# Bridging the Divide: End-to-End Sequence-Graph Learning

**Yuen Chen**[1] **Yulun Wu**[2] **Samuel Sharpe**[2] **Igor Melnyk**[2]

**Nam H. Nguyen**[2] **Furong Huang**[3] **C. Bayan Bruss**[2] **Rizal Fathony**[2]
[1]University of Illinois at Urbana–Champaign [2]Capital One
[3]University of Maryland, College Park

## Abstract

Many real–world datasets are both *sequential* and *relational*: each node carries an event sequence while edges encode interactions. Existing methods in sequence modeling and graph modeling often neglect one modality or the other. We argue that sequences and graphs are not separate problems but complementary facets of the same dataset, and should be learned jointly. We introduce BRIDGE, a unified end-to-end architecture that couples a sequence encoder with a GNN under a single objective, allowing gradients to flow across both modules and learning task-aligned representations. To enable fine-grained token-level message passing among neighbors, we add TOKENXATTN, a token-level cross-attention layer that passes messages between events in neighboring sequences. Across two settings, friendship prediction (Brightkite) and fraud detection (Amazon), BRIDGE consistently outperforms static GNNs, temporal graph methods, and sequence-only baselines on ranking and classification metrics.

## 1 Introduction

Modern machine learning increasingly faces settings where data are both *sequential* and *relational*. We study the general case in which an entity may have multiple relations with other entities, where each entity generates a time-stamped event sequence. We adopt user-event modeling [1] as our running example to motivate design choices and evaluation, since social platforms epitomize this sequence-on-graph regime; the approach, however, applies broadly beyond this domain. On social platforms, each user generates a sequence of personal events (e.g., logins, posts) while simultaneously maintaining friendships with others. In e-commerce, a user may purchase items or write reviews as an event sequence, while being connected to others through, for example, friendship relations. In Figure 1, we illustrate the e-commerce setting: each user generates a personal sequence of events (e.g., logins, page visits, purchases, reviews), while also engaging in user-to-user relations that connect them to other users (e.g., sending or receiving gifts, commenting on reviews). Such data are inherently multimodal, combining temporal dynamics at the sequence level with structural dependencies across the network, making it essential to model both modalities jointly for accurate prediction.

There have been efforts to incorporate temporal and sequential information into relational models, i.e., graphs. For example, the temporal heterogeneous graph formulations, such as CTDG [2] and its derived models [3–5], encode relational events between multiple entities with a timestamp assigned to each relation. However, this formulation only works for events describing interactions with other entities, such as transitive actions. User events, on the other hand, describe intransitive actions (e.g., 'sign in', 'login', and 'subscribe') or status changes (e.g., 'payment successful', 'payment declined', 'subscription status update') that involve only a single user. Forcing such events into a graph requires self-loops or introducing event nodes, resulting in a misleading graph where, for instance, everyone who 'logs in' looks related through a shared login node.

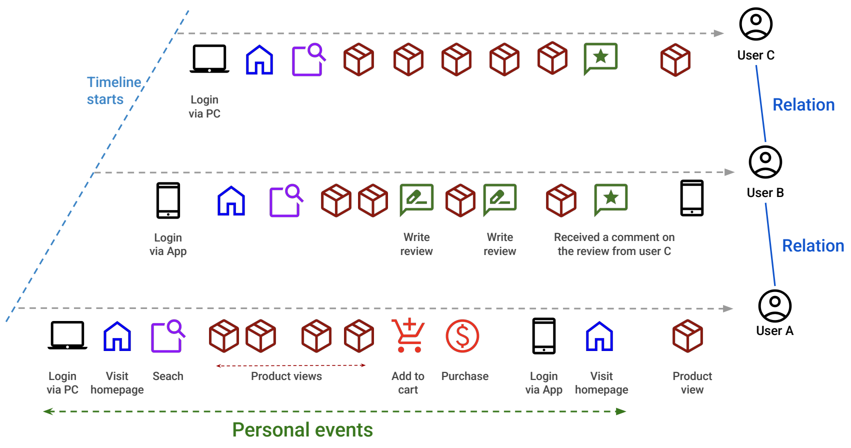

Figure 1: Illustration of data combining sequential and relational structure. Each entity is associated with a sequence of events, while edges capture interactions between entities. For example, in e-commerce, a user generates a sequence of purchases or reviews while also forming relations with other users. Some example relations are friendship, family relation, sharing the same payment method, or sharing the same address, etc.

When dealing with datasets that have both sequential and relational components, existing approaches are less suitable for fully capturing this interplay. On one hand, sequence models such as recurrent networks [6, 7] and Transformers [8, 9] excel at extracting temporal patterns within a sequence but ignore how sequences are connected. On the other hand, graph neural networks (GNNs) [10, 11] and temporal graph models [3, 12] focus on relational structure and evolving interactions, but typically compress each sequence into a single feature vector, discarding fine-grained temporal information. Spatial-temporal methods [13, 14] attempt to combine both, but often rely on the unrealistic assumption of synchronized time steps. As a result, current methods either capture sequential detail without relational context or relational structure without event-level granularity.

To bridge this gap, we propose a new perspective: treating each sequence not as a static node embedding but as a dynamic set of event-level representations that can communicate across the graph. Our approach introduces two complementary components.

- At the *model level*, we design BRIDGE, an end-to-end architecture that integrates a sequential encoder with a GNN, to learn temporal and relational signals jointly.
- At the *layer level*, we develop TOKENXATTN, a token-wise cross-attention mechanism that allows individual events in one sequence to attend to events in neighboring sequences.

The two components together enable richer information exchange and preserve the temporal information of sequences while respecting the graph topology.

Across link prediction and fraud detection experiments, our approach consistently surpasses strong GNN and temporal-graph baselines by treating sequences and graphs as a single system. This joint view unlocks richer temporal–relational reasoning and points toward a new class of hybrid models for complex, multi-modal data.

