Liu, Yuxiao Dong, Evgeny Kharlamov, and Jie Tang. GraphMAE2: A decoding-enhanced masked self-supervised graph learner. In *ACM Web Conference*, 2023.

Harish Kannan, Emil Saucan, Indrava Roy, and Areejit Samal. Persistent homology of unweighted complex networks via discrete morse theory. *Scientific Reports*, 9(1), 9 2019. ISSN 2045-2322. doi: 10.1038/s41598-019-50202-3. URL http://dx.doi.org/10.1038/s41598-019-50202-3.

Yann LeCun. A path towards autonomous machine intelligence. OpenReview, 2022. URL https://openreview.net/pdf?id=BZ5a1r-kVsf. Version 0.9.2.

Hao Li, Hao Wan, Yuzhou Chen, Dongsheng Ye, Yulia Gel, and Hao Jiang. TMetaNet: Topological meta-learning framework for dynamic link prediction. In *International Conference on Machine Learning*, 2025a.

Haoyang Li, Yuming Xu, Yiming Li, Hanmo Liu, Darian Li, Chen Jason Zhang, Lei Chen, and Qing Li. When speed meets accuracy: an efficient and effective graph model for temporal link prediction. *Proc. VLDB Endow.*, 2025b.

Jintang Li, Ruofan Wu, Wangbin Sun, Liang Chen, Sheng Tian, Liang Zhu, Changhua Meng, Zibin Zheng, and Weiqiang Wang. What's behind the mask: Understanding masked graph modeling for graph autoencoders. In *ACM SIGKDD*, 2023.

J. Milnor, M. Spivak, and R. Wells. *Morse Theory*. Princeton University Press, 1969. ISBN 9780691080086. URL http://www.jstor.org/stable/j.ctv3f8rb6.

Ashwin Paranjape, Austin R Benson, and Jure Leskovec. Motifs in temporal networks. In *ACM International Conference on Web Search and Data Mining*, 2017.

Aldo Pareja, Giacomo Domeniconi, Jie Chen, Tengfei Ma, Toyotaro Suzumura, Hiroki Kanezashi, Tim Kaler, Tao Schardl, and Charles Leiserson. Evolvegcn: Evolving graph convolutional networks for dynamic graphs. In *AAAI conference on Artificial Intelligence*, 2020.

QingGuo Qi, Hongyang Chen, Minhao Cheng, and Han Liu. Input snapshots fusion for scalable discrete-time dynamic graph neural networks. In *ACM SIGKDD*, 2025.

Shubhajit Roy and Anirban Dasgupta. SiST-GNN: Simultaneous spatial-temporal message passing for dynamic graph representation learning. *arXiv preprint arXiv:2605.25548*, 2026. URL https://arxiv.org/abs/2605.25548.

Kiarash Shamsi, Friedhelm Victor, Murat Kantarcioglu, Yulia Gel, and Cuneyt G Akcora. Chartalist: Labeled graph datasets for UTXO and account-based blockchains. In *Advances in Neural Information Processing Systems*, 2022.

Kiarash Shamsi, Tran Gia Bao Ngo, Razieh Shirzadkhani, Shenyang Huang, Farimah Poursafaei, Poupak Azad, Reihaneh Rabbany, Baris Coskunuzer, Guillaume Rabusseau, and Cuneyt G. Akcora. MiNT: Multi-network transfer benchmark for temporal graph learning. In *Neural Information Processing Systems*, 2025.

Geri Skenderi, Hang Li, Jiliang Tang, and Marco Cristani. Graph-level representation learning with joint-embedding predictive architectures. *Trans. Mach. Learn. Res.*, 2025.

Stephen Smale. Morse inequalities for a dynamical system. *Bulletin of the American Mathematical Society*, 66(1): 43–49, 1960. doi: 10.1090/S0002-9904-1960-10386-2.

Stephen Smale. On gradient dynamical systems. *Annals of Mathematics*, 74(1):199–206, 1961. ISSN 0003486X, 19398980. URL http://www.jstor.org/stable/1970311.

Zhe Su, Xiang Liu, Layal Bou Hamdan, Vasileios Maroulas, Jie Wu, Gunnar Carlsson, and Guo-Wei Wei. Topological data analysis and topological deep learning beyond persistent homology: a review. *Artificial intelligence review*, 59(2):58, 2025.

Guo-Wei Wei. Topological deep learning: An emerging paradigm in data science. *SIAM News*, 2024.

Danni Wu, Yuanyuan Xu, Xuemin Lin, Wenjie Zhang, and Ying Zhang. Understanding evolving graph structures for large discrete-time dynamic graph representation. *Proc. of the VLDB Endowment*, 2026a.

Haiyu Wu, Randall Balestriero, and Morgan Levine. VISReg: Variance-invariance-sketching regularization for jepa training. *arXiv preprint arXiv:2606.02572*, 2026b.

Jiaxuan You, Tianyu Du, and Jure Leskovec. ROLAND: Graph learning framework for dynamic graphs. In *ACM SIGKDD*, 2022.

Huizhe Zhang, Yuchang Zhu, Huazhen Zhong, Liang Chen, and Zibin Zheng. Scalable and efficient joint spiking embedding predictive architecture for large-scale dynamic graphs, 2026. URL https://arxiv.org/abs/2607.18412.

Xiaohui Zhang, Yanbo Wang, Xiyuan Wang, and Muhan Zhang. Efficient neural common neighbor for temporal graph link prediction, 2025. URL https://arxiv.org/abs/2406.07926.

Yifan Zhu, Fangpeng Cong, Dan Zhang, Wenwen Gong, Qika Lin, Wenzheng Feng, Yuxiao Dong, and Jie Tang. Wingnn: Dynamic graph neural networks with random gradient aggregation window. In *ACM SIGKDD*, 2023.

# A Extended Related Work

## A.1 Dynamic Graph Pretraining

Learning from dynamic graphs requires an encoder to combine relational structure with changes over time. Temporal information can enter through evolving model parameters, as in EvolveGCN (Pareja et al., 2020), or through node representations updated across snapshots and GNN layers, as in ROLAND (You et al., 2022). Pretraining determines what these encoders learn from observed interactions before downstream adaptation. This distinction between the encoder and its learning objective is central to our use of JEPA for dynamic graphs.

Temporal pretraining objectives use interaction histories to expose relationships that should be captured in the learned representations. DyG2Vec learns from temporal windows (Alomrani et al., 2024), while CPDG combines structural-temporal subgraph sampling with structural and temporal contrastive objectives (Bei et al., 2024). These approaches make the organization of observations part of the pretraining task. MoJEPA organizes an observed window into context positions and a partially masked target block. The context determines both the observations available for prediction and a reference forest that supplies communication paths across the window.

Cross-dataset transfer tests whether the prediction function learned from one graph remains useful on another. MiNT studies generalization to unseen temporal networks through multi-network pretraining (Shamsi et al., 2025).

Our experiments pretrain and fine-tune on a single source dataset, then apply the encoder and link head to another dataset with their learned parameters fixed. The target dataset supplies its own interaction history and reference context. This setting evaluates transfer across graphs within a domain without adapting model parameters on the target graph.

### A.2 Joint-Embedding Predictive Learning

Prediction in embedding space allows the learning target to be defined by an encoder. In visual JEPA models, the task is to infer representations of unobserved content from the visible context. I-JEPA studies how context coverage and target-block size shape this task for images (Assran et al., 2023). V-JEPA extends feature prediction to video (Bardes et al., 2024), and V-JEPA 2 scales video pretraining to learn representations for understanding and prediction (Assran et al., 2025). Its action-conditioned extension, V-JEPA 2-AC, uses robot interaction data during post-training to support planning. These applications show how the observations and prediction targets can be adapted to different uses of learned representations.

Learning the target representation also makes collapse prevention part of the training design. LeJEPA introduces Sketched Isotropic Gaussian Regularization (SIGReg), which uses random projections to encourage an isotropic Gaussian embedding distribution (Balestriero & LeCun, 2025). Combining this regularization with a predictive objective simplifies self-supervised training. VISReg separates control of embedding scale from distributional shape (Wu et al., 2026b). A variance term controls scale, while a sliced-Wasserstein sketching objective regularizes shape. This separation addresses vanishing regularization gradients under collapse. These methods study how the embedding distribution is constrained during learning.

Applying JEPA to graphs additionally requires defining the relational context that supports target prediction. Graph-JEPA predicts representations of masked subgraphs from a context subgraph for graph-level learning (Skenderi et al., 2025). SG-JEPA partitions context and target sets along time and uses spiking encoders for dynamic graphs (Zhang et al., 2026). MoJEPA instead defines targets at node–time positions inside an observed window. Partial interaction masking determines the evidence withheld at these positions, while a reference forest constructed from observations outside the target block supplies paths shared by both encoding branches. This design links the prediction task to the relational structure available to solve it.

Masked graph reconstruction provides a related comparison in how supervision is defined. Feature reconstruction asks the encoder to infer withheld node attributes, as in GraphMAE (Hou et al., 2022); structure reconstruction asks it to recover withheld relations, as in MaskGAE (Li et al., 2023). GraphMAE2 combines feature reconstruction with latent representation prediction (Hou et al., 2023). MoJEPA uses complete-view node embeddings as targets. The withheld interactions change the information available to the student, and their effect is learned through prediction of the teacher's representations.

MRC uses topology to construct communication paths from observed interactions. Discrete Morse theory describes a cell complex through acyclic pairings and critical cells (Forman, 2002). On graphs, discrete gradients correspond to rooted spanning forests (Chari & Joswig, 2005; Contreras & Tawfeek, 2024). MoJEPA uses this correspondence to construct a sparse reference forest guided by a node scalar field. Its edges carry current node states at different times, including snapshots in which the corresponding observed interactions are absent. The topological construction therefore determines part of the encoder's message-passing structure. Different scalar fields can guide the forest within the same framework. See Appendices. B and C for more detailed discussion on the construction and its conditional stability properties.

## B Discrete Morse Theory

Here we add additional details on the background needed regarding discrete Morse theory. See Table 3 for notation. For classical background material on computational and algebraic topology, we refer the reader to Edelsbrunner & Harer (2010) and Hatcher (2002) respectively.

For the graph statements below, let $G = (V, E)$ be a finite simple undirected graph. Betti numbers are computed with coefficients in a fixed field. We first recall the discrete vector field and gradient definitions of Forman (2002).

**Definition B.1** (Discrete vector fields and gradients)**.** *A* discrete vector field *on $G$ is a set $M$ of pairs $(v, e) \in V \times E$ with $v \in \partial e$, such that every vertex and every edge occurs in at most one pair of $M$. A $V$-*path *of $M$ is a sequence*

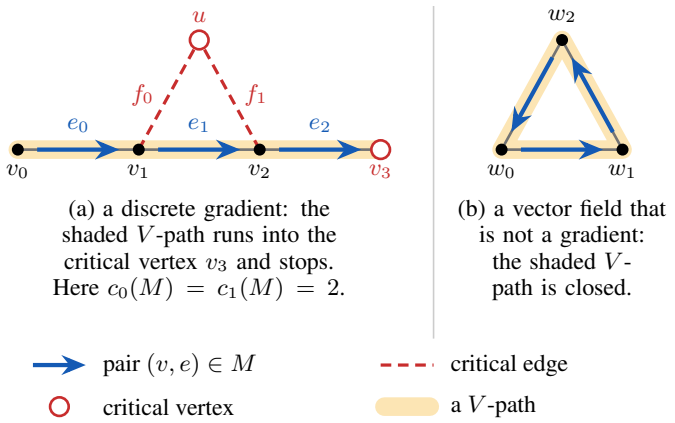
$$v_0,\, e_0,\, v_1,\, e_1,\, \ldots,\, e_{k-1},\, v_k \qquad (k \geq 0)$$

Table 3: Table of notation.

| Symbol | Meaning |
|---|---|
| *Dynamic graphs* | |
| $\mathcal{G} = (G_1, \dots, G_T)$ | Dynamic graph; $G_t = (V_t, E_t, c_t)$ a directed count snapshot |
| $\mathcal{H}_q = (G_{q-K}, \dots, G_{q-1})$ | Completed history of length $K$ for prediction at time $q$. |
| $U_t = (V_t, \mathcal{E}_t)$ | Daily simple undirected support of $G_t$ |
| $n_t = \lvert V_t \rvert,\ m_t = \lvert \mathcal{E}_t \rvert$ | Number of vertices and edges of $U_t$ |
| $c_t(u, v)$ | Directed interaction count, zero for an absent relation; symmetrized count $c_t(u,v) + c_t(v,u)$. |
| $N_{U_t}(v),\ N_{U_t}[v]$ | Open and closed undirected neighborhoods $\deg_{U_t}(v) = \lvert N_{U_t}(v) \rvert$. |
| $\beta_0(U_t),\ \beta_1(U_t)$ | Number of connected components; cycle rank $m_t - n_t + \beta_0$ |
| $\chi(U_t) = n_t - m_t$ | Euler characteristic $= \beta_0 - \beta_1$ |
| *Cell complexes and smooth Morse theory* | |
| $\Lambda,\ \Lambda^{(p)}$ | Finite regular CW complex; its set of $p$-cells |
| $\alpha \prec \beta$ | $\alpha$ is a codimension-one face of $\beta$ |
| $M,\ f : M \to \mathbb{R}$ | Smooth compact manifold; smooth function on it |
| $\mathrm{ind}(p)$ | Morse index of a critical point $p$ |
| $W^s(p),\ W^u(p)$ | Stable (descending) and unstable (ascending) manifold of $p$ |
| $(f_\lambda)_{\lambda \in [0,1]}$ | One-parameter family (path) of functions |
| *Discrete Morse theory* | |
| $\mu : \Lambda \to \mathbb{R}$ | Discrete Morse function; on graphs $\mu_t : V_t \cup \mathcal{E}_t \to \mathbb{R}$ |
| $U_\mu(\alpha),\ V_\mu(\alpha)$ | Cofaces with $\mu(\beta) \le \mu(\alpha)$; faces with $\mu(\gamma) \ge \mu(\alpha)$ |
| $\mathcal{V}$ | Discrete vector field (a matching of cells across one dimension) |
| $\mathcal{P}_t$ | The discrete gradient field on $U_t$ used by the method (a specific $\mathcal{V}$) |
| $\mathcal{V}$-path | Alternating sequence $\sigma_0 \prec \tau_0 \succ \sigma_1 \prec \cdots$ with $(\sigma_i, \tau_i) \in \mathcal{V}$ |
| $c_p(\mathcal{V})$ | Number of critical $p$-cells |
| $\mathcal{D}_\mathcal{V}(\sigma)$ | Descending set of a critical cell $\sigma$ |
| *Scores, pairings, and the skeleton* | |
| $g_t : V_t \to \mathbb{R}$ | Vertex score; $g_t = \mu_t \lvert_{V_t}$ |
| $\mu_t(e)$ | Morse edge value from matching dependent extension |
| $\nu(g)$ | Number of local minima of $g$ |
| $E_t^{\mathrm{forest}},\ E_t^{\mathrm{crit}}$ | Matched (forest) and unmatched (critical) edges; $\mathcal{E}_t = E_t^{\mathrm{forest}} \sqcup E_t^{\mathrm{crit}}$ |
| $F_t = (V_t, E_t^{\mathrm{forest}})$ | Rooted spanning forest of $U_t$ |
| $R_t \subseteq V_t$ | Roots of $F_t$ = critical vertices; $\lvert R_t \rvert = c_0$ |
| $p_t : V_t \to V_t \cup \{\perp\}$ | Pointer (parent) map; $p_t(v) = \perp$ iff $v \in R_t$ |
| $T_v,\ r(v)$ | Tree of $F_t$ containing $v$; its root |
| $\gamma_t(v)$ | $F_t$-geodesic from $v$ to $r(v)$ |
| $\mathrm{Skel}(U_t, g_t) = (R_t, p_t, E_t^{\mathrm{crit}})$ | The Morse skeleton |
| $\bar{U}_t = U_t / F_t$ | Morse quotient: contract every tree of $F_t$ to its root |
| $s(v),\ \varphi(v)$ | Strength $\mathrm{in}(v) + \mathrm{out}(v)$ and net flow $\mathrm{in}(v) - \mathrm{out}(v)$ |
| $d_r(v),\ H_r(v)$ | Hop distance and random-walk hitting time to the hub $r$ |
| $m_v^\star(g)$ | Local margin: gap between the two smallest scores on $N[v]$ |
| $m^\star(g) = \min_v m_v^\star(g)$ | Global margin |
| *Persistence* | |
| $U^{\le a}$ | Lower-star sublevel complex $\{v : g(v) \le a\} \cup \{e : \max_{v \in e} g(v) \le a\}$ |
| $\mathrm{Dgm}_p(g)$ | $p$-th persistence diagram of the lower-star filtration of $g$ |
| *Architecture* | |
| $\mathrm{Enc}_\theta,\ \mathrm{Enc}_{\bar{\theta}}$ | Context and target encoders; $\bar{\theta}$ an EMA of $\theta$ |
| $\mathrm{Pred}_\phi$ | Predictor; $z_{\mathrm{ctx}}, z_{\mathrm{tgt}}, \widehat{z}_{\mathrm{tgt}}$ its inputs and outputs |
| $\mathcal{L}_{\mathrm{JEPA}}$ | Latent prediction loss |