## 2 Related Works

This section reviews approaches that combine sequences and graphs. We leave additional related works in Appendix A. A growing body of work seeks to leverage sequence and graph information, motivated by applications where entities are connected but also evolve through temporal event sequences. This line of research spans two major directions: (i) spatial-temporal graph methods, which integrate time-series with graph structure, and (ii) temporal graph methods, which model the sequence as edges with timestamps. We refer interested readers to Fathony et al. [1] for an extensive comparison between our problem formulation and existing works.

**Spatial-Temporal Graph Methods** Spatial-temporal graph networks were originally designed for time-series prediction problems such as traffic forecasting. Models like STGCN [15], STGformer [16], MTGCN [17], and MST-GAT [18] encode spatial relations (e.g., road networks) and temporal patterns. In recommendation systems, hybrid approaches have also emerged. Shui et al. [19] models the relational graph among items while simultaneously capturing user consumption sequences. Similarly, Zhang et al. [20] treats users and items as different node types within a heterogeneous graph, enabling message passing between items (sequential) and users (relational). A common limitation of these methods is the assumption that all time series are synchronized and share identical time steps. By contrast, our setting imposes no such constraint: sequences may be asynchronous, vary in length, and even be non-overlapping. This flexibility better reflects real-world scenarios where user interactions occur at irregular intervals and exhibit diverse temporal patterns across different users.

**Temporal Graph Methods** Temporal graph models such as TGN [3], TGAT [4], DyRep [5],TNCN [21], and others [16, 20, 22, 23], attach timestamps to edges to model interactions without aligned time steps. However, a temporal graph requires all events to be described as a relation between two entities/users. Personal events that involve only a single entity cannot be easily represented by a temporal graph. A common workaround is to introduce event nodes (e.g., representing "login" or "payment declined" as nodes) and connect them to the corresponding users, but this can create high-degree hubs (e.g., a single "login" node connected to many users) and is often unnatural in a graph formulation. As argued by Bechler-Speicher et al. [24], forcing an unsuitable data modality into a graph form can fail to encode important information and may not be meaningful.

## 3 Problem Setup

We adopt the PRES formalization for personal and relational user events [1], simplified in that PRES allows time-stamped relational events with possibly multiple events per user pair; we instead assume a single relation type, so each user pair is either related or not. Let $G = (\mathcal{V}, \mathcal{E})$ be a user graph with nodes $\mathcal{V} = \{v_1, \ldots, v_N\}$ and (binary) edges $\mathcal{E} \subseteq \mathcal{V} \times \mathcal{V}$. Each node $v_i$ carries a sequence $S_i = [e_1, \ldots, e_{M_i}]$ of length $M_i$, where each element $e_t$ is a personal event. Although some events may have timestamps, our method only use the ordering of user events, not the exact absolute time, so we omit them from the notation.

## 4 Methodology

To capture both sequential and relational structure in the general setting described above, our approach consists of two main components:

1. **Model level (BRIDGE):** an end-to-end architecture that integrates a sequential model with a graph neural network (GNN).
2. **Layer level (TOKENXATTN):** a token-wise cross-attention message-passing layer that enables tokens in a sequence to attend to tokens in neighboring sequences.

### 4.1 BRIDGE: End-to-End Model Architecture

Our architecture integrates sequential and graph components end-to-end. The sequential module encodes each user's event sequence $S_i$ into a sequence of embedding vectors, $X_i \in \mathbb{R}^{M_i \times d}$, where $M_i$ is the sequence length and $d$ is the embedding dimension. We then pass this to a graph module. As standard GNN layers assume single-vector node features $Z_i \in \mathbb{R}^d$, we compress $X_i$ via pooling mechanism: $Z_i = \text{Compress}(X_i)$, before performing graph message-passing operations among neighboring nodes. Alternatively, our proposed TOKENXATTN (Section 4.2) operates directly on $X_i$, enabling event-level interaction across the graph structure. The graph model's output is added back to the original representation, $Z_i$, via a residual connection.

**Variants.** Algorithm 1 specifies a family of sequence–graph models that we denote BRIDGE. The graph step can be instantiated with standard GNN layers (GCN, GAT, TransformerConv) or with the token-wise cross-attention layer introduced next. We refer to these variants as BRIDGE-GCN, BRIDGE-GAT, BRIDGE-TransformerConv, and BRIDGE-TOKENXATTN.

### 4.2 TOKENXATTN: Token-Level Cross-Attention

In the previous subsection, we mentioned that conventional GNN layers [25–27] assume each node is represented by a single feature vector. However, in our setting, each user corresponds to a sequence of events with inherent temporal structure. Compressing an entire sequence into one vector $Z_i := \text{Compress}(X_i) \in \mathbb{R}^d$, whether by pooling or taking the final event's embedding, inevitably discards the temporal information. We propose keeping the full sequence embedding for message

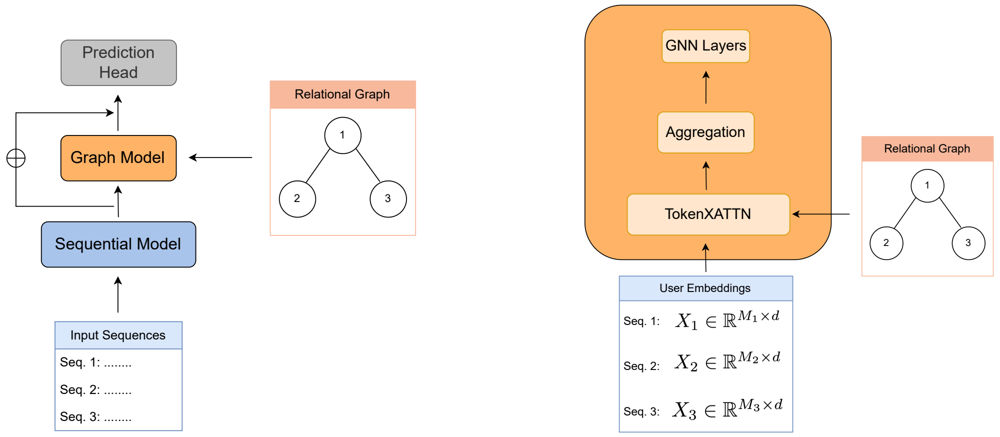

Figure 2: Illustration of our proposed model. *(Left)* BRIDGE, the end-to-end sequence–graph architecture, where the sequential module encodes node sequences and the resulting embeddings serve as node features for the graph module. *(Right)* TOKENXATTN, a zoomed-in view of the token-wise cross-attention layer that enables event-level message passing between neighboring sequences.

**Algorithm 1** BRIDGE

```
Require: Graph $G = (\mathcal{V}, \mathcal{E})$; sequences $\{S_i\}_{i=1}^N$; Number of message passing layers $L$
Ensure: Final node representations $\{h_i\}_{i=1}^N$
 1: for $v_i \in \mathcal{V}$ do
 2:     $X_i \leftarrow \mathrm{SeqEncoder}(S_i)$
 3: end for
 4: for $\ell = 1, \dots, L$ do
 5:     for $v_i \in \mathcal{V}$ do
 6:         $H_i^{(\ell)} \leftarrow \mathrm{MessagePassing}\big(i,\, X_i,\, \{X_j\}_{j \in \mathcal{N}(i)}\big)$        ▷ GCN, GAT, TOKENXATTN, etc.
 7:         $X_i \leftarrow X_i + H_i^{(\ell)}$        ▷ residual
 8:     end for
 9: end for
10: for $v_i \in \mathcal{V}$ do
11:     $h_i \leftarrow \mathrm{vec}(\mathrm{Compress}(X_i))$
12: end for
```

passing, i.e., $Z_i = X_i \in \mathbb{R}^{M_i \times d}$, as different events in a neighbor's history have different effects on the target user (e.g., the most recent action is often most influential). To preserve this granularity, we retain event-level embeddings: each user $i$ is represented by a matrix $X_i \in \mathbb{R}^{M_i \times d}$, where $M_i$ is the sequence length. Building on this representation, we introduce a token-wise cross-attention mechanism (Algorithm 2) that enables token-level message passing.

**General neighbor aggregation.** After computing the per-neighbor messages $H_{i \leftarrow j}$, the final step is to aggregate them into an updated representation for user $i$. We denote this step by a flexible aggregation function $f_{\text{agg}}$, which combines neighbor messages using edge weights $w_{ij}$ and an operator $\mathrm{AGG} \in \{\mathrm{sum}, \mathrm{mean}\}$:

$$H_i \;=\; f_{\text{agg}}\big(\{H_{i\leftarrow j}\}_{j \in \mathcal{N}(i)}\big) \;=\; \mathrm{AGG}_{\,j \in \mathcal{N}(i)}\big(w_{ij}\, H_{i \leftarrow j}\big), \qquad H_i \in \mathbb{R}^{M_i \times d_h}.$$

Different GNN variants define $w_{ij}$ differently. For example, mean aggregation sets $w_{ij} = \frac{1}{|\mathcal{N}(i)|}$, GCN uses $w_{ij} = \frac{1}{\sqrt{\deg(i)\deg(j)}}$, and GAT learns $w_{ij}$ through edge attention mechanisms.

Finally, to capture multi-hop dependencies, we stack standard GNN layers on top of TOKENXATTN, enabling information to propagate beyond immediate neighbors and reach higher-order neighbors.

**Algorithm 2** Token-Level Cross-Attention (TOKENXATTN)

```
Require: Node $i$; token matrix $X_i \in \mathbb{R}^{M_i \times d}$; neighbor tokens $\{X_j\}_{j \in \mathcal{N}(i)}$; weights
    $W_Q, W_K, W_V \in \mathbb{R}^{d \times d_h}$; edge weights $\{w_{ij}\}_{j \in \mathcal{N}(i)}$; $f_{\text{agg}} \in \{\text{sum}, \text{mean}\}$
Ensure: $H_i \in \mathbb{R}^{M_i \times d_h}$
 1: $Q_i \leftarrow X_i W_Q$
 2: for all $j \in \mathcal{N}(i)$ do
 3:     $K_j \leftarrow X_j W_K; \quad V_j \leftarrow X_j W_V$
 4:     $A_{i \leftarrow j} \leftarrow \text{softmax}\big(Q_i K_j^\top / \sqrt{d_h}\big)$        ▷ $\in \mathbb{R}^{M_i \times M_j}$
 5:     $H_{i \leftarrow j} \leftarrow A_{i \leftarrow j} V_j$        ▷ $\in \mathbb{R}^{M_i \times d_h}$
 6: end for
 7: if $f_{\text{agg}} = \text{sum}$ then
 8:     $H_i \leftarrow \sum_{j \in \mathcal{N}(i)} w_{ij} \, H_{i \leftarrow j}$
 9: else        ▷ $f_{\text{agg}} = \text{mean}$
10:     $H_i \leftarrow \frac{1}{|\mathcal{N}(i)|} \sum_{j \in \mathcal{N}(i)} w_{ij} \, H_{i \leftarrow j}$
11: end if
12: return $H_i$
```

## 5 Experiments

We validate our methods on two tasks, friendship prediction and fraud detection. For both experiments, we report the average over three runs.