*with $(v_i, e_i) \in M$, $\partial e_i = \{v_i, v_{i+1}\}$ and $v_{i+1} \neq v_i$ for $0 \le i < k$. It is* closed *if $k \ge 1$ and $v_k = v_0$. The field $M$ is a* discrete gradient *if it admits no closed $V$-path. A vertex or edge occurring in no pair of $M$ is* critical*; we write $c_0(M)$ and $c_1(M)$ for the numbers of critical vertices and edges.*

**Definition B.2** (State, Rooted Spanning Forest)**.** *A state or rooted spanning forest of $G$ is a pair $(F, R)$ with $F \subseteq E$ acyclic and $R \subseteq V$ containing exactly one vertex from each tree of $F$. The elements of $R$ are then called the roots. For a vertex $v \in V$, we write that $T_v$ is the tree of $F$ in which $v$ lives, and $r(v) \in R$ is its root. For $u, v \in T_v$, the $F-$geodesic from $u$ to $v$ is the unique simple path in $(V, F)$ from $u$ to $v$.*




(a) a discrete gradient: the shaded $V$-path runs into the critical vertex $v_3$ and stops. Here $c_0(M) = c_1(M) = 2$.

(b) a vector field that is not a gradient: the shaded $V$-path is closed.

Figure 4: Discrete vector fields on a graph $G$. An arrow drawn from $v$ along $e$ records the pair $(v, e) \in M$, so a $V$-path is a walk that follows the arrows; the matching condition says no vertex and no edge carries two arrows. In (a) no such walk returns to its start, so $M$ is a discrete gradient, and the cells $u, v_3, f_0, f_1$ are critical. In (b) every cell is matched and the three arrows close up, which is exactly what a gradient forbids.

Furthermore, it is known that there is an equivalence between discrete gradient vector fields on a graph and a rooted spanning forest on it (Chari & Joswig, 2005) Contreras & Tawfeek (2024).

Consider $\mathcal{S}(G)$ for the set of all rooted spanning forests $(F, R)$ of $G$, including isolated vertices as trees. For each $v \in V \setminus R$, let $e_v$ be the first edge of the unique path in $F$ from $v$ to its root. Define

$$M(F, R) = \{(v, e_v) : v \in V \setminus R\}.$$

**Theorem B.3** (Structure theorem (Chari & Joswig (2005))). *The assignment*

$$(F, R) \longmapsto M(F, R)$$

*defines a bijection from $\mathcal{S}(G)$ onto the set of discrete gradients on $G$. More precisely, for every state $(F, R)$, the matched vertices of $M(F, R)$ are exactly $V \setminus R$ and the matched edges are exactly $F$. Hence the critical vertices are exactly $R$ and the critical edges are exactly $E \setminus F$.*

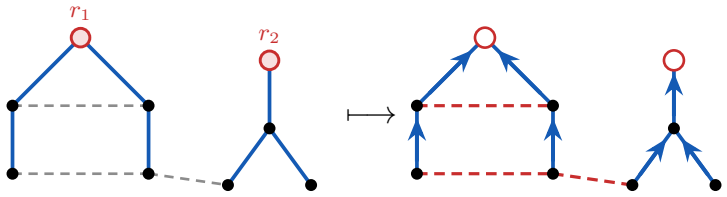
*Consequently, every discrete gradient $M$ satisfies*

$$c_0(M) - c_1(M) = |V| - |E| = \chi(G).$$

*Moreover, if $M = M(F, R)$ is the corresponding state, then*

$$c_0(M) \geq \beta_0(G),$$

*with equality if and only if the restriction of $F$ to each connected component of $G$ is a spanning tree of that component.*




$(F, R)$ is a spanning forest with roots R

$M(F, R)$, a discrete gradient

Figure 5: The map of Theorem B.3. Each non-root vertex is matched with the first edge of its $F$-geodesic to its root, so the arrows all point rootwards; the roots and the edges outside $F$ receive no arrow. The inverse recovers $F$ as the set of matched edges and $R$ as the set of unmatched vertices, which is why the correspondence is a bijection.

**Definition B.4** (Discrete Morse function)**.** *Let $\Lambda$ be a finite regular CW complex, and use $\Lambda$ also to denote its set of cells. Write $\Lambda^{(p)}$ for the set of $p$-cells and $\alpha \prec \beta$ when $\alpha$ is a codimension-one face of $\beta$. Let $\mu : \Lambda \to \mathbb{R}$. For a $p$-cell $\alpha$, set*

$$U_\mu(\alpha) = \{\beta \in \Lambda^{(p+1)} : \alpha \prec \beta,\ \mu(\beta) \le \mu(\alpha)\}, \tag{6}$$

$$V_\mu(\alpha) = \{\gamma \in \Lambda^{(p-1)} : \gamma \prec \alpha,\ \mu(\gamma) \ge \mu(\alpha)\}. \tag{7}$$

*$\mu$ is a* discrete Morse function *if $|U_\mu(\alpha)| \le 1$ and $|V_\mu(\alpha)| \le 1$ for every cell $\alpha$. A cell is* critical *if $|U_\mu(\alpha)| = |V_\mu(\alpha)| = 0$, that is, every coface has strictly larger value and every face strictly smaller.*

$U_\mu(\alpha)$ collects the cofaces that fail to be larger, while similarly $V_\mu(\alpha)$ collects the faces that fail to be smaller; the Morse condition says that a cell can have at most one exception in each direction. Forman shows that a cell cannot have an exception in both directions simultaneously, so each cell is either critical or belongs to exactly one exceptional pair $(\alpha, \beta)$ with $\alpha \prec \beta$ and $\mu(\beta) \le \mu(\alpha)$.

### B.1 Positioning of Morse Theory

Persistent homology is a central tool of topological data analysis, where given a real-valued function on a space, or a filtered complex, the persistence diagram $\mathrm{Dgm}_*$ records the scales at which homological features are born and die, and is stable under $||\cdot||_\infty$ perturbation of the function (Cohen-Steiner et al., 2007). Within dynamic graphs, a similar idea appears through the notion of *zigzag* persistence (Carlsson et al., 2009), which tracks homology classes across a sequence of snapshots joint via their unions or intersections.

Morse theory distinguishes itself from persistence. Namely, Morse theory describes a CW complex by enforcing how a gradient flow of a Morse function imposed onto it partitions the space into critical and noncritical cells, and how flow lines along the gradient connect these critical cells together (Milnor et al., 1969; Smale, 1961). In particular, the Morse complex describes where the topology sits and how it is connected aside from just the homology features that exist. In other words, instead of descriptions of homology, discrete Morse theory describes the homotopy type of sublevel types (or how the homology of the space interacts) (Bauer, 2011). Catanzaro et al. (2020) construct pairs of Morse functions on the sphere with identical persistence barcodes and Reeb graphs are geometrically equivalent but not graph equivalent, and study the space of Morse functions that share a barcode. A similar phenomenon is known to exist on graphs. In MoJEPA, a discrete gradient selects a reference forest that supplies communication routes to the encoder, while the prediction targets are teacher-produced node embeddings.

### B.2 As One-dimensional Cell Complexes

A discrete-time dynamic graph is a sequence $\mathcal{G} = (G_1, \dots, G_T)$ of directed weighted snapshots where $G_t = (V_t, E_t, c_t)$ and $c_t(u, v)$ is its directed interaction count, equal to zero for absent relations. Its daily undirected support is

$$U_t = (V_t, \mathcal{E}_t), \qquad \mathcal{E}_t = \big\{\{u, v\} : u, v \in V_t,\ u \ne v,\ c_t(u, v) + c_t(v, u) > 0\big\}.$$

This support has one edge for each observed non-self endpoint pair; it discards direction and multiplicity but retains the vertex set $V_t$; in particular, a vertex observed only through self-interactions is isolated in $U_t$. The directed records and their counts remain available to the encoder.

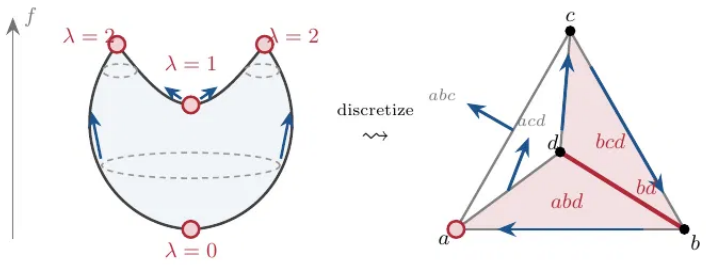
We regard $U_t$ as a finite simplicial complex, hence a finite regular CW complex, of dimension at most one. Vertices are 0-cells and edges are 1-cells; $v \prec e$ means $v \in e$. Writing $n_t = |V_t|$ and $m_t = |\mathcal{E}_t|$, we have

$$\beta_0(U_t) = \text{number of connected components}, \qquad \beta_1(U_t) = m_t - n_t + \beta_0(U_t), \qquad \beta_k(U_t) = 0 \quad (k \ge 2).$$

The reference forest in Section 4.2 is constructed on the aggregated observed-context support $U_{\mathcal{C}}$, rather than separately on each daily support $U_t$. Both are simple graphs, so the graph constructions and results below apply to either when the underlying support is specified.

### B.3 Smooth Morse Theory on Manifolds

Let $M$ be a smooth, compact, $d$-dimensional manifold without boundary, equipped with a Riemannian metric, so that gradients and gradient flows are well-defined. The definitions below apply to smooth functions $f : M \to \mathbb{R}$ and in Sections B.4, assume that $f$ is Morse.




$f : S^2 \to \mathbb{R}$ Morse, $\nabla f$ in blue  $V$ discrete gradient on $\partial\Delta^3$

1 min, 1 saddle, 2 maxima $\longleftrightarrow$ $c_0 = 1,\ c_1 = 1,\ c_2 = 2$

**Definition B.5** (Critical Point)**.** *Let $f\colon M \to \mathbb{R}$ be smooth. A point $p \in M$ is a* critical point *of $f$ if $df_p = 0$. Equivalently, all first partial derivatives of $f$ vanish at $p$ in any local coordinate chart. Otherwise, $p$ is called a* regular point*.*

**Definition B.6** (Morse Function and Index)**.** *A smooth function $f\colon M \to \mathbb{R}$ is a* Morse function *if every critical point $p$ is non-degenerate; that is, if the Hessian denoted via $\mathrm{Hess}_p f$ is nonsingular.*

*We further define the* index *of a non-degenerate critical point $p$, denoted $\mathrm{ind}(p)$, to be the number of negative eigenvalues of $\mathrm{Hess}_p f$.*

Non-degeneracy implies that critical points are isolated and since $M$ is compact, a Morse function has only finitely many critical points. The index records the number of independent directions in which $f$ decreases near a critical point. In particular, $\mathrm{ind}(p) = 0$ for a local minimum, $\mathrm{ind}(p) = d$ for a local maximum, and intermediate values correspond to saddle-type critical points.

### B.4 Gradient Flow and Stable and Unstable Manifolds

Because $M$ is compact and has no boundary, the smooth vector field $-\nabla f$ is complete. It generates a flow $\varphi_\tau : M \to M$, defined for every $\tau \in \mathbb{R}$, satisfying

$$\frac{d}{d\tau}\varphi_\tau(x) = -\nabla f\big(\varphi_\tau(x)\big).$$

Along such a flow line,

$$\frac{d}{d\tau} f\big(\varphi_\tau(x)\big) = -\left\|\nabla f\big(\varphi_\tau(x)\big)\right\|^2 \le 0,$$

with equality precisely at critical points. Thus, $f$ strictly decreases along every nonconstant negative-gradient trajectory. For a critical point $p$, define the stable and unstable manifolds by

$$W^s(p) = \left\{ x \in M : \lim_{\tau \to +\infty} \varphi_\tau(x) = p \right\},$$

and

$$W^u(p) = \left\{ x \in M : \lim_{\tau \to -\infty} \varphi_\tau(x) = p \right\}.$$

The stable manifold $W^s(p)$ is a smooth submanifold of dimension $\dim W^s(p) = d - \mathrm{ind}(p)$, while the unstable manifold $W^u(p)$ has dimension $\dim W^u(p) = \mathrm{ind}(p)$.

Every nonconstant trajectory converges to critical points as $\tau \to -\infty$ and $\tau \to +\infty$. When stable and unstable manifolds intersect transversely for every pair of critical points, the function and the metric together are said to satisfy the *Morse–Smale condition*. For distinct $p$ and $q$, the intersection $W^u(p) \cap W^s(q)$ consists of points on trajectories from $p$ to $q$; when nonempty, it is a smooth manifold of dimension $\mathrm{ind}(p) - \mathrm{ind}(q)$. The resulting intersection structure forms the Morse-Smale complex (Smale, 1960).

On a graph, each vertex has a unique maximal gradient path ending at a critical vertex, and for a critical edge $e = \{u, v\}$, its endpoints therefore determine two paths toward the roots, which may share vertices and edges.

Then, $e$ and these paths describe how the critical edge connects to the roots when the forest is contracted. MoJEPA uses the paired forest edges as reference communication routes, where its prediction targets are teacher-produced node embeddings.

### B.5 The Fundamental Theorem of Morse Theory

Morse theory states that $M$ has the homotopy type of a CW complex containing one cell of dimension $\mathrm{ind}(p)$ for each critical point $p$ of $f$ (Milnor et al., 1969). Equivalently, as one increases the height threshold $a$ in the sublevel sets

$$M_a = \{x \in M : f(x) \leq a\},$$

their homotopy type can change only when $a$ passes through a critical value of $f$. If a single critical point of index $\lambda$ occurs at that critical value, then the topology changes by the attachment of a $\lambda$-dimensional cell. An immediate consequence of the critical point change is the family of *Morse inequalities*. If $c_k$ denotes the number of critical points of index $k$, then

$$c_k \geq \beta_k(M),$$

where $\beta_k(M)$ is the $k$th Betti number of $M$. Moreover,

$$\sum_{k=0}^{d}(-1)^k c_k = \sum_{k=0}^{d}(-1)^k \beta_k(M) = \chi(M),$$

where $\chi(M)$ denotes the Euler characteristic.

From the sublevel-set perspective, index-0 critical points create new connected components. Index-1 critical points attach one-dimensional handles, which may either merge previously distinct connected components or create a new one-dimensional cycle. For one-dimensional complexes such as graphs, only dimensions 0 and 1 occur and is the reason that discrete Morse theory on graphs admits a particularly concrete combinatorial description.

### B.6 Morse Inequalities and Collapse

**Theorem B.7** (Forman)**.** *Let $\mathcal{V}$ be a discrete gradient on $\Lambda$. Then for every $p$, $c_p(\mathcal{V}) \geq \beta_p(\Lambda)$, and $\sum_p(-1)^p c_p(\mathcal{V}) = \chi(\Lambda)$. Moreover $\Lambda$ is homotopy equivalent to a CW complex with exactly $c_p(\mathcal{V})$ cells of dimension $p$.*

The last statement is the discrete analog of the fundamental theorem of Section B.5: the critical cells of any gradient field form a CW model for $\Lambda$, and a field with $c_p = \beta_p$ for all $p$ (a *perfect* field) gives a minimal one. On a graph the homotopy equivalence is realized concretely by contracting the matched edges. Furthermore, the dimension function $\mu(\sigma) = \dim \sigma$ satisfies Definition B.4 with $U_\mu(\alpha) = V_\mu(\alpha) = \varnothing$ for every cell, so every cell is critical and nothing is paired. This is a valid discrete Morse function whose skeleton is the whole complex; it is noted in Kannan et al. (2019) as the case a construction must avoid.

## C Stability

### C.1 Minimum Neighbor Construction

To discuss stability, we must first acknowledge that we use a particular gradient field in our skeleton construction, and we therefore state its selection rule.

Let $U = (V, \mathcal{E})$ be a finite simple graph and $g : V \to \mathbb{R}$ be a vertex score such that the minimum of $g$ on every closed neighborhood $N[v]$ is unique. Here, $g$ denotes the score used for comparison by the construction, including any deterministic tie-breaking as specified in following sections. Note that all score perturbations and local margins below refer to these comparison scores. Now, for each $v \in V$, write a $a_g(v) = \arg\min_{u \in N[v]} g(u)$. Define

$$p_g(v) = \begin{cases} \perp, & a_g(v) = v, \\ a_g(v), & a_g(v) \neq v. \end{cases}$$

Equivalently,

$$p_g(v) = \begin{cases} \perp, & g(v) = \min_{u \in N[v]} g(u), \\ \arg\min\limits_{u \in N(v)} g(u), & \text{otherwise.} \end{cases}$$

Let $\mathcal{M}_g = \{(v, \{v, p_g(v)\}) : p_g(v) \neq \perp\}$, then $\mathcal{M}_g$ is an acyclic partial matching between vertices and edges, and therefore defines a discrete gradient vector field on $U$. Furthermore, $\mathcal{M}_g$ is induced by a discrete Morse function $\mu_g : V \cup \mathcal{E} \to \mathbb{R}$ extending $g$. In particular, one may take $\mu_g(v) = g(v)$ for $v \in V$, and for an edge $e = \{u, v\}$,

$$\mu_g(e) = \begin{cases} \dfrac{g(u) + g(v)}{2}, & e = \{x, p_g(x)\} \text{ for some } x \in V, \\ \max\{g(u), g(v)\} + \varepsilon_e, & \text{otherwise}, \end{cases}$$

for $\varepsilon_e > 0$.

In other words, we can show that the critical vertices are the local minima of $g$ with respect to closed neighborhoods, and the forest edges are

$$\mathcal{E}_{\text{for}}(p) = \big\{\{v, p(v)\} : v \in V,\ p(v) \neq \perp\big\},$$

If $p_g(v) = u \neq \perp$, then $g(u) < g(v)$, so every pointer strictly decreases the score. Each vertex belongs to at most one pair by definition of $p_g$ and if an edge $\{u, v\}$ belonged to two pairs, then $p_g(u) = v$ and $p_g(v) = u$, implying both $g(v) < g(u)$ and $g(u) < g(v)$, which is a contradiction. So, $M_g$ is a partial matching. A nontrivial closed $V$-path would give a strictly decreasing sequence of scores returning to its starting value, which is not possible, and so $M_g$ is a discrete gradient.

The proposed extension $\mu_g$ induces exactly this matching, in that for every selected edge $e = \{v, p_g(v)\}$, $g(p_g(v)) < \mu_g(e) < g(v)$. Every unselected edge has value strictly greater than both endpoint scores, and any other selected edge incident to $v$ is selected by a neighbor $w$ with $p_g(w) = v$, and so has value $(g(w) + g(v))/2 > g(v)$. Subsequently, each vertex has at most one incident edge whose value is at most its own score, and each edge has at most one endpoint whose score is at least its value. Finally, $\mu_g$ satisfies the conditions to be discrete Morse and its exceptional pairs are precisely $M_g$.

### C.2 A Distance Between Morse Skeletons

We describe a Morse skeleton by its vertex pointers and its critical edge set; the pointers encode the gradient assignment of each vertex, while the critical edges are those not selected for the gradient forest. The distance below compares the Morse skeleton combinatorial object across snapshots.

Fix a finite vertex time-frame $\mathcal{V} = \bigcup_{t=1}^{T} V_t$. For a snapshot $U = (V, \mathcal{E})$ with $V \subseteq \mathcal{V}$, let $p : V \to V \cup \{\perp\}$ denote its gradient pointer map, where $p(v) = \perp$ if and only if $v$ is a root. The corresponding forest edge set is

$$\mathcal{E}_{\text{for}}(p) = \big\{\{v, p(v)\} : v \in V,\ p(v) \neq \perp\big\},$$

and the critical-edge set is

$$\mathcal{E}_{\text{crit}} = \mathcal{E} \setminus \mathcal{E}_{\text{for}}(p).$$

To compare snapshots with different active vertex sets, introduce a symbol † indicating that a vertex is absent and extend $p$ to $\widetilde{p} : \mathcal{V} \longrightarrow \mathcal{V} \cup \{\perp, \dagger\}$ by

$$\widetilde{p}(v) = \begin{cases} p(v), & v \in V, \\ \dagger, & v \notin V. \end{cases}$$

**Definition C.1** (Morse-skeleton distance)**.** *Let*

$$\mathcal{S} = (\widetilde{p}, \mathcal{E}_{\text{crit}}) \qquad \textit{and} \qquad \mathcal{S}' = (\widetilde{p}', \mathcal{E}'_{\text{crit}})$$

*be two Morse skeletons on the common vertex time-frame $\mathcal{V}$. Define*

$$d_{\text{Sk}}(\mathcal{S}, \mathcal{S}') = \big|\{v \in \mathcal{V} : \widetilde{p}(v) \neq \widetilde{p}'(v)\}\big| + \big|\mathcal{E}_{\text{crit}} \triangle \mathcal{E}'_{\text{crit}}\big|,$$

*where $\triangle$ denotes symmetric difference.*

For convenience, we write

$$d_p(S, S') = \big|\{v \in \mathcal{V} : \widetilde{p}(v) \neq \widetilde{p}'(v)\}\big|, \qquad d_c(S, S') = |E_{\text{crit}} \triangle E'_{\text{crit}}|,$$

so that $d_{\text{Sk}} = d_p + d_c$.

The first term counts changes in vertex pointers, including changes in root status and the arrival or departure of vertices; the second term counts the changes in critical edge membership. Therefore, $d_{\mathrm{Sk}}$ measures the change in the combinatorial Morse skeleton and is not a distance between the teacher embeddings used as prediction targets.

Let $\mathfrak{S}(\mathcal{V})$ denote the set of Morse skeletons on the finite vertex time-frame $\mathcal{V}$. By definition, an element $\mathcal{S} = (\widetilde{p}, \mathcal{E}_{\mathrm{crit}}) \in \mathfrak{S}(\mathcal{V})$ consists of an extended pointer map $\widetilde{p} : \mathcal{V} \to \mathcal{V} \cup \{\perp, \dagger\}$ and a critical-edge set $\mathcal{E}_{\mathrm{crit}} \subseteq \binom{\mathcal{V}}{2}$, subject to the requirement that they arise from a valid discrete gradient on some undirected support graph $U = (V, \mathcal{E})$ with $V \subseteq \mathcal{V}$. In particular,

$$V = \{v \in \mathcal{V} : \widetilde{p}(v) \neq \dagger\},$$

and whenever $\widetilde{p}(v) \in \mathcal{V}$, one has $\widetilde{p}(v) \in V$ and $\{v, \widetilde{p}(v)\} \in \mathcal{E}$.

**Metric property.** The distance $d_{\mathrm{Sk}}$ is a metric on $\mathcal{S}(\mathcal{V})$. Indeed, its two terms are Hamming distances on the extended pointer maps and the critical-edge indicator vectors, respectively. Both are nonnegative, symmetric, and satisfy the triangle inequality. Moreover, $d_{\mathrm{Sk}}(S, S') = 0$ holds exactly when both coordinates agree, which is precisely $S = S'$.

### C.3 Static Stability Under Score Perturbations

We first consider a simpler case, in which we fix an undirected support graph $U = (V, \mathcal{E})$ and ask how the Morse skeleton changes when only the vertex score is perturbed. For the skeleton construction we use, the pointer at a vertex $v$ is determined by the minimum score on its closed neighborhood by the scoring function

$$p_g(v) = \begin{cases} \perp, & g(v) = \min_{u \in N[v]} g(u), \\ \arg\min\limits_{u \in N(v)} g(u), & \text{otherwise.} \end{cases} \tag{8}$$

Whenever the minimum on $N[v]$ is unique, this is equivalent to the rule that $p_g(v)$ records the unique minimizer of $g$ on $N[v]$, with $\perp$ used when that minimizer is $v$ itself.

**Definition C.2** (Local score margin). *Let $v \in V$. If $|N[v]| \geq 2$, let $g_{(1)}(v) \leq g_{(2)}(v)$ denote the two smallest values of $g$ on $N[v]$ with multiplicity, and define $m_v(g) = g_{(2)}(v) - g_{(1)}(v)$. If $N[v] = \{v\}$, we set $m_v(g) = +\infty$. The global score margin can be viewed as*

$$m_\star(g) = \min_{v \in V} m_v(g).$$

*Therefore, we see that $m_v(g) > 0$ if and only if the minimum of $g$ on $N[v]$ is unique. For $V = \varnothing$, we use the conventions $m_\star(g) = +\infty$ and $\|\delta\|_\infty = 0$.*

**Theorem C.3** (Local-margin stability). *Let $U = (V, \mathcal{E})$ be fixed, let $g : V \to \mathbb{R}$, and let $\delta : V \to \mathbb{R}$ be a score perturbation. For any vertex $v$ with $m_v(g) > 0$, if*

$$\max_{u \in N[v]} |\delta(u)| < \frac{m_v(g)}{2}, \tag{9}$$

*then $p_{g+\delta}(v) = p_g(v)$. Consequently, if $m_\star(g) > 0$ and $\|\delta\|_\infty < \frac{m_\star(g)}{2}$, then the entire Morse skeleton is unchanged:*

$$\mathrm{Skel}(U, g + \delta) = \mathrm{Skel}(U, g).$$

*Equivalently,*

$$d_{\mathrm{Sk}}\big(\mathrm{Skel}(U, g + \delta), \mathrm{Skel}(U, g)\big) = 0.$$

*Proof.* Fix $v \in V$ with $m_v(g) > 0$, and let $a = \arg\min_{u \in N[v]} g(u)$. Since $m_v(g) > 0$, this minimizer is unique. Now, for every $b \in N[v] \setminus \{a\}$, the definition of the local margin gives us $g(b) - g(a) \geq m_v(g)$. After perturbing the score, we have

$$\begin{aligned} (g + \delta)(b) - (g + \delta)(a) &= g(b) - g(a) + \delta(b) - \delta(a) && (10) \\ &\geq m_v(g) - |\delta(b)| - |\delta(a)|. && (11) \end{aligned}$$

Under equation 9,

$$|\delta(b)| < \frac{m_v(g)}{2}, \qquad |\delta(a)| < \frac{m_v(g)}{2},$$

and therefore $(g+\delta)(b) - (g+\delta)(a) > 0$. Therefore

$$(g+\delta)(a) < (g+\delta)(b) \qquad \text{for every } b \in N[v] \setminus \{a\}.$$

So, we see that $a$ remains the unique minimizer of the perturbed score on $N[v]$. What follows is that there are now two cases, the first where $a = v$ and the second where $a \neq v$.

If $a = v$, then $v$ is the unique minimum of its closed neighborhood both before and after the perturbation. By equation 8, we find that $p_g(v) = \perp = p_{g+\delta}(v)$.