### 5.1 Friendship Prediction

**Experimental Setup.** We perform friendship prediction on the Brightkite dataset [28, 29] and Amazon reviews [30]. Brightkite is a location-based social networking platform, and the dataset contains location check-ins and friendship relationships among Brightkite users. Each check-in is originally recorded in latitude and longitude, which we converted into Geohash-8 representation [31]. Example geohashes include 9v6kpmr1, gcpwkeq6, and u0yhxgm1, where nearby locations share similar prefixes. Each user has a sequence of geohash check-ins, where we further split each geohash8 into 4 tokens, where the tokens represent increasingly higher resolution (smaller areas) of the location. For Amazon, we use the *Musical*, *Clothing*, and *Electronics* categories. Each user's sequence consists of product IDs and ratings; the "friendship" graph is constructed via co-review ties—two users are connected if they have co-reviewed at least three products.

**Compared Methods.** We compare three categories of baselines with BRIDGE:

1. ***Graph-Only Models.*** GCN [25], GAT [26], and TransformerConv [27] operate on a static user friendship graph using single-vector node representations, ignoring the user sequences.
2. ***Temporal Graph Models.*** TGN [3], DyRep [5], and TNCN [21] are dynamic graph learning methods that model event sequences as timestamped edges. Here, we convert each Geohash8 into a node (we do not perform a tokenized split of the geohash) and connect a user to the geohash node every time they check in. The friendship info does not have a timestamp, so we randomly add a time to it within each user's check-in timeline.
3. ***Two-Stage Training.*** GCN + S, GAT + S, and TransformerConv + S extend the static graphs by incorporating sequence embeddings as node features. These embeddings are obtained from a bidirectional encoder transformer model that we trained with masked language modeling on the user sequence tokens.
4. **BRIDGE.** We stack a BERT sequence encoder with different GNN backbones, trained jointly in an end-to-end manner. In particular, the proposed TOKENXATTN layer achieves the best performance across all metrics. Unlike the Two-Stage Training, BRIDGE does not use frozen embeddings but learns the BERT and GNN components jointly on the target task.

Note that we do not compare against spatio-temporal models because they require synchronized time series with identical time steps across all nodes, which is incompatible with our problem setting.

**Evaluation Metrics.** For the friendship prediction task, we follow prior works [32–34] and use ranking-based metrics: mean reciprocal rank (MRR) and Hits-at-k, for k in {1,3,5,10}. At evaluation time, we rank the ground-truth friendship of each user against 100 negative samples, generated by uniformly sampling users not already connected to the target user in the training set.

Table 1: Link ranking performance across four datasets (higher is better). **H@k** denotes the fraction of test queries for which at least one ground-truth link appears among the top-$k$ predictions. Best results are **bolded** and the runner-ups are underlined.

| | Brightkite | | | | | Amazon–Clothing | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| **Method** | **MRR** | **H@1** | **H@3** | **H@5** | **H@10** | **MRR** | **H@1** | **H@3** | **H@5** | **H@10** |
| ***Graph Only (Static)*** | | | | | | | | | | |
| GCN | 67.4 ±1.9 | 55.6 ±2.4 | 75.5 ±1.8 | 82.3 ±1.3 | 89.2 ±0.7 | 17.3 ±1.5 | 9.1 ±1.7 | 17.4 ±2.0 | 23.3 ±1.9 | 34.1 ±0.7 |
| GAT | 65.4 ±1.6 | 53.6 ±2.0 | 73.0 ±1.4 | 80.1 ±1.0 | 87.8 ±0.6 | 16.2 ±2.3 | 10.3 ±2.5 | 14.3 ±2.7 | 18.0 ±2.1 | 27.5 ±1.6 |
| TransformerConv | 68.5 ±0.2 | 56.9 ±0.2 | 76.6 ±0.4 | 83.2 ±0.4 | 89.8 ±0.3 | 18.8 ±0.7 | 10.5 ±1.1 | 19.2 ±0.3 | 25.2 ±0.3 | 35.5 ±0.4 |
| ***Two-Stage (Static + Sequence)*** | | | | | | | | | | |
| GCN + S | 68.6 ±1.1 | 56.9 ±1.4 | 76.7 ±1.0 | 83.4 ±0.7 | 90.0 ±0.3 | 13.5 ±2.7 | 5.3 ±2.8 | 12.0 ±3.5 | 18.2 ±2.8 | 31.6 ±0.7 |
| GAT + S | 67.9 ±0.5 | 56.4 ±0.6 | 75.7 ±0.6 | 82.3 ±0.5 | 89.4 ±0.2 | 14.1 ±4.6 | 7.3 ±4.8 | 12.5 ±5.5 | 17.2 ±5.6 | 28.8 ±3.6 |
| TransformerConv + S | 72.7 ±0.3 | 61.9 ±0.4 | 80.5 ±0.3 | 86.3 ±0.1 | 91.7 ±0.1 | 17.7 ±1.1 | 8.9 ±1.4 | 17.6 ±1.3 | 24.2 ±1.1 | 35.8 ±1.1 |
| ***Temporal Graphs*** | | | | | | | | | | |
| TGN | 38.7 ±1.6 | 24.8 ±1.8 | 43.8 ±1.7 | 53.9 ±1.3 | 68.1 ±1.1 | 14.8 ±1.4 | 4.9 ±0.8 | 13.4 ±2.1 | 20.8 ±3.0 | 36.9 ±3.4 |
| DyRep | 26.4 ±3.7 | 14.0 ±3.1 | 28.2 ±4.8 | 37.5 ±5.5 | 53.4 ±5.5 | 13.3 ±0.8 | 3.6 ±0.6 | 11.2 ±1.0 | 18.9 ±1.3 | 34.3 ±1.5 |
| TNCN | 53.8 ±0.5 | 42.9 ±0.4 | 58.2 ±0.6 | 65.9 ±1.0 | 76.0 ±1.1 | 31.7 ±1.8 | 22.5 ±1.7 | 33.9 ±2.5 | 39.6 ±2.1 | 48.1 ±1.6 |
| **BRIDGE** | | | | | | | | | | |
| BRIDGE-GCN | 92.2 ±0.6 | 89.9 ±0.7 | 93.6 ±0.6 | 94.8 ±0.5 | 96.3 ±0.4 | 49.5 ±3.4 | 32.6 ±2.9 | 61.1 ±7.3 | 73.9 ±3.2 | **82.3 ±1.1** |
| BRIDGE-GAT | 78.8 ±0.4 | 70.2 ±0.5 | 85.3 ±0.3 | 89.5 ±0.1 | 93.4 ±0.1 | 69.6 ±0.4 | 63.0 ±0.5 | 74.1 ±0.3 | **77.5 ±0.3** | 80.4 ±0.1 |
| BRIDGE-TransformerConv | 78.4 ±0.7 | 69.6 ±1.0 | 84.9 ±0.7 | 89.3 ±0.4 | 93.4 ±0.3 | 43.5 ±0.6 | 30.4 ±0.7 | 44.8 ±1.3 | 58.1 ±3.3 | 82.0 ±0.2 |
| BRIDGE-TOKENXATTN | **92.9 ±0.7** | **90.7 ±0.9** | **94.2 ±0.6** | **95.4 ±0.5** | **96.7 ±0.3** | **70.3 ±1.7** | **64.5 ±2.6** | **74.2 ±1.0** | 77.0 ±0.6 | 79.6 ±0.2 |