On the other hand, if $a \neq v$, then $a \in N(v)$ and is the unique minimum of $N[v]$ both before and after the perturbation. Therefore, we see that $p_g(v) = a = p_{g+\delta}(v)$.

In either case, we see that $p_{g+\delta}(v) = p_g(v)$. If $\|\delta\|_\infty < m_\star(g)/2$, the local condition holds at every vertex, so all pointers remain unchanged. Since the graph is fixed, the forest and critical-edge sets also remain unchanged. And therefore, $\mathrm{Skel}(U, g+\delta) = \mathrm{Skel}(U, g)$. Note that for an empty graph, this conclusion holds trivially.

□

### C.4 Stability Under Graph Edits

We next hold the vertex score fixed and allow the underlying support graph to change. Let $\mathcal{V}$ be the finite vertex time-frame introduced in Section C.2, and let $g : \mathcal{V} \to \mathbb{R}$ be injective. Here $g$ denotes injective comparison scores representing the scalar-field ordering, with exact ties resolved by fixed vertex identifiers and all strict scalar-field inequalities preserved. These comparison scores are held fixed throughout the graph-edit comparison. Now for a $V \subseteq \mathcal{V}$ and a support graph $U = (V, \mathcal{E})$, we write $\mathcal{S}(U, g) = \mathrm{Skel}(U, g)$. Injectivity of $g$ ensures that the minimum on every closed neighborhood is unique for every support graph considered below, so pointer selection applies throughout.

**Lemma C.4** (Single-edge stability)**.** *Let $U = (V, \mathcal{E})$ and $U' = (V, \mathcal{E}')$ have the same vertex set and suppose they differ such that $|E \triangle \mathcal{E}'| = 1$. Then*

$$d_{\mathrm{Sk}}\big(\mathcal{S}(U, g), \mathcal{S}(U', g)\big) \leq 2.$$

*Proof.* Let $e = \{u, w\}$ be the unique edited edge. Since all closed neighborhoods other than $N[u]$ and $N[w]$ are unchanged, by the minimum-neighbor rule in Section C, only the pointers at $u$ and $w$ can change.

We first consider the case of an edge insertion or equivalently where the edge set becomes

$$\mathcal{E}' = \mathcal{E} \cup \{e\}.$$

Without loss of generality, suppose $g(u) < g(w)$. Since $u \in N_{U'}[u]$ and $g(u) < g(w)$, the newly added vertex $w$ cannot become the minimum of $N_{U'}[u]$, and therefore $p_{U',g}(u) = p_{U,g}(u)$ and only the pointer at $w$ can change.

There are now three cases we must examine.

*(i): $p_{U',g}(w) = p_{U,g}(w)$.* If $p_{U',g}(w) = p_{U,g}(w)$ neither endpoint selects the new edge $e$, so $e$ is critical in $U'$ and no other forest edge changes. Therefore, we see that $d_p = 0$ and $d_c = 1$, and so $d_{\mathrm{Sk}} = 1$.

*(ii): $w$ was a root and now points to $u$.* Then $p_{U,g}(w) = \perp, p_{U',g}(w) = u$ and the new edge $e$ becomes a forest edge, and no previous forest edge is removed. Hence the critical-edge set is unchanged, $d_p = 1, d_c = 0$, and $d_{\mathrm{Sk}} = 1$.

*(iii): $w$ changes parent from some $x$ to $u$.* Then $p_{U,g}(w) = x,$ and $p_{U',g}(w) = u$. Let f={w,x} such that, before the insertion, $f$ is a forest edge. After the insertion, the new edge $e$ replaces $f$ as the forest edge selected by $w$. Since $f$ remains present in the graph but is no longer selected by either endpoint, it becomes critical. So, $d_p = 1,$ and $d_c = 1,$ so $d_{\mathrm{Sk}} = 2$. Therefore, we see that every single-edge insertion changes the skeleton by at most 2.

An edge deletion is the reverse of an edge insertion, and applying the insertion bound to the reversed pair of graphs and using symmetry gives

$$d_{\mathrm{Sk}}\big(S(U, g), S(U', g)\big) = d_{\mathrm{Sk}}\big(S(U', g), S(U, g)\big) \leq 2.$$

So, we see the bound holds for both insertions and deletions. □

**Theorem C.5** (Stability under graph edits)**.** *Let $U = (V, \mathcal{E})$ and $U' = (V', \mathcal{E}')$ be support graphs with $V, V' \subseteq \mathcal{V}$, and let $g : \mathcal{V} \to \mathbb{R}$ be fixed and injective. Then*

$$d_{\mathrm{Sk}}\big(\mathcal{S}(U, g), \mathcal{S}(U', g)\big) \leq |V \triangle V'| + 2|\mathcal{E} \triangle \mathcal{E}'|.$$

*Proof.* First delete the edges in $E \setminus E'$; note that each deletion changes the skeleton distance by at most 2, by the single-edge stability lemma. Further, every vertex in $V \setminus V'$ is now isolated. If we now delete these vertices, then insert the vertices in $V' \setminus V$ as isolated vertices, each such operation changes exactly one extended pointer between the root and absent symbols, contributing 1 to the distance.

Finally, insert the edges in $E' \setminus E$, each contributing at most 2. The triangle inequality therefore gives

$$d_{\mathrm{Sk}}\big(S(U, g), S(U', g)\big) \leq |V \triangle V'| + 2|E \triangle E'|.$$

□

### C.5 Combined Stability Under Graph and Score Changes

We see that there are two sources of variation in the Morse construction, namely the changes in the node score on a fixed graph and changes in the graph under a fixed score. We now address a combination of these two. Let $U = (V, \mathcal{E})$ be a support graph and let $g, g' : V \to \mathbb{R}$ be two scoring functions. For $v \in V$, let $m_v^U(g)$ denote the local margin of $g$ on the closed neighborhood of $v$, $N_U[v]$, as in definition C.2. Now, let us define the set of vertices whose local stability attributes are exhausted by the score change $g \to g'$ as

$$\mathcal{H}_U(g, g') = \left\{ v \in V : \max_{u \in N_U[v]} |g'(u) - g(u)| \geq \frac{m_v^U(g)}{2} \right\}$$

(note that vertices with zero local margin are included in $\mathcal{H}_U(g, g')$ automatically).

**Lemma C.6** (Score-induced skeleton churn)**.** *Let $U = (V, \mathcal{E})$ be fixed. Suppose the skeleton construction is well-defined for both $g$ and $g'$. Then*

$$d_{\mathrm{Sk}}\big(\mathcal{S}(U, g), \mathcal{S}(U, g')\big) \leq 3\,|\mathcal{H}_U(g, g')|.$$

*Proof.* Let $\mathcal{C} = \{v \in V : p_g(v) \neq p_{g'}(v)\}$ be the set of vertices whose pointers actually change. By Theorem C.3, if $v \notin \mathcal{H}_U(g, g')$, then $p_g(v) = p_{g'}(v)$. Therefore $\mathcal{C} \subseteq \mathcal{H}_U(g, g')$, and subsequently $d_p = |\mathcal{C}| \leq |\mathcal{H}_U(g, g')|$.

Because the underlying edge set $\mathcal{E}$ is fixed, $\mathcal{E}_{\mathrm{crit}}(g) = \mathcal{E} \setminus \mathcal{E}_{\mathrm{for}}(g)$ and similarly for $g'$. Consequently,

$$\mathcal{E}_{\mathrm{crit}}(g) \triangle \mathcal{E}_{\mathrm{crit}}(g') = \mathcal{E}_{\mathrm{for}}(g) \triangle \mathcal{E}_{\mathrm{for}}(g').$$

A vertex whose pointer does not change contributes the same selected forest edge under both scores, and a vertex whose pointer does change can remove at most one old forest edge and introduce at most one new forest edge. Therefore

$$|\mathcal{E}_{\mathrm{for}}(g) \triangle \mathcal{E}_{\mathrm{for}}(g')| \leq 2|\mathcal{C}| \leq 2|\mathcal{H}_U(g, g')|.$$

Implying that

$$d_c \leq 2|\mathcal{H}_U(g, g')|.$$

Adding the pointer and critical-edge terms gives

$$d_{\mathrm{Sk}}\big(\mathcal{S}(U, g), \mathcal{S}(U, g')\big) = d_p + d_c \leq 3|\mathcal{H}_U(g, g')|.$$

□

We now prove Theorem 4.2 from the main text.

*Proof.* Insert the intermediate skeleton $\mathcal{S}(U, g')$. By the triangle inequality,

$$\begin{aligned} d_{\mathrm{Sk}}\big(\mathcal{S}(U, g), \mathcal{S}(U', g')\big) \leq\ & d_{\mathrm{Sk}}\big(\mathcal{S}(U, g), \mathcal{S}(U, g')\big) \\ & + d_{\mathrm{Sk}}\big(\mathcal{S}(U, g'), \mathcal{S}(U', g')\big). \end{aligned} \tag{12}$$

Lemma C.6 bounds the first term by $3|\mathcal{H}_U(g, g')|$, while Theorem C.5 bounds the second by

$$|V \triangle V'| + 2|\mathcal{E} \triangle \mathcal{E}'|.$$

Hence

$$d_{\mathrm{Sk}}\big(\mathcal{S}(U, g), \mathcal{S}(U', g')\big) \leq 3|\mathcal{H}_U(g, g')| + |V \triangle V'| + 2|\mathcal{E} \triangle \mathcal{E}'|.$$

Alternatively, insert the intermediate skeleton $\mathcal{S}(U', g)$:

$$\mathcal{S}(U, g) \longrightarrow \mathcal{S}(U', g) \longrightarrow \mathcal{S}(U', g').$$

Applying Theorem C.5 first and Lemma C.6 second gives

$$d_{\mathrm{Sk}}\big(\mathcal{S}(U, g), \mathcal{S}(U', g')\big) \leq |V \triangle V'| + 2|\mathcal{E} \triangle \mathcal{E}'| + 3|\mathcal{H}_{U'}(g, g')|.$$

Both bounds hold, so taking the smaller of the two score-churn terms yields the stated result. □

Theorem C.5 controls the graph edits while Theorem 4.2 accounts for score changes through the sets $\mathcal{H}_U(g, g')$ and $\mathcal{H}_{U'}(g, g')$. When scores are recomputed from observations, applying these bounds requires evaluating or bounding those score-change terms. Specifically, changes in the selected hub or hitting-time field can affect scores throughout a connected component. The results therefore give conditional bounds on changes in local pointer and critical-edge membership; they do not directly bound changes in complete paths towards roots or in learned embeddings.

We now prove Proposition 4.1 from the main text.
Fix a non-singleton connected component $U = (V, E)$, with symmetric weight matrix $W$ and chosen hub $h$. Let $H(v) = \mathbb{E}_v[\tau_h]$. Then $H(h) = 0$, and for $v \neq h$,

$$H(v) = 1 + \sum_{u \in V} P(v, u)H(u), \qquad P(v, u) = \frac{W(v, u)}{\sum_{w \in V} W(v, w)}. \tag{13}$$

At $h$, set $P(h, h) = 1$ and $P(h, u) = 0$ for $u \neq h$, where we let $g$ be distinct comparison scores representing the lexicographic order $(H(v), \mathrm{id}(v))$.

*Proof.* Because $U$ is finite and connected, every vertex has a path to $h$ consisting of positive-weight edges. Hence the absorbing random walk reaches $h$ almost surely and has finite expected hitting time from every vertex. In particular,

$$0 \leq H(v) < \infty \qquad \text{for all } v \in V.$$

Fix $v \neq h$. Rearranging our hitting time equation gives

$$\sum_{u \in V} P(v, u)H(u) = H(v) - 1 < H(v). \tag{14}$$

Suppose, toward a contradiction, that every support neighbor of $v$ satisfied $H(u) \geq H(v)$. Any self-loop contribution has value $H(v)$ itself. Since $P(v, \cdot)$ is a probability distribution supported on $v$ and its positive-weight neighbors, this would imply

$$\sum_{u \in V} P(v, u)H(u) \geq H(v),$$

contradicting equation 14. Therefore there exists at least one support neighbor $u \in N_U(v)$ such that $H(u) < H(v)$.

For every $v \neq h$, the strict raw inequality just proved survives in the comparison scores, so $v$ has a neighbor of smaller $g$-value and cannot be a root. Moreover, $H(h) = 0$ and $H(v) \geq 1$ for $v \neq h$, since reaching $h$ from another vertex takes at least one step. Hence $h$ remains the unique minimum after tie-breaking and is a root.

Every non-root has exactly one outgoing pointer, and every pointer strictly decreases $g$. As previously shown, these pointers form a forest. Finiteness implies that each pointer path terminates at a root; the only possible root is $h$. The forest is therefore a spanning tree. The componentwise statement follows by applying the same argument in every non-singleton component and taking the isolated vertex as the root of each singleton component. □

# D TOPOLOGY OF DATASETS

As previously described, each dataset is a sequence of snapshots $G_t = (V_t, E_t)$, where nodes are users ($Q\&A$, Reddit) or addresses (token), and $V_t$ contains only nodes that are active on particular day $t$. Directed records are collapsed to a simple, undirected support, where a self loop is dropped and the records $(u, v)$ and $(v, u)$ become a single edge $\{u, v\}$. We compute the statistics below on the daily simple undirected supports $U_t = (V_t, \mathcal{E}_t)$, before constructing a reference forest. Write $n = |V_t|$, $m = |\mathcal{E}_t|$, and $\beta_0 = \beta_0(U_t)$. The cycle rank is

$$\beta_1(U_t) = m - n + \beta_0(U_t).$$

The Euler characteristic $\chi = n - m = \beta_0 - \beta_1$ is positive exactly when the snapshot has fewer independent cycles than components. In the following table, each entry is the median over non-empty training snapshots, with the 10th and 90th percentiles in brackets depicting relevant spread. Empty snapshots (no records on that day) are counted separately.