| | Amazon–Electronics | | | | | Amazon–Musical | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| **Method** | **MRR** | **H@1** | **H@3** | **H@5** | **H@10** | **MRR** | **H@1** | **H@3** | **H@5** | **H@10** |
| ***Graph Only (Static)*** | | | | | | | | | | |
| GCN | 48.9 ±3.9 | 34.9 ±4.4 | 56.9 ±4.2 | 66.6 ±3.3 | 76.5 ±2.2 | 14.4 ±0.7 | 6.9 ±0.7 | 14.3 ±1.6 | 19.3 ±0.5 | 27.9 ±1.9 |
| GAT | 35.8 ±14.9 | 30.1 ±0.6 | 51.3 ±0.2 | 49.9 ±21.2 | 61.4 ±22.5 | 13.0 ±1.1 | 7.4 ±1.1 | 10.4 ±1.3 | 13.7 ±2.8 | 22.3 ±6.9 |
| TransformerConv | 55.8 ±0.7 | 43.0 ±1.0 | 63.8 ±0.6 | 71.5 ±0.4 | 79.5 ±0.1 | 25.0 ±0.1 | 15.5 ±0.4 | 25.8 ±0.4 | 32.6 ±1.0 | 45.1 ±1.2 |
| ***Two-Stage (Static + Sequence)*** | | | | | | | | | | |
| GCN + S | 50.7 ±0.8 | 37.0 ±0.8 | 58.8 ±1.1 | 68.1 ±0.9 | 77.2 ±0.5 | 16.6 ±4.9 | 9.0 ±4.8 | 15.1 ±5.7 | 20.0 ±5.9 | 32.0 ±5.3 |
| GAT + S | 41.0 ±16.0 | 36.7 ±0.4 | 58.0 ±1.1 | 55.5 ±20.0 | 65.8 ±18.7 | 12.4 ±0.5 | 5.7 ±1.1 | 10.4 ±0.4 | 15.1 ±0.1 | 24.8 ±0.8 |
| TransformerConv + S | 58.4 ±0.2 | 46.2 ±0.3 | 66.2 ±0.1 | 73.5 ±0.1 | 80.8 ±0.0 | 23.6 ±1.9 | 14.1 ±1.7 | 23.4 ±2.4 | 30.2 ±1.7 | 44.6 ±2.9 |
| ***Temporal Graphs*** | | | | | | | | | | |
| TGN | 33.8 ±1.5 | 21.1 ±1.3 | 37.6 ±1.8 | 46.5 ±2.1 | 60.0 ±1.8 | 13.2 ±0.8 | 5.9 ±0.6 | 12.1 ±1.3 | 17.3 ±1.9 | 25.8 ±1.4 |
| DyRep | 17.0 ±1.9 | 7.0 ±1.8 | 16.8 ±2.4 | 23.7 ±2.1 | 37.0 ±2.1 | 11.6 ±0.7 | 4.7 ±1.0 | 9.6 ±0.3 | 14.1 ±1.0 | 24.6 ±0.5 |
| TNCN | 45.4 ±1.1 | 35.7 ±1.2 | 49.8 ±1.1 | 55.8 ±0.9 | 63.0 ±1.2 | 19.8 ±3.0 | 11.1 ±2.5 | 20.3 ±3.5 | 26.2 ±4.8 | 36.7 ±4.5 |
| **BRIDGE** | | | | | | | | | | |
| BRIDGE-GCN | 74.6 ±0.2 | 68.5 ±0.4 | 78.5 ±0.1 | 81.8 ±0.2 | 85.2 ±0.1 | 44.5 ±1.2 | 30.0 ±1.4 | 54.7 ±1.7 | 63.2 ±1.0 | 70.2 ±0.5 |
| BRIDGE-GAT | 68.9 ±0.2 | 60.3 ±0.3 | 75.0 ±0.2 | 79.1 ±0.1 | 82.5 ±0.1 | 48.9 ±1.5 | 37.2 ±1.5 | 54.9 ±2.4 | **65.2 ±2.1** | **72.0 ±0.8** |
| BRIDGE-TransformerConv | 69.6 ±0.3 | 60.4 ±0.4 | 75.9 ±0.2 | 80.6 ±0.2 | 84.8 ±0.2 | 47.3 ±2.1 | 36.9 ±1.9 | 51.2 ±1.6 | 58.4 ±3.3 | 69.4 ±6.2 |
| BRIDGE-TOKENXATTN | **76.6 ±0.4** | **69.3 ±0.5** | **82.1 ±0.3** | **85.5 ±0.1** | **88.3 ±0.1** | **54.7 ±4.1** | **45.7 ±4.5** | **59.9 ±5.7** | 64.4 ±4.9 | 69.2 ±4.0 |