Table 4: Datasets and snapshot inventory. "Empty" counts training days on which no record exists, and we note that splits are chronological.

| Domain | Dataset | Train days | Empty | Val / test days | Edge semantics |
|---|---|---|---|---|---|
| Token | Bytom | 181 | 5 | 58 / 59 | token transfer, sender→receiver |
| | Decentraland | 160 | 2 | 51 / 52 | |
| | Golem | 328 | 0 | 105 / 105 | |
| Q&A | MathOverflow | 1645 | 1 | 353 / 353 | answer or comment, replier→author |
| | AskUbuntu | 1830 | 555 | 392 / 393 | |
| | SuperUser | 1941 | 131 | 416 / 416 | |
| Reddit | Gaming | 255 | 0 | 55 / 55 | reply, replier→parent author |
| | Politics | 255 | 0 | 55 / 55 | |
| | Movies | 255 | 0 | 55 / 55 | |

## D.1 SIZE AND SPARSITY

Table 5: Size and local structure of a typical snapshot. The median snapshot mean degree is below 2 for each Token and Q&A dataset and between 2.4 and 3.0 for the Reddit datasets. Separately, the forest-share statistics in Table 7 show that a maximal spanning forest retains approximately $90 - 97\%$ of the support edges on a typical Token or Q&A. We let *Hub share* be the fraction of all edges incident to the single highest-degree node. *Leaf* and *deg. 2* are the fractions of nodes of degree exactly 1 and 2 to characterize nodes with few connections. *Recip.* is the fraction of directed records that collapse onto an already-present support edge (i.e. the reverse pair also occurred that day). *Loops* denotes the median number of self-records per day, which the support drops.

| Dataset | $n$ | $m$ | $\bar{d}$ | $d_{\max}$ | Hub share | Leaf | Deg. 2 | Recip. | Loops |
|---|---|---|---|---|---|---|---|---|---|
| Bytom | 510 [214, 1647] | 480 [200, 1443] | 1.87 | 173 [43, 689] | 0.34 | 0.82 | 0.15 | 0.002 | 0 |
| Decentraland | 237 [8, 1065] | 192 [4, 1045] | 1.64 | 54 [1, 340] | 0.31 | 0.85 | 0.12 | 0.000 | 0 |
| Golem | 623 [29, 2337] | 587 [25, 2288] | 1.88 | 192 [11, 975] | 0.43 | 0.79 | 0.18 | 0.001 | 0 |
| MathOverflow | 153 [116, 191] | 130 [90, 192] | 1.67 | 9 [6, 16] | 0.07 | 0.60 | 0.20 | 0.084 | 33 |
| AskUbuntu | 299 [106, 418] | 240 [94, 339] | 1.65 | 14 [8, 25] | 0.07 | 0.65 | 0.16 | 0.108 | 54 |
| SuperUser | 345 [11, 513] | 314 [8, 444] | 1.74 | 16 [5, 27] | 0.06 | 0.63 | 0.17 | 0.111 | 70 |
| Gaming | 12393 [9813, 18471] | 15072 [11620, 24774] | 2.44 | 444 [250, 873] | 0.03 | 0.61 | 0.18 | 0.112 | 0 |
| Politics | 4582 [3342, 5758] | 6774 [4832, 8747] | 2.96 | 292 [159, 510] | 0.04 | 0.54 | 0.19 | 0.158 | 0 |
| Movies | 3798 [2793, 5148] | 4434 [3174, 6376] | 2.38 | 278 [149, 655] | 0.06 | 0.62 | 0.18 | 0.105 | 0 |

Mean degree is below 2 on every token and Q&A dataset, while on Reddit it is 2.4–3.0. Furthermore, the Q&A and Reddit graphs also have hubs, but they carry 3–7% of the edges, so these graphs are not stars. See Table 6 for further detail.

### D.2 CONNECTIVITY

Table 6: Component structure. $\beta_0$ is the number of connected components. *Nontriv.* counts components with at least three vertices; *dyad share* is the fraction of components consisting of two vertices joined by one edge; *giant* refers to the largest connected component, reported as a fraction of vertices and by its numbers of vertices and edges.

| **Dataset** | $\beta_0$ | **Non-triv.** | **Dyad share** | **Giant frac.** | **Giant** $n$ | **Giant** $m$ |
|---|---|---|---|---|---|---|
| Bytom | 59 [25, 191] | 5 [2, 10] | 0.93 | 0.71 [0.48, 0.93] | 360 | 403 |
| Decentraland | 45 [4, 119] | 5 [0, 12] | 0.91 | 0.54 [0.25, 0.81] | 106 | 111 |
| Golem | 67 [5, 230] | 6 [1, 23] | 0.87 | 0.76 [0.56, 0.85] | 442 | 465 |
| MathOverflow | 33 [15, 46] | 10 [4, 16] | 0.53 | 0.44 [0.15, 0.80] | 67 | 73 |
| AskUbuntu | 71 [15, 113] | 11 [3, 20] | 0.64 | 0.50 [0.35, 0.73] | 130 | 148 |
| SuperUser | 72 [3, 128] | 12 [2, 21] | 0.64 | 0.56 [0.42, 0.77] | 195 | 231 |
| Gaming | 520 [464, 590] | 115 [97, 135] | 0.78 | 0.90 [0.88, 0.93] | 11155 | 14352 |
| Politics | 163 [118, 201] | 31 [21, 41] | 0.81 | 0.92 [0.89, 0.94] | 4197 | 6575 |
| Movies | 187 [153, 237] | 47 [36, 63] | 0.75 | 0.88 [0.82, 0.91] | 3302 | 4150 |

We see that most components are isolated single interactions on every dataset, where between half and $93\%$ of components are dyads. Here a dyad is a connected component consisting of two vertices joined by one support edge; that edge may represent repeated directed interactions. A large $\beta_0$ therefore does not indicate many communities, but instead indicates many one-off pairs. So, we see that the substantive difference between domains is the giant component, which holds $88$–$92\%$ of nodes on Reddit, $54$–$76\%$ on token (the hub's main contributing star), and only $44$–$56\%$ on Q&A, where the remaining nodes are spread over $10$–$12$ non-trivial components.

### D.3 CYCLE STRUCTURE

Table 7: Cycle rank and related quantities. $\beta_1$ columns give the median, the 10th/90th percentiles, and the maximum over training days. *Days* $\beta_1 > 0$ is the fraction of non-empty days with at least one cycle; *in giant* the median fraction of $\beta_1$ carried by the giant component; *triangles* the median number of 3-cliques; $\chi = n - m$; *forest share* $= (n - \beta_0)/m$ is the fraction of edges retained by any spanning forest containing one spanning tree per component,and *chord share* $= \beta_1/m$ the fraction of edges outside that forest.

| **Dataset** | $\beta_1$ | $[p_{10}, p_{90}]$ | $\max \beta_1$ | **Days** $\beta_1 > 0$ | **In giant** | **Triangles** | $\chi$ | **Forest** | **Chord** |
|---|---|---|---|---|---|---|---|---|---|
| Bytom | 31 | [10, 126] | 1512 | 0.97 | 1.00 | 1 | $+27$ | 0.93 | 0.07 |
| Decentraland | 6 | [0, 127] | 390 | 0.77 | 1.00 | 0 | $+19$ | 0.97 | 0.03 |
| Golem | 25 | [0, 170] | 742 | 0.89 | 1.00 | 10 | $+12$ | 0.95 | 0.05 |
| MathOverflow | 8 | [2, 40] | 102 | 0.98 | 0.93 | 5 | $+24$ | 0.93 | 0.07 |
| AskUbuntu | 16 | [5, 38] | 179 | 0.98 | 1.00 | 5 | $+48$ | 0.92 | 0.08 |
| SuperUser | 34 | [0, 68] | 599 | 0.89 | 1.00 | 10 | $+35$ | 0.90 | 0.11 |
| Gaming | 3132 | [2090, 6957] | 35512 | 1.00 | 1.00 | 384 | $-2625$ | 0.78 | 0.22 |
| Politics | 2338 | [1494, 3165] | 4877 | 1.00 | 1.00 | 458 | $-2187$ | 0.65 | 0.35 |
| Movies | 898 | [537, 1455] | 3313 | 1.00 | 1.00 | 171 | $-707$ | 0.80 | 0.20 |

Within Table 7, Token and Q&A snapshots are forests alongside some chords, where $\chi > 0$, the median cycle rank remains relatively low, and a spanning forest retains 90-97% of the edges. Reddit snapshots shown to be different to this, as $\chi$ is well below zero, where every day contains cycles (the quietest Gaming day containing $1406$), and a spanning forest discards 20-35% of all edges. On each of the nine datasets the median fraction of cycle rank carried by the gan component ranges from $0.93$ to $1.00$; dyad components are trees, while other small components may also contain cycles.

Essentially, we see that triangles are generally absent from token data (0–10 per day on graphs of hundreds of edges), reflecting the bipartite-like structure of exchange-mediated transfers; we then see that addresses transact with the hub, largely not with one other. Q&A graphs have a few triangles per day, but again we see that Reddit graphs have hundreds. The presence of triangles implies that the flag complex has dimension at least two; larger cliques produce simplices of higher dimension. Filling triangles can reduce the first Betti number relative to that of the graph viewed as a one-dimensional complex. The Morse constructions in this paper act on the one-dimensional support graph, and comparisons with persistent homology methods should therefore specify the complex used by each method.

### D.4 Non-Stationary across the training window

We note that token training windows begin at or near each token's launch. Decentraland's first fifth of training days has a median of $8$ active addresses, while its last fifth has $552$. Golem grows from $29$ to $1751$. The validation and test periods lie later, so a token model is trained on networks one to two orders of magnitude smaller than those it is evaluated on. We see that AskUbuntu and SuperUser show a similar (though milder) onset, and in addition AskUbuntu has no records on $555$ of its $1830$ training days. We see that MathOverflow is stationary in size but its cycle rank falls by an order of magnitude across the window. In exploring Reddit, we see that Gaming and Movies grow slightly and Politics shrinks by about a quarter, so the Politics test period is a sparser graph than its training period. These drifts should be kept in mind when reading the within-dataset values, as the test segment is not drawn from the same distribution as the training segment on any token dataset. The details of this paragraph are captured within Table 8.

It is now only fair that we examine the per-snapshot distribution.

Table 8: Drift across the training window. Each entry is the median over the first and last fifth of the training days.

| | $n$ | | $m$ | | $\beta_1$ | |
|---|---|---|---|---|---|---|
| **Dataset** | first | last | first | last | first | last |
| Bytom | 434 | 1056 | 483 | 921 | 75 | 58 |
| Decentraland | 8 | 552 | 4 | 555 | 0 | 77 |
| Golem | 29 | 1751 | 25 | 1635 | 1 | 94 |
| MathOverflow | 142 | 148 | 164 | 110 | 36 | 4 |
| AskUbuntu | 109 | 371 | 103 | 297 | 13 | 21 |
| SuperUser | 11 | 485 | 8 | 397 | 0 | 35 |
| Gaming | 10902 | 11565 | 13048 | 14042 | 2594 | 2978 |
| Politics | 4823 | 3692 | 7250 | 5571 | 2548 | 1976 |
| Movies | 3146 | 3482 | 3651 | 4039 | 623 | 876 |

As in previous sections, tables above report medians and percentiles rather than means and standard deviations because the per-snapshot statistics are not normally distributed. Figure6 shows the empirical distributions of four of them over the training snapshots of each dataset, Figure7 shows the cycle rank in time order, and Table 9 gives the corresponding shape diagnostics.

We now discuss the four main patterns that are seen in this analysis. First, we note that sizes and cycle counts are right-skewed, with heavytails on Token and Gaming. Bytom's cycle rank has skewness $7.8$ and excess kurtosis $77$, while its mean is more than twice its median (31), and a single day (day $\sim$13; note that $\beta_1 = 1512$) contributes $13\%$ of all cycles counted over the training window. We further see that Decentraland's mean $\beta_1$ is nearly seven times its median. Gaming's $n$, $\beta_1$ and $d_{\max}$ all have excess kurtosis above 12, driven by a sustained surge between training days $110$ and $150$ during which the cycle rank increased three- to tenfold above its median, peaking at $35{,}512$ on day 130, and stayed elevated for about six weeks (Figure 7). Based on the structure of these datasets, we note that solely a mean and standard deviation would misrepresent a typical day by a factor of two or more.

Furthermore, we can see that every token panel in the first three rows is composed of a cluster of very small days and a cluster of large days, where we can view small days to be $n < 50$. We note that these are not noise, but instead two halves of the growth that can be seen in Table 8, where the small mode is the period before which the token is active on an exchange, and the larger mode follows. In the early period, we see that Decentraland's $\beta_1$ is zero on $23\%$ of its training days (Golem $11\%$). We note that logarithmic transform does not repair the distribution

Table 9: Shape diagnostics for the per-snapshot distributions of $n$, $\beta_1$ and $d_{\max}$. *Skew* and *ex. kurt.* are the sample skewness and excess kurtosis of the raw values; a normal distribution has both equal to 0. *Mean/med.* is the ratio of mean to median. *Skew (log)* is the skewness after a logarithmic transform of the positive values. $p$ is the Shapiro–Wilk $p$-value on the raw values and on their logarithms; every raw test rejects normality at $p < 0.01$ except Politics $n$ ($p = 0.06$) and Gaming giant share ($p = 0.28$, not shown).

| | $n$ | | | | | $\beta_1$ | | | | | $d_{\max}$ | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **Dataset** | skew | ex. kurt. | mean/med. | skew (log) | $p_{\log}$ | skew | ex. kurt. | mean/med. | skew (log) | $p_{\log}$ | skew | ex. kurt. | mean/med. | skew (log) | $p_{\log}$ |
| Bytom | +3.0 | 11.4 | 1.61 | −2.4 | < .01 | +7.8 | 77.0 | 2.15 | +0.2 | < .01 | +3.2 | 10.9 | 1.82 | −1.2 | < .01 |
| Decentraland | +1.8 | 3.3 | 1.73 | −1.0 | < .01 | +2.4 | 7.3 | 6.87 | +0.0 | < .01 | +2.5 | 7.8 | 2.27 | −0.8 | < .01 |
| Golem | +1.0 | 1.3 | 1.54 | −0.3 | < .01 | +2.6 | 12.0 | 2.56 | −0.4 | < .01 | +1.2 | 1.0 | 2.01 | −0.4 | < .01 |
| MathOverflow | −0.6 | 2.3 | 1.00 | −4.8 | < .01 | +2.0 | 3.9 | 1.89 | +0.0 | < .01 | +1.7 | 4.4 | 1.13 | +0.4 | < .01 |
| AskUbuntu | −0.1 | −1.0 | 0.91 | −2.8 | < .01 | +2.8 | 18.7 | 1.23 | −0.6 | < .01 | +1.4 | 4.0 | 1.11 | −1.1 | < .01 |
| SuperUser | −0.7 | 0.2 | 0.97 | −2.4 | < .01 | +5.0 | 61.5 | 1.12 | −1.3 | < .01 | +0.4 | 1.5 | 1.05 | −2.0 | < .01 |
| Gaming | +2.8 | 12.0 | 1.08 | +1.3 | < .01 | +4.5 | 28.6 | 1.35 | +1.4 | < .01 | +4.3 | 27.7 | 1.20 | +0.7 | < .01 |
| Politics | +0.3 | 0.5 | 0.99 | −0.4 | .01 | +0.5 | 0.9 | 1.01 | −0.5 | < .01 | +1.7 | 3.8 | 1.12 | +0.2 | .09 |
| Movies | +1.3 | 2.8 | 1.03 | +0.4 | .08 | +1.7 | 4.8 | 1.09 | +0.2 | .57 | +4.5 | 25.1 | 1.37 | +1.0 | < .01 |