**Results.** From Table 1, we observe a clear hierarchy of performance. Augmenting static graph methods with sequence embeddings as node features (*Two-Stage*) consistently improves their results. Our proposed variants of BRIDGE achieve the best scores across all metrics, substantially outperforming methods in the other three categories. By contrast, temporal graph models do not perform well, likely because encoding geohash as a node and user check-in sequences as time-stamped edges fails to capture fine-grained sequential patterns and can introduce spurious neighbor relations (e.g., treating any two users who check in at the same location as second-degree neighbors). In addition, by converting a Geohash into a node, we lose all the hierarchical information in the Geohash (i.e. first digit of a Geohash conveys larger area information than the second, third, etc.), whereas in our sequence model, we still retain this information by tokenizing a Geohash8 into 4 tokens. Finally, as the friendship does not have time information, we need to incorporate randomized timestamps into the temporal graph model, which may also cause performance degradation.

## 5.2 Fraud Detection

**Experimental Setup.** We evaluate fraud detection on Amazon review data [30]. Each user's sequence comprises product IDs and ratings; for this task, we also encode the associated review text. Each review may receive "helpful" or "unhelpful" votes from other users. Following Dou et al. [35], we use this helpfulness information as a proxy for fraud review incidents. Specifically, we label a user with at least 5 reviews as fraudulent if more than 70% of their reviews are marked unhelpful. Relational edges are defined by co-review patterns: two users are connected if they have co-reviewed at least three products.

**Compared Methods.** Similar to the friendship prediction task, we evaluate static graph and temporal graph methods on the relational graph, using the resulting node embeddings for classification. We further compare against a text-based baseline that uses pretrained review embeddings.

1. ***Graph-Only Models.*** GCN [25], GAT [26], and TransformerConv [27] operate on a static user co-review graph with single-vector node representations, ignoring user sequences.
2. ***Temporal Graph Models.*** We run TGN [3] and DyRep [5]. TNCN [21] is not directly applicable here, as it is designed for link prediction tasks. In this setup, the combination of product and rating is converted to a node. If a user rates the product with a particular rating, we connect the user node to the product rating node with a timestamped edge.
3. ***SBert Seq.*** We encode each review with Sentence-BERT[1] to obtain a sequence of embeddings per user. For classification, we average the embeddings to form a single user representation, which is passed to a classifier.
4. **BRIDGE.** Our approach combines Sentence-BERT embedding of review texts with a learnable rating embedding. The rating embedding and GNN backbone are trained jointly for fraud detection, allowing the model to integrate both textual and relational signals.

**Evaluation Metrics.** For fraud detection, we report two classification metrics: **Max F1** (best F1 across thresholds) and **PR-AUC** (area under the precision–recall curve).

Table 2: Fraud detection performance across three Amazon categories.

| | **Amazon-Movies** | | **Amazon-Electronics** | | **Amazon-Clothing** | |
|---|---|---|---|---|---|---|
| **Method** | **Max F1** | **PR AUC** | **Max F1** | **PR AUC** | **Max F1** | **PR AUC** |
| ***Graph Only (Static)*** | | | | | | |
| GCN | 32.4 ±2.2 | 19.4 ±2.0 | 12.0 ±0.0 | 6.4 ±0.2 | 4.9 ±0.1 | 2.4 ±0.3 |
| GAT | 33.5 ±0.1 | 20.2 ±0.5 | 12.0 ±0.1 | 6.5 ±0.4 | 4.6 ±0.1 | 2.2 ±0.0 |
| TransformerConv | 33.6 ±0.1 | 20.3 ±1.1 | 12.0 ±0.2 | 6.4 ±0.2 | 4.6 ±0.2 | 2.2 ±0.1 |
| ***Temporal Graphs*** | | | | | | |
| TGN | 44.9 ±0.4 | 34.6 ±0.3 | 14.5 ±0.9 | 9.0 ±0.6 | 8.0 ±1.6 | 3.7 ±0.6 |
| DyRep | 44.1 ±0.5 | 31.2 ±0.2 | 12.4 ±0.5 | 7.0 ±0.2 | 5.4 ±0.5 | 2.7 ±0.4 |
| ***Sequence Only (Review Text Embedding)*** | | | | | | |
| SBert | 75.6 ±0.7 | 83.1 ±1.0 | 42.2 ±1.3 | 38.7 ±1.9 | 34.2 ±3.2 | 31.9 ±3.1 |
| **BRIDGE** | | | | | | |
| BRIDGE-GCN | 78.8 ±0.9 | 85.6 ±0.6 | 45.2 ±1.9 | 43.4 ±2.1 | 36.0 ±1.7 | 34.8 ±2.6 |
| BRIDGE-GAT | **80.1 ±0.8** | 86.9 ±0.4 | 46.2 ±1.3 | 45.2 ±2.5 | **38.4 ±1.2** | **36.3 ±1.6** |
| BRIDGE-TransformerConv | 79.8 ±0.9 | 86.9 ±1.0 | 47.9 ±2.6 | 46.6 ±4.0 | 37.0 ±1.7 | 35.4 ±2.6 |
| BRIDGE-TOKENXATTN | 80.0 ±1.0 | **87.7 ±0.8** | **49.2 ±2.1** | **51.6 ±3.4** | 36.1 ±1.8 | 34.4 ±3.2 |

**Results.** From Table 2, a consistent ordering emerges: graph-only < temporal graph < sequence-only (SBERT) < BRIDGE. Temporal structure helps over static graphs but still trails a simple SBERT average, highlighting the importance of textual information for this task. All BRIDGE variants outperform the baselines, showing clear gains from coupling event sequences with graph context.