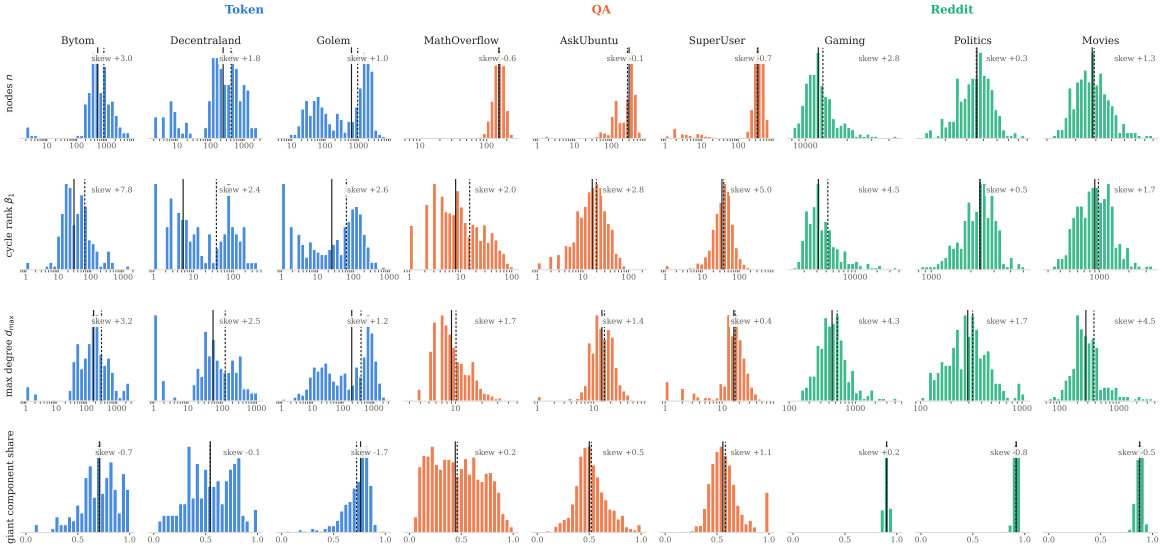



Figure 6: Per-snapshot distributions of the number of nodes $n$, the cycle rank $\beta_1$, the maximum degree $d_{\max}$ and the giant-component share, one histogram per dataset over its non-empty training days. The solid vertical line is the median and the dashed line the mean, both computed over all non-empty training days. The first three rows use a logarithmic horizontal axis, on which days with a zero value are omitted from the histogram but included in the median, the mean and the annotated skewness; the annotated skewness is the raw-scale value of Table 9. Color here indicates the domain.

as the log-skewness is strongly negative ($-2.4$ for Bytom $n$) since the left mode remains. The token training set is a mixture of two populations, and a model that sees both is fitting a network in two different states.

We also see that the three Q&A datasets have $n$ and $m$ distributions that are close to symmetric on the raw scale (skewness between $-0.7$ and $-0.1$, mean within $10\%$ of the median) apart from a thin left tail of onset days. Their $\beta_1$ distributions are nevertheless right-skewed with excess kurtosis 4–62, where SuperUser has a median of 34 cycles but a maximum of 599. The cycle rank of a Q&A day is a much more variable quantity than its size, as we see it depends on how many answers landed on a few popular questions. Furthermore, MathOverflow's cycle rank also drifts downward across the window by nearly an order of magnitude (Figure 7), so its distribution is a superposition of an early high-$\beta_1$ regime and a late low one.

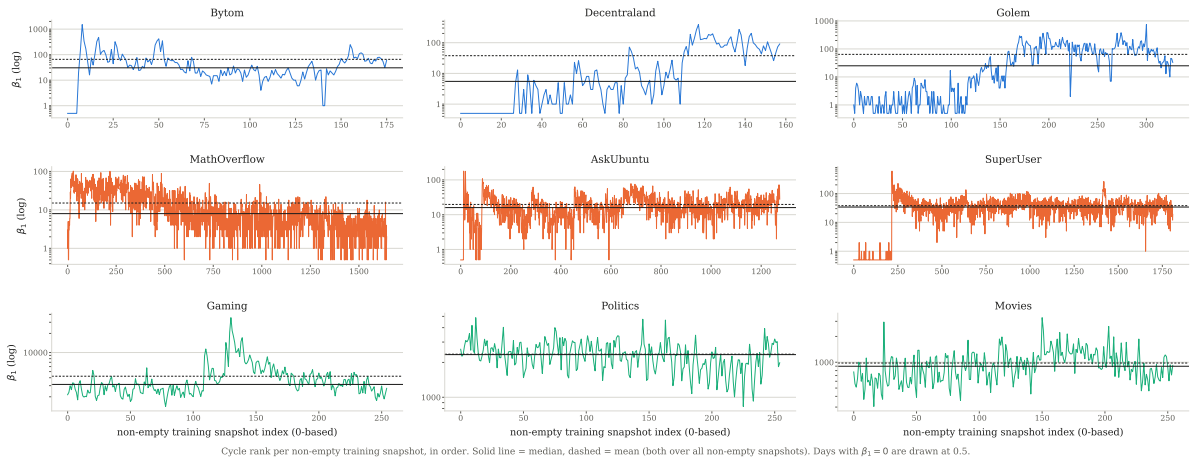



Figure 7: Cycle rank $\beta_1$ of each non-empty training snapshot in chronological order; the horizontal axis is the 0-based index among the non-empty training snapshots, which differs from the training-day index on datasets with empty days. Logarithmic vertical axis. Solid line: median; dashed line: mean, both over all non-empty training days. Days with $\beta_1 = 0$ are drawn at 0.5.

# E Additional Method Details

## E.1 Encoding and Historical Statistics

Count features are log-transformed and standardized using source-training statistics. The target-position predictor uses a residual connection from the query, followed by LayerNorm and an MLP. Positions with no eligible target nodes contribute no loss. Here $\sigma$ is the sigmoid function, and $\pi_v(q)$ is a sparse temporal personalized PageRank vector updated from real interactions before $q$; its inner product measures overlap in the endpoints' historical neighborhoods. The neural term is weighted by endpoint recency, $a(u, v, q) = \frac{1}{2}\sum_{x\in\{u,v\}} \exp(1 - r_x(q)/\bar{r}(q))$. For each endpoint $x$, $r_x$ is the mean age of its retained interaction records, with missing records assigned the maximum age. The quantity $\bar{r}$ averages these node-level ages over the observed node timeframe. These statistics use the observed prefix, which may extend beyond $\mathcal{H}_q$, and exclude reference arcs. After fixing the neural checkpoint, we select $\eta$ using ranking performance on the source validation set.

## E.2 Computing the Scalar Fields for Morse Reference Forests

The ablation changes the node scalar field used to construct the reference forest. All fields are computed from the same observed context, with the target block excluded during pretraining and only completed history used downstream. No forest uses the same encoder and learning task without reference arcs. The quantities below are node scalar fields that guide the pairing; the discrete gradient is obtained from the resulting node–edge matching.

**Shared graph and statistics.** Let $A(u, v) = \sum_{t\in\mathcal{C}} c_t(u, v)$ denote aggregated directed interaction counts, and let $W = A + A^\top$. The positive off-diagonal entries of $W$ define the undirected support $U$. Incoming strength, outgoing strength, total strength, and net inflow are

$$s_{\text{in}}(v) = \sum_u A(u, v), \quad s_{\text{out}}(v) = \sum_u A(v, u), \quad s(v) = s_{\text{in}}(v) + s_{\text{out}}(v), \quad b(v) = s_{\text{in}}(v) - s_{\text{out}}(v). \tag{15}$$

Self-interaction counts contribute to these statistics, but self-loops are excluded from the support used for pairing. For each connected component, the hub $h$ maximizes $s(v)$, with ties resolved by node identifier.

**Hub distance.** Breadth-first search on $U$ gives the unweighted shortest-path distance $d_U(v, h)$ to the component hub. Within a distance level, a strength correction orders higher-strength nodes earlier:

$$f_{\text{hub}}(v) = d_U(v, h) + \epsilon\left(1 - \frac{s(v)}{s(h)}\right), \qquad \epsilon = 0.25. \tag{16}$$

The correction lies in $[0, \epsilon]$, so it preserves the ordering between distinct distance levels. If the component has zero strength, the correction is set to zero.

**Strength and net flow.** The three count-based fields are computed directly from the aggregated statistics:

$$f_{-s}(v) = -s(v), \qquad f_{+s}(v) = s(v), \qquad f_{-b}(v) = -b(v). \tag{17}$$

Negative strength directs descent toward higher-strength nodes, whereas positive strength reverses this ordering. Negative net flow directs descent toward nodes with greater net inflow. Negation has the same ordering as subtracting each value from the component maximum and avoids an unnecessary large offset.

**Hitting time.** In each non-singleton component, let

$$P(v,u) = \frac{W(v,u)}{\sum_w W(v,w)} \quad (v \neq h), \qquad P(h,u) = \begin{cases} 1, & u = h, \\ 0, & u \neq h. \end{cases}$$

The expected number of steps to reach the absorbing hub satisfies

$$f_{\mathrm{hit}}(h) = 0, \qquad f_{\mathrm{hit}}(v) = 1 + \sum_u P(v,u) f_{\mathrm{hit}}(u), \quad v \neq h. \tag{18}$$

For a component with non-hub node set $R$, let $D = \mathrm{diag}(W\mathbf{1})$. We solve the grounded system

$$(D - W)_{R,R}\, f_{\mathrm{hit},R} = D_{R,R}\mathbf{1}. \tag{19}$$

Self-loop mass remains in $D$ and therefore affects waiting times, although it cancels from the diagonal of $D - W$. Singleton components receive value zero. A finite solution is checked by its relative equation residual and the existence of a lower-valued neighbor for each non-hub node.

**From scalar values to a forest.** Let $f_{\mathrm{raw}}$ denote a scalar field before tie-breaking. We can order vertices lexicographically by $(f_{\mathrm{raw}}(v), \mathrm{id}(v))$ and use distinct comparison scores $f(v)$ that represent this order. So, every strict inequality in $f_{\mathrm{raw}}$ is preserved, and ties are resolved by the fixed node identifiers. The hitting-time equations and the equation-residual check apply to $f_{\mathrm{raw}}$; all comparisons in the forest construction use $f$. The perturbed comparison scores are not required to satisfy the hitting-time equations. For each vertex $v$, define

$$N^-(v) = \{u \in N_U(v) : f(u) < f(v)\}.$$

If this set is nonempty, select

$$p(v) = \operatorname*{arg\,min}_{u \in N^-(v)} \frac{f(v) + f(u)}{2} \tag{20}$$

and pair $v$ with $\{v, p(v)\}$. Nodes without a lower-valued neighbor remain roots. Strict descent excludes directed cycles, and the selected edges form the reference forest. Hub distance and hitting time lead to one root per connected component under this construction; the count-based fields can yield multiple roots. Each selected edge supplies two directed reference arcs, and the same forest is used throughout the window.

### E.3 Reference-Forest Construction Time

We measure the time required to construct a reference forest from a seven-day history window. For each dataset, we select 30 distinct windows at evenly spaced training origins and supply the same windows to all five scalar fields. Each construction runs on a single CPU thread of an Intel Xeon Gold 6530 host, with five timed repetitions after warm-up. Total time includes history aggregation, graph-support preparation, scalar-field computation, and forest generation. All fields use the mean edge extension.

Table 10 reports the mean and sample standard deviation across windows after averaging the five repetitions within each window. The standard deviation therefore reflects variation across historical graphs. Hitting time has the highest construction cost on every dataset, whereas the other four fields have similar total costs.

Reference forests are computed once per distinct historical context and construction configuration, then cached. Repeated visits to the same context load the stored forest, and runs with the same context and configuration can share the cache across seeds and training stages. These measurements therefore describe a one-time preprocessing cost for each cache entry. A new context or a corrupted history requires its own entry.

Table 10: Total reference-forest construction time per history window in milliseconds, reported as mean ± sample standard deviation across 30 training windows. Each window is timed five times.

| Dataset | Hub distance/ms | Negative strength/ms | Negative net flow/ms | Positive strength/ms | Hitting time/ms |
|---|---|---|---|---|---|
| Bytom | $6.13 \pm 4.62$ | $5.86 \pm 4.44$ | $5.99 \pm 4.51$ | $5.87 \pm 4.52$ | $389.81 \pm 371.20$ |
| Decentraland | $3.42 \pm 2.29$ | $3.22 \pm 2.16$ | $3.29 \pm 2.22$ | $3.24 \pm 2.18$ | $339.74 \pm 236.41$ |
| Golem | $7.38 \pm 6.47$ | $7.04 \pm 6.18$ | $7.26 \pm 6.38$ | $7.03 \pm 6.17$ | $451.01 \pm 428.60$ |
| MathOverflow | $1.92 \pm 0.27$ | $1.87 \pm 0.27$ | $1.89 \pm 0.27$ | $1.87 \pm 0.27$ | $34.53 \pm 15.65$ |
| AskUbuntu | $2.20 \pm 1.39$ | $2.12 \pm 1.33$ | $2.16 \pm 1.36$ | $2.11 \pm 1.32$ | $120.14 \pm 128.66$ |
| SuperUser | $3.25 \pm 1.32$ | $3.12 \pm 1.27$ | $3.14 \pm 1.28$ | $3.10 \pm 1.26$ | $184.07 \pm 138.53$ |
| Movies | $42.03 \pm 7.97$ | $40.95 \pm 7.85$ | $41.50 \pm 7.91$ | $40.98 \pm 7.86$ | $456.97 \pm 45.56$ |
| Politics | $60.72 \pm 8.16$ | $59.63 \pm 8.17$ | $60.30 \pm 8.25$ | $60.13 \pm 8.25$ | $329.70 \pm 55.91$ |
| Gaming | $166.20 \pm 71.50$ | $162.96 \pm 70.42$ | $164.75 \pm 72.36$ | $162.74 \pm 70.60$ | $1191.75 \pm 160.09$ |

### E.4 Realized Masking Rates

The per-node sampling fraction does not equal the fraction of interaction records removed from a target snapshot. We measure the realized rate under the default rule, where every active node samples $\lceil 0.25|I_t(v)| \rceil$ incident records and the union is deleted. For each dataset and seed, we sum the numbers of removed records and original records over the target snapshots in 6,000 pretraining iterations, then divide the two totals. Repeatedly sampled snapshots contribute each time they are used. Each removed record is counted once per snapshot exposure, even if both endpoints select it. Context snapshots and reference arcs are excluded, and interaction counts carried by records do not weight the ratio.

Table 11 reports the mean and sample standard deviation of these percentages across seeds 21, 42, and 63. The two columns distinguish source training on the original within-dataset splits from source training on the calendar-aligned transfer splits. The statistics cover the first 6,000 iterations of the registered training trajectories. No masking is applied during downstream evaluation.

Table 11: Actual removal of target-snapshot interaction records under the default masking rule (%, mean ± sample standard deviation across seeds). Each seed covers 6,000 pretraining iterations. Transfer statistics refer to source pretraining.

| Domain | Dataset | Within-dataset | Transfer (source) |
|---|---|---|---|
| Token | Bytom | $70.38 \pm 0.02$ | $72.83 \pm 0.07$ |
| Token | Decentraland | $69.57 \pm 0.04$ | $71.33 \pm 0.05$ |
| Token | Golem | $70.13 \pm 0.01$ | $70.35 \pm 0.02$ |
| QA | MathOverflow | $70.51 \pm 0.03$ | $69.20 \pm 0.06$ |
| QA | AskUbuntu | $72.51 \pm 0.02$ | $71.54 \pm 0.06$ |
| QA | SuperUser | $69.92 \pm 0.02$ | $69.78 \pm 0.02$ |
| Reddit | Gaming | $71.50 \pm 0.06$ | $71.17 \pm 0.15$ |
| Reddit | Politics | $65.23 \pm 0.01$ | $65.45 \pm 0.02$ |
| Reddit | Movies | $73.64 \pm 0.01$ | $73.85 \pm 0.01$ |

The realized rates substantially exceed 25% because sampling is performed at both endpoints and rounded upward for each node. In particular, nodes with a single incident record always select it.

Nodewise sampling distributes the corruption across active nodes, including those with few interactions. Rounding upward ensures that each active node has at least one incident record withheld. For a node with only one record, removing that record is necessary to hide its direct interaction evidence at that time. Preserving an incident record for every node would leave these degree-one nodes' direct interactions visible. The resulting deletion rate therefore reflects the degree distribution as well as the sampling fraction.

Prediction draws on the entire observed window. Only the one-to-five-snapshot target block is corrupted; the surrounding snapshots remain visible, and the loss is evaluated on nodes also observed in that context. A node that loses all current interactions can still draw on its observations at other times. MoJEPA additionally supplies reference routes built exclusively from the context snapshots, allowing historical relations to support current-state

exchange. The teacher retains the complete target view. The student predicts these complete-view representations from the remaining temporal and relational context. The matched results in Appx. G.1 show that pretraining under this rule improves neural-only prediction over random initialization on Token and QA, supporting its usefulness despite the high target-snapshot deletion rates.

# F Experimental Settings

## F.1 Datasets

**Graph construction.** Token graphs retain directed sender-to-recipient transfers, including interaction counts and transaction attributes. QA graphs combine answers and comments into directed user interactions and retain their multiplicities. Reddit graphs connect reply authors to the authors of their direct parent posts or comments. Self-replies and edges with deleted or removed authors are excluded, and repeated directed user pairs are merged within each day.

**Temporal resolution and splits.** Token, QA, and Reddit use daily snapshots. Token and QA retain interaction multiplicities. Within-dataset evaluation uses the original observation intervals and splits described in Appx. F.4. For cross-dataset transfer, we restrict all three datasets in each domain to their common observation interval, align timestamps in UTC, and retain empty snapshots. We split this interval chronologically into training (50%), validation (20%), and test (30%), rounding the first two lengths down. Table 12 gives the resulting snapshot counts for transfer.

Table 12: Datasets and calendar-aligned splits for cross-dataset transfer. Counts refer to snapshots in each dataset, including empty snapshots. Observation intervals are half-open and expressed in UTC.

| Domain | Observation interval | Resolution | Train / Val. / Test |
|---|---|---|---|
| Token | 2017-08-17 – 2018-05-07 | Daily | 131 / 52 / 80 |
| QA | 2009-09-29 – 2016-03-07 | Daily | 1175 / 470 / 706 |
| Reddit | 2013-02-01 – 2014-02-01 | Daily | 182 / 73 / 110 |

## F.2 Baselines

1. **EGCH (EvolveGCN-H)**(Pareja et al., 2020) uses a recurrent network to evolve graph convolution parameters across snapshots.
2. **ROLAND**(You et al., 2022) maintains hierarchical node states and updates them at successive GNN layers.
3. **SiST-GNN** (Roy & Dasgupta, 2026) performs spatial and temporal message passing jointly by connecting current node features to historical hidden states.
4. **WinGNN** (Zhu et al., 2023) learns temporal dependencies through gradient aggregation over randomly sampled snapshot windows.
5. **HawGCN (Hawkes-GCN)** (Qi et al., 2025) is the GCN variant of SFDyG, which fuses historical snapshots and applies Hawkes-inspired temporal weighting.
6. **UnderGS** (Wu et al., 2026a) uses a temporal neighbor store and late-snapshot gradient aggregation to learn from evolving graph structure.
7. **EAGLE** (Li et al., 2025b) combines recent-neighbor representations with temporal personalized PageRank (TPPR) to capture short-term activity and longer-range structural information.
8. **TNCN** (Zhang et al., 2025) incorporates temporal common neighbors into pairwise representations for link prediction.
9. **SG-JEPA** (Zhang et al., 2026) learns predictive embeddings from temporal context and target views using spiking neural networks.
10. **CPDG** (Bei et al., 2024) uses contrastive pretraining for dynamic graph neural networks.
11. **TMetaNet** (Li et al., 2025a) uses Dowker zigzag persistence to represent evolving graph topology and guides meta-learning parameter updates with distances between these topological representations.

Table 13: MoJEPA configuration shared across datasets. Budget scans vary only the indicated pretraining or fine-tuning limit.

| Setting | Value |
|---|---|
| Pretraining / downstream window | 10 / 7 snapshots |
| Target-block length | Uniform over 1–5 snapshots |
| Active-node sampling fraction | 1.0 |
| Per-node incident-record fraction $\rho$ | 0.25 |
| Representation dimension | 64 |
| Spatial encoder | 2 directed GINE layers |
| Temporal encoder | 1 BiGRU layer, 32 units per direction |
| Predictor attention heads / MLP width | 4 / 128 |
| Dropout | 0.1 |
| Teacher EMA coefficient $\tau$ | 0.996, fixed |
| Optimizer | AdamW |
| Pretraining learning rate | $10^{-4}$ |
| Fine-tuning learning rates | Encoder $10^{-4}$; link head $10^{-3}$ |
| Weight decay | $10^{-4}$ |
| Gradient norm limit | 1.0 |
| Main pretraining / fine-tuning budget | 6,000 iterations / at most 2,000 updates |
| Fine-tuning validation interval | 100 updates |

**Implementation settings.** All methods use the same evaluation queries and candidates, while retaining their respective history windows, objectives, and training samplers. Input normalization is fitted on source training data, and checkpoint selection and score calibration use source validation MRR. MoJEPA and SG-JEPA receive 6,000 pretraining updates followed by at most 2,000 updates jointly fine-tuning the encoder and link readout. MoJEPA uses 64-dimensional representations, ten-snapshot pretraining windows, and seven-snapshot inputs for link prediction; SG-JEPA uses eight-snapshot pretraining windows with the first seven predicting the last. ROLAND traverses the training sequence once and optimizes within each snapshot. EGCH, SiST-GNN, WinGNN, EAGLE, TNCN, HawGCN, and UnderGS use at most 100 epochs. The shared encoder settings are held fixed across domains. Experiments were run on Linux servers, including one equipped with four NVIDIA RTX A6000 GPUs, each with 48 GB of memory.

### F.3 MoJEPA Implementation Details

**Architecture and masking.** Table 13 lists the shared configuration. The encoder uses two directed GINE layers followed by a single-layer bidirectional GRU. Each GRU direction has 32 hidden units, giving a 64-dimensional representation. The predictor uses four-head cross-attention and an MLP with hidden width 128. The directed link head is an MLP with dimensions $128 \to 128 \to 1$ and a ReLU between its linear layers. Downstream prediction uses the encoder output at the final history position.

Each pretraining iteration samples one ten-snapshot window from the source training split. The target-block length is sampled uniformly from $\{1, \ldots, 5\}$, and its start is sampled uniformly among positions that leave both boundary snapshots observed. All active nodes in each target snapshot participate in masking. Each samples $\lceil 0.25|I_t(v)| \rceil$ incident directed records without replacement, and the union is removed. The value 0.25 is the per-node sampling fraction, not a prescribed global deletion rate. No neighbor-retention constraint is imposed.

**Optimization and model selection.** Learning rates remain constant. Pretraining updates the student and predictor, and the teacher receives an EMA update after each iteration. Windows without eligible targets count toward the iteration budget but do not update the student or predictor. The final EMA teacher initializes the downstream encoder. Each fine-tuning update samples, with replacement, one source-training date with eligible queries and uses all its queries and their candidate destinations. Thus, the number of queries per update varies with the date. We average binary cross-entropy over candidates within each query and then over queries. Every 100 updates, we evaluate the neural score on the source validation split and retain the checkpoint with the highest MRR; ties retain the earlier checkpoint.

**Historical scores and calibration.** The TPPR implementation uses $\alpha = 0.9$, $\beta = 0.8$, a sparse-vector capacity of 100, and its `mul_wo_norm` update mode. Updates process completed snapshots chronologically, with directed endpoint pairs ordered lexicographically within each snapshot. Endpoint recency is computed from the latest 15 incident records retained per node. Both statistics exclude reference arcs. After selecting the neural checkpoint, we choose the fusion coefficient from

$$\eta \in \{1, 2, 3, 5, 8\} \times \{10^{-6}, 10^{-5}, \ldots, 1\}$$

using source-validation MRR, retaining the smaller coefficient in a tie. The selected coefficient, feature scaling, encoder, and link head remain fixed during transfer.

**Numerical forest construction.** We solve the grounded hitting-time systems in double precision using SciPy's sparse direct solver. Component weights are divided by their maximum before forming the system, which preserves the hitting times. For a system $Ah = b$, we check $\max_i |(Ah - b)_i|/b_i \leq 10^{-6}$, finite solutions, and strict descent from every non-hub node. Invalid systems or failed checks raise an error rather than substituting a different forest. Gaming uses the separately registered Jacobi-preconditioned conjugate-gradient implementation for grounded systems of size at least 256; smaller systems use the direct solver. Its relative tolerance is $10^{-12}$, absolute tolerance is zero, and the iteration limit is 10,000 per solve. It permits up to three residual-correction solves and requires a final componentwise relative residual at most $10^{-9}$. A 60-second per-component time limit and failed convergence checks terminate the solve. The Gaming configuration solves the same hitting-time equations; floating-point differences near ties can change selected forest edges.

### F.4 Within-Dataset Link Prediction

**Training and evaluation.** Within-dataset evaluation retains each dataset's original observation interval and chronological split. For Token, the initial seven days provide history only; the remaining target days are split into 60% training, 20% validation, and 20% test. QA uses its original 70%/15%/15% chronological splits. Reddit uses 365 daily snapshots from 2013-02-01 through 2014-01-31, with 255 training, 55 validation, and 55 test snapshots; test indices are 310–364 under zero-based indexing. Parameters and input normalization are fitted on training data. Checkpoints and score-fusion coefficients are selected on validation data. Test predictions use only observations preceding the predicted snapshot. The node catalog is fixed for each retained dataset interval.

**Query construction.** At the start of snapshot $t$, predictions use only observations before $t$. Let $A_t$ denote the nodes active in the preceding snapshots, and let $N_t(u)$ be the outgoing neighbors of $u$ in $G_t$. For $u \in A_t$, the positive destinations are

$$Q_t(u) = \big(N_t(u) \setminus N_{t-1}(u)\big) \cap A_t. \tag{21}$$

A positive link may therefore have occurred before snapshot $t-1$. For each source node $u$ with $Q_t(u) \neq \varnothing$, define the eligible negative pool as

$$B_t(u) = A_t \setminus \big(\{u\} \cup N_t(u) \cup N_{t-1}(u)\big). \tag{22}$$

We retain the query only when $|B_t(u)| \geq 20$ and sample twenty distinct negatives uniformly without replacement from $B_t(u)$. The sampled candidates are fixed across methods and model seeds. MRR is computed over the retained queries.

### F.5 Cross-Dataset Transfer

**Transfer directions.** Each domain contains three datasets. We evaluate all six ordered source–target pairs with distinct datasets, giving eighteen directions overall. The domain groups are (Bytom, Decentraland, Golem), (MathOverflow, AskUbuntu, SuperUser), and (gaming, politics, movies). A model trained on each source is reused for the two other datasets in its domain.

**Calendar alignment and temporal splits.** Within each domain, the three datasets are restricted to the intersection of their observation periods and share the same chronological 50%/20%/30% split. Table 14 gives the exact boundaries for all six transfer directions in each domain. All timestamps are in UTC, and intervals include their start and exclude their end.

Table 14: Calendar boundaries for cross-dataset transfer. The three consecutive intervals define source training, source validation, and target testing. Daily boundaries are at 00:00 UTC.

| Domain | Start ($b_0$) | Validation ($b_1$) | Test ($b_2$) | End ($b_3$) |
|---|---|---|---|---|
| Token | 2017-08-17 | 2017-12-26 | 2018-02-16 | 2018-05-07 |
| QA | 2009-09-29 | 2012-12-17 | 2014-04-01 | 2016-03-07 |
| Reddit | 2013-02-01 | 2013-08-02 | 2013-10-14 | 2014-02-01 |

### F.6 Evaluation Aggregation

We report mean reciprocal rank (MRR), averaging reciprocal ranks over positive destinations, then over source nodes within each snapshot, and finally over nonempty evaluation snapshots:

$$\mathrm{MRR} = \frac{1}{|\mathcal{T}_{\mathrm{eval}}|} \sum_{t \in \mathcal{T}_{\mathrm{eval}}} \frac{1}{|\mathcal{U}_t|} \sum_{u \in \mathcal{U}_t} \frac{1}{|Q_t(u)|} \sum_{v \in Q_t(u)} \frac{1}{\mathrm{rank}_t(u, v)}, \tag{23}$$

where $\mathcal{T}_{\mathrm{eval}}$ is the set of nonempty evaluation snapshots, $\mathcal{U}_t$ is the set of eligible source nodes at $t$, and $Q_t(u) \neq \varnothing$ contains their positive destinations. Ties receive average ranks. We report the mean and sample standard deviation over three seeds. More formally we let $\mathcal{T}_{\mathrm{eval}} = \{t : |U_t| > 0\}$ and is defined when $|\mathcal{T}_{\mathrm{eval}}| > 0$; an interval with no eligible queries has undefined MRR. Ties receive average ranks.