## 6 Conclusion

We presented BRIDGE, a unified end-to-end architecture that learns from sequences and graphs jointly, eliminating the common handoff where sequence encoders are frozen and passed to a separate GNN. In BRIDGE, the sequential and relational components are trained together under a single objective, allowing gradients to flow across both parts of the model and yielding task-aligned representations. This simple design choice, treating sequence and graph modeling as one training problem, consistently outperforms static GNNs, temporal graph models, and sequence-only baselines across friendship prediction and fraud detection, demonstrating its effectiveness. We further proposed TOKENXATTN, a token-level cross-attention layer that enables interaction between events across neighboring sequences. Rather than viewing sequences and graphs as separate problems, we argue they are complementary components of the same dataset—BRIDGE unifies them under a single architecture, with TOKENXATTN enabling event-level interactions across them. Looking forward, we believe this perspective opens the door to new models that can seamlessly integrate multimodal sequential and relational signals across a wide range of domains, from social networks to recommendation systems and beyond.

[1]`https://huggingface.co/sentence-transformers/all-MiniLM-L6-v2`

## References

[1] Rizal Fathony, Igor Melnyk, Owen Reinert, Nam H. Nguyen, Daniele Rosa, and C. Bayan Bruss. Integrating Sequential and Relational Modeling for User Events: Datasets and Prediction Tasks, October 2025. URL `http://arxiv.org/abs/2510.11903`. arXiv:2510.11903 [cs].

[2] Giang Hoang Nguyen, John Boaz Lee, Ryan A Rossi, Nesreen K Ahmed, Eunyee Koh, and Sungchul Kim. Continuous-time dynamic network embeddings. In *Companion Proceedings of the The Web Conference 2018*, pages 969–976, 2018.

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

[4] Da Xu, Chuanwei Ruan, Evren Korpeoglu, Sushant Kumar, and Kannan Achan. Inductive Representation Learning on Temporal Graphs, February 2020. URL `http://arxiv.org/abs/2002.07962`. arXiv:2002.07962 [cs].

[5] Rakshit Trivedi, Mehrdad Farajtabar, Prasenjeet Biswal, and Hongyuan Zha. DyRep: Learning Representations over Dynamic Graphs. September 2018. URL `https://openreview.net/forum?id=HyePrhR5KX`.

[6] Alex Sherstinsky. Fundamentals of Recurrent Neural Network (RNN) and Long Short-Term Memory (LSTM) network. *Physica D: Nonlinear Phenomena*, 404:132306, March 2020. ISSN 0167-2789. doi: 10.1016/j.physd.2019.132306. URL `https://www.sciencedirect.com/science/article/pii/S0167278919305974`.

[7] Sepp Hochreiter and Jürgen Schmidhuber. Long Short-Term Memory. *Neural Computation*, 9 (8):1735–1780, November 1997. ISSN 0899-7667. doi: 10.1162/neco.1997.9.8.1735. URL `https://ieeexplore.ieee.org/abstract/document/6795963`.

[8] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is All you Need. In *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc., 2017. URL `https://proceedings.neurips.cc/paper_files/paper/2017/hash/3f5ee243547dee91fbd053c1c4a845aa-Abstract.html`.

[9] Bryan Lim, Sercan Ö. Arık, Nicolas Loeff, and Tomas Pfister. Temporal Fusion Transformers for interpretable multi-horizon time series forecasting. *International Journal of Forecasting*, 37 (4):1748–1764, October 2021. ISSN 0169-2070. doi: 10.1016/j.ijforecast.2021.03.012. URL `https://www.sciencedirect.com/science/article/pii/S0169207021000637`.

[10] Jie Zhou, Ganqu Cui, Shengding Hu, Zhengyan Zhang, Cheng Yang, Zhiyuan Liu, Lifeng Wang, Changcheng Li, and Maosong Sun. Graph Neural Networks: A Review of Methods and Applications, October 2021. URL `http://arxiv.org/abs/1812.08434`. arXiv:1812.08434 [cs].

[11] Yu Huang, Min Zhou, Menglin Yang, Zhen Wang, Muhan Zhang, Jie Wang, Hong Xie, Hao Wang, Defu Lian, and Enhong Chen. Foundations and Frontiers of Graph Learning Theory, July 2024. URL `http://arxiv.org/abs/2407.03125`. arXiv:2407.03125 [cs].

[12] Seyed Mehran Kazemi, Rishab Goel, Kshitij Jain, Ivan Kobyzev, Akshay Sethi, Peter Forsyth, and Pascal Poupart. Representation Learning for Dynamic Graphs: A Survey, April 2020. URL `http://arxiv.org/abs/1905.11485`. arXiv:1905.11485 [cs].

[13] Senzhang Wang, Jiannong Cao, and Philip S. Yu. Deep Learning for Spatio-Temporal Data Mining: A Survey, June 2019. URL `http://arxiv.org/abs/1906.04928`. arXiv:1906.04928 [cs].

[14] Yaguang Li, Rose Yu, Cyrus Shahabi, and Yan Liu. Diffusion Convolutional Recurrent Neural Network: Data-Driven Traffic Forecasting, February 2018. URL `http://arxiv.org/abs/1707.01926`. arXiv:1707.01926 [cs].

[15] Bing Yu, Haoteng Yin, and Zhanxing Zhu. Spatio-Temporal Graph Convolutional Networks: A Deep Learning Framework for Traffic Forecasting. In *Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence*, pages 3634–3640, July 2018. doi: 10.24963/ijcai.2018/505. URL `http://arxiv.org/abs/1709.04875`. arXiv:1709.04875 [cs].

[16] Hongjun Wang, Jiyuan Chen, Tong Pan, Zheng Dong, Lingyu Zhang, Renhe Jiang, and Xuan Song. STGformer: Efficient Spatiotemporal Graph Transformer for Traffic Forecasting, October 2024. URL `http://arxiv.org/abs/2410.00385`. arXiv:2410.00385 [cs] version: 1.