### F.7 Observation Corruption

Drop removes sampled daily relations and all their counts. Add introduces absent directed relations between historically observed endpoints. Rewire swaps endpoints of pairs of directed edges while preserving unweighted in- and out-degrees; invalid swaps are rejected. Delay postpones sampled relations by one day. Each corruption follows a shared temporal trajectory across methods. Evaluation dates, queries, positive labels, and negative candidates remain fixed.

For each of three model seeds, we evaluate three noise seeds. All features, reference forests, and historical statistics are recomputed from the corrupted prefix. Clean-training feature scales, selected checkpoints, and calibration remain fixed. We average noise seeds within each model seed, then datasets within each domain, before computing the mean and sample standard deviation across model seeds. Incomplete curves are not interpolated.

## G Additional Experiments

### G.1 Ablation of Pretraining and Score Fusion

We separate the contribution of JEPA pretraining from that of historical score fusion. The matched MoJEPA conditions use the same downstream encoder, hitting-time reference forest, link head, training queries, and fine-tuning limit of 2,000 updates. One condition starts from random initialization; the other starts from the EMA teacher after 6,000 pretraining iterations. TPPR-only uses the historical neighborhood inner product without the recency-weighted neural term. Tables 15 and 16 report within-dataset prediction and fixed-parameter transfer separately.

Table 15: Within-dataset ablation of pretraining and score fusion. MRR ($\times 100$), mean $\pm$ sample standard deviation. $P$ is the pretraining iteration count. TPPR-only is deterministic under the fixed candidates; its zero deviation reflects identical repeated evaluations.

| Method | $P$ | Token | QA | Reddit |
|---|---|---|---|---|
| MoJEPA | 0 | $67.11 \pm 0.31$ | $47.22 \pm 0.05$ | $41.72 \pm 0.03$ |
| TPPR-only | — | $33.41 \pm 0.00$ | $46.21 \pm 0.00$ | $40.11 \pm 0.00$ |
| MoJEPA | 6,000 | $\mathbf{78.14 \pm 0.47}$ | $\mathbf{57.39 \pm 0.08}$ | $\mathbf{55.27 \pm 0.03}$ |

JEPA pretraining improves prediction in both domains and both evaluation settings. Models trained from scratch also perform well, showing that the downstream architecture and supervised training account for part of the complete system's performance. TPPR-only generally remains below pretrained MoJEPA, although it outperforms the

Table 16: Cross-dataset transfer ablation of pretraining and score fusion. MRR ($\times 100$), mean $\pm$ sample standard deviation. $P$ is the pretraining iteration count. TPPR-only is deterministic under the fixed candidates; its zero deviation reflects identical repeated evaluations.

| Method | $P$ | Token | QA | Reddit |
|---|---|---|---|---|
| MoJEPA | 0 | $60.08 \pm 0.38$ | $43.32 \pm 0.13$ | $48.63 \pm 0.19$ |
| TPPR-only | — | $34.20 \pm 0.00$ | $46.43 \pm 0.00$ | $40.98 \pm 0.00$ |
| MoJEPA | 6,000 | $\mathbf{76.11 \pm 0.29}$ | $\mathbf{56.58 \pm 0.07}$ | $\mathbf{54.47 \pm 0.05}$ |

randomly initialized model in QA transfer. These results support complementary contributions from pretrained representations and historical scoring rather than attributing the full-system gains to pretraining alone.

### G.2 Observation Robustness on Token and QA

Figure 8 extends the observation-corruption evaluation to Token and QA using the same models, corruption rates, and fixed-query protocol as Section 5.2. MoJEPA maintains the highest domain-averaged MRR among the three methods under deletion, addition, and delay at all tested rates. On Token, one-day delays have a smaller effect than deletion and addition. On QA, deletion produces the largest decline for MoJEPA. As on Reddit, higher MRR under corruption does not imply a uniformly smaller decline from clean performance.

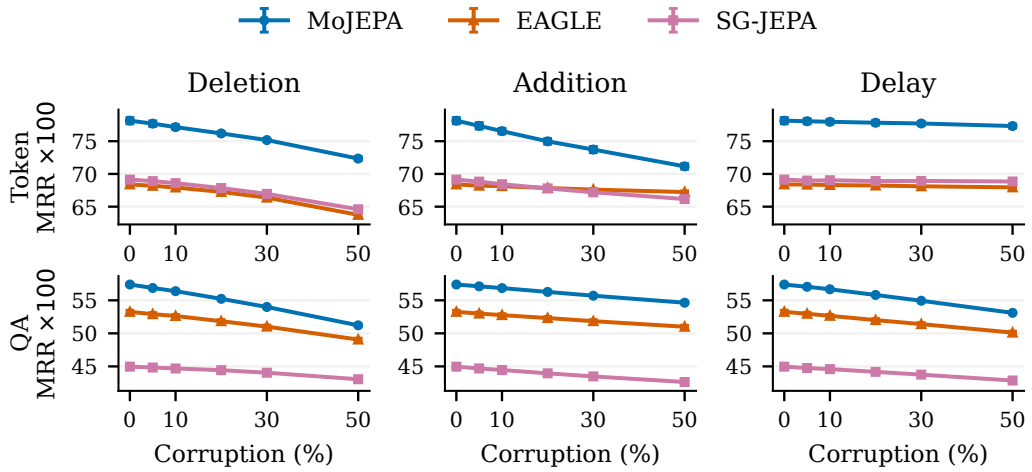



Figure 8: Observation robustness on Token (top) and QA (bottom). Columns show deletion, addition, and delay. Error bars show sample standard deviations across model seeds after averaging noise seeds and datasets.

### G.3 Training Budget

We compare MoJEPA and SG-JEPA under two budget scans. The pretraining scan varies $P \in \{1200, 2400, 4800, 6000, 8000\}$ with the fine-tuning limit fixed at $F = 2000$. The fine-tuning scan varies $F \in \{400, 800, 1200, 1600, 2000, 3000, 4000\}$ with $P = 6000$. Checkpoints are selected on source validation data. Figure 9 separates the two scans for each domain and reports within-dataset MRR. Here, $F$ denotes the update limit rather than the selected checkpoint step.

MoJEPA outperforms SG-JEPA across the available comparisons in both budget scans. Its performance varies little over the tested pretraining budgets, while increasing the fine-tuning budget produces larger gains, particularly on Token. Extending either budget beyond the default setting yields only modest further improvements.

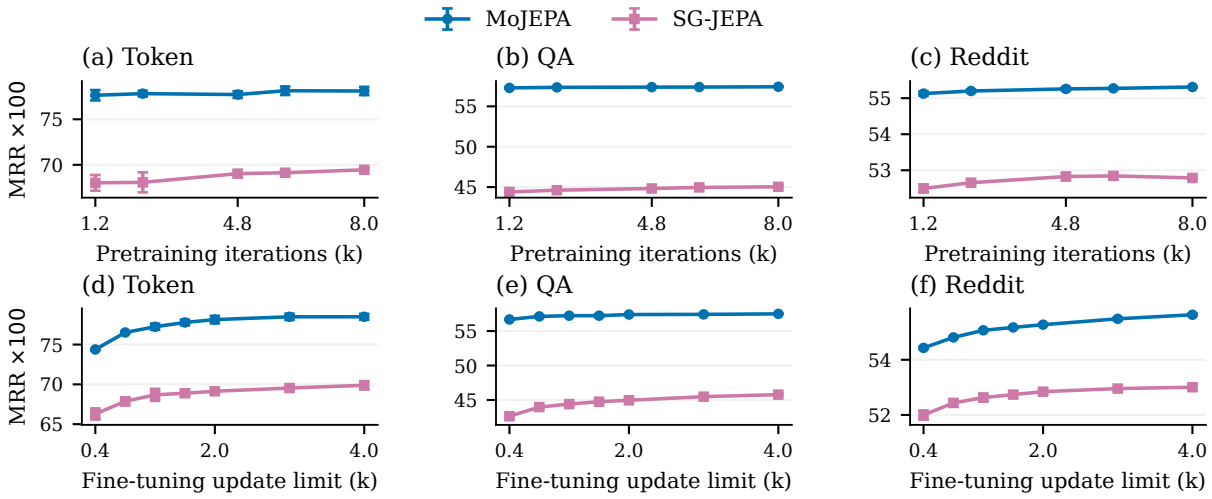



Figure 9: Training-budget experiments. Columns correspond to Token, QA, and Reddit. The top row varies pretraining iterations with $F = 2000$; the bottom row varies the fine-tuning limit with $P = 6000$. Error bars show sample standard deviations of domain-averaged MRR ($\times 100$). Only complete domain estimates are shown.

## H Context-Graph Structure & Morse Reference Forests

Here we examine the structure of aggregated context graphs and the Morse reference forests constructed from them. The daily-support statistics in Appx. D provide dataset background, but do not directly characterize the aggregated contexts on which MoJEPA constructs its reference forests.

The empirical context-graph summaries below concern seven-day histories drawn from the training split, matching the downstream history construction. Pretraining uses a different context: an internal target block of one to five snapshots is excluded from a ten-snapshot window, leaving five to nine observed snapshots on both sides of the target block. Consequently, the seven-day summaries do not characterize the full distribution of pretraining contexts. The structural identities below apply to either context construction.

### H.1 Aggregated Context Graphs and Forest Size

The reference forest is constructed on the aggregated observed-context support, rather than on an individual daily graph. For a prediction day $t$ with seven preceding calendar snapshots, define

$$U_C(t) = \bigcup_{s=t-7}^{t-1} U_s, \qquad n_C = |V_C|, \quad m_C = |\mathcal{E}_C|, \quad c_C = \beta_0(U_C).$$

For readability, dependence on $t$ is suppressed below. Table 17 summarizes the size, connectivity, cycle rank, and component-spanning forest retention of these training contexts. If a discrete gradient on $U_C$ has $r_C$ critical vertices, its paired forest satisfies

$$|\mathcal{E}_F| = n_C - r_C, \qquad |\mathcal{E}_C \setminus \mathcal{E}_F| = \beta_1(U_C) + (r_C - c_C).$$

Hitting time and hub distance give $r_C = c_C$, so their forests contain $n_C - c_C$ edges and omit exactly $\beta_1(U_C)$ chords. These forests preserve the connected components of $U_C$, but satisfy $\beta_1 = 0$. For fields with additional roots, omitted edges do not all need to be chords, as some can connect distinct trees of the selected forest.

For $m_C > 0$, the component-spanning forest share is

$$\frac{n_C - c_C}{m_C} = \frac{2}{\bar{d}_C}\left(1 - \frac{c_C}{n_C}\right), \qquad \bar{d}_C = \frac{2m_C}{n_C}.$$

It is therefore algebraically related to mean degree and component density, rather than an independent structural statistic.

Table 17: Structure of seven-day context graphs in the training data. Entries are medians across the analyzed contexts. The retained-edge fraction is $(n_C - \beta_0(U_C))/m_C$, defined for $m_C > 0$. It gives the fraction of support edges selected by either the hitting-time or hub-distance construction. Medians are reported separately for each quantity, so identities holding within each context need not hold between the displayed column medians.

| Dataset | $n_C$ | $m_C$ | $\beta_0(U_C)$ | $\beta_1(U_C)$ | Retained fraction |
|---|---|---|---|---|---|
| Bytom | 3907 | 4059 | 227 | 453 | 0.88 |
| Decentraland | 1782 | 1650 | 290 | 73 | 0.94 |
| Golem | 3190 | 3511 | 276 | 418 | 0.89 |
| MathOverflow | 561 | 807 | 38 | 271 | 0.66 |
| AskUbuntu | 1481 | 1612 | 212 | 387 | 0.73 |
| SuperUser | 1656 | 2133 | 230 | 648 | 0.69 |
| Gaming | 58544 | 102207 | 663 | 44857 | 0.56 |
| Politics | 20369 | 46310 | 172 | 25975 | 0.44 |
| Movies | 19902 | 31612 | 311 | 11785 | 0.62 |

### H.2 Reference Routes and Observed Benefit

For downstream prediction at time $t$, the forest constructed from the preceding seven snapshots is attached to each of those history snapshots. It therefore makes selected historical relations available for state exchange even at history positions where the corresponding real interaction is absent. The unobserved graph at prediction time $t$ is not an encoder input.

A component-spanning forest preserves connectivity of the aggregated context, but connectivity alone does not characterize access within the encoder's two spatial message-passing layers. Different forests can provide different short paths, and reference arcs have a distinct relation type from real interactions. Neither context connectivity nor the retained-edge fraction therefore establishes whether reference messages duplicate or supplement useful information.

The reference-context ablation (Table 2) reports the largest benefit on Token, smaller changes on QA, and no consistent benefit across fields on Reddit. These observations motivate examining the context graphs and their selected forests, but do not establish that edge retention or fragmentation causes the observed performance differences.

### H.3 Scalar Fields and Forest Fragmentation

The scalar field affects both the number and the placement of reference edges. Fix an aggregated context support $U_C = (V_C, \mathcal{E}_C)$, and let $r_C(f)$ denote the number of roots produced by the minimum-neighbor construction under field $f$, including the prescribed tie-breaking. The resulting forest $F_C(f)$ satisfies

$$|\mathcal{E}_{F_C(f)}| = n_C - r_C(f).$$

Relative to a forest with one tree per connected component, its normalized edge-count deficit is

$$L_C(f) = \frac{(n_C - \beta_0(U_C)) - |\mathcal{E}_{F_C(f)}|}{n_C - \beta_0(U_C)} = \frac{r_C(f) - \beta_0(U_C)}{n_C - \beta_0(U_C)},$$

provided $n_C > \beta_0(U_C)$. Each additional root increases the number of forest components by one and reduces the selected edge count by one. Additional roots are valid critical vertices, rather than failures of the discrete Morse construction.

Hitting time and hub distance produce one root per connected component, giving $L_C(f) = 0$. The strength and net-flow fields can instead produce multiple roots within a component. For example, under positive strength, a context-support leaf whose neighbor has strictly larger aggregated strength is a root. Under negative net flow, roots are local maxima of aggregated net inflow, with ties resolved by the fixed comparison order. These properties depend on the context-level counts and neighborhoods.

Forests selected by different fields need not be nested. Consequently, $L_C(f)$ measures a cardinality deficit, not deletion from a particular reference forest. Fields with identical root counts can also select different edges and yield different branching patterns and path lengths. Root counts therefore characterize forest size and fragmentation, but do not determine the communication routes or their predictive utility.