[17] Zonghan Wu, Shirui Pan, Guodong Long, Jing Jiang, Xiaojun Chang, and Chengqi Zhang. Connecting the Dots: Multivariate Time Series Forecasting with Graph Neural Networks. In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, KDD '20, pages 753–763, New York, NY, USA, August 2020. Association for Computing Machinery. ISBN 978-1-4503-7998-4. doi: 10.1145/3394486.3403118. URL `https://dl.acm.org/doi/10.1145/3394486.3403118`.

[18] Chaoyue Ding, Shiliang Sun, and Jing Zhao. MST-GAT: A Multimodal Spatial-Temporal Graph Attention Network for Time Series Anomaly Detection. *Information Fusion*, 89:527–536, January 2023. ISSN 15662535. doi: 10.1016/j.inffus.2022.08.011. URL `http://arxiv.org/abs/2310.11169`. arXiv:2310.11169 [cs].

[19] Zeren Shui, Ge Liu, Anoop Deoras, and George Karypis. Sequence-graph duality: Unifying user modeling with self-attention for sequential recommendation, 2022. URL `https://www.amazon.science/publications/sequence-graph-duality-unifying-user-modeling-with-self-attention-for-sequential-recommendat`

[20] Mengqi Zhang, Shu Wu, Xueli Yu, Qiang Liu, and Liang Wang. Dynamic Graph Neural Networks for Sequential Recommendation. *IEEE Transactions on Knowledge and Data Engineering*, 35(05):4741–4753, May 2023. ISSN 1041-4347. doi: 10.1109/TKDE.2022.3151618. URL `https://www.computer.org/csdl/journal/tk/2023/05/09714053/1B0XPB9Fgk0`. Publisher: IEEE Computer Society.

[21] Xiaohui Zhang, Yanbo Wang, Xiyuan Wang, and Muhan Zhang. Efficient neural common neighbor for temporal graph link prediction, 2024. URL `https://arxiv.org/abs/2406.07926`.

[22] Sangwoo Seo, Sungwon Kim, Jihyeong Jung, Yoonho Lee, and Chanyoung Park. Self-Explainable Temporal Graph Networks based on Graph Information Bottleneck, June 2024. URL `http://arxiv.org/abs/2406.13214`. arXiv:2406.13214 [cs].

[23] Lisi Qarkaxhija, Vincenzo Perri, and Ingo Scholtes. De Bruijn goes Neural: Causality-Aware Graph Neural Networks for Time Series Data on Dynamic Graphs, September 2022. URL `http://arxiv.org/abs/2209.08311`. arXiv:2209.08311 [cs].

[24] Maya Bechler-Speicher, Ben Finkelshtein, Fabrizio Frasca, Luis Müller, Jan Tönshoff, Antoine Siraudin, Viktor Zaverkin, Michael M. Bronstein, Mathias Niepert, Bryan Perozzi, Mikhail Galkin, and Christopher Morris. Position: Graph learning will lose relevance due to poor benchmarks, 2025. URL `https://arxiv.org/abs/2502.14546`.

[25] Thomas N. Kipf and Max Welling. Semi-supervised classification with graph convolutional networks, 2017. URL `https://arxiv.org/abs/1609.02907`.

[26] Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Liò, and Yoshua Bengio. Graph attention networks, 2018. URL `https://arxiv.org/abs/1710.10903`.

[27] Yunsheng Shi, Zhengjie Huang, Shikun Feng, Hui Zhong, Wenjin Wang, and Yu Sun. Masked label prediction: Unified message passing model for semi-supervised classification, 2021. URL `https://arxiv.org/abs/2009.03509`.

[28] Eunjoon Cho, Seth A. Myers, and Jure Leskovec. Friendship and mobility: user movement in location-based social networks. In *Proceedings of the 17th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '11, page 1082–1090, New

York, NY, USA, 2011. Association for Computing Machinery. ISBN 9781450308137. doi: 10.1145/2020408.2020579. URL https://doi.org/10.1145/2020408.2020579.

[29] Jure Leskovec and Andrej Krevl. SNAP Datasets: Stanford large network dataset collection. http://snap.stanford.edu/data, June 2014.

[30] Julian McAuley, Christopher Targett, Qinfeng Shi, and Anton van den Hengel. Image-based recommendations on styles and substitutes. In *Proceedings of the 38th International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '15, page 43–52, New York, NY, USA, 2015. Association for Computing Machinery. ISBN 9781450336215. doi: 10.1145/2766462.2767755. URL https://doi.org/10.1145/2766462.2767755.

[31] Gustavo Niemeyer. Geohash, 2008.

[32] Julia Gastinger, Shenyang Huang, Mikhail Galkin, Erfan Loghmani, Ali Parviz, Farimah Poursafaei, Jacob Danovitch, Emanuele Rossi, Ioannis Koutis, Heiner Stuckenschmidt, Reihaneh Rabbany, and Guillaume Rabusseau. Tgb 2.0: A benchmark for learning on temporal knowledge graphs and heterogeneous graphs, 2024. URL https://arxiv.org/abs/2406.09639.

[33] Shenyang Huang, Farimah Poursafaei, Jacob Danovitch, Matthias Fey, Weihua Hu, Emanuele Rossi, Jure Leskovec, Michael Bronstein, Guillaume Rabusseau, and Reihaneh Rabbany. Temporal graph benchmark for machine learning on temporal graphs, 2023. URL https://arxiv.org/abs/2307.01026.

[34] Lu Yi, Jie Peng, Yanping Zheng, Fengran Mo, Zhewei Wei, Yuhang Ye, Yue Zixuan, and Zengfeng Huang. Tgb-seq benchmark: Challenging temporal gnns with complex sequential dynamics, 2025. URL https://arxiv.org/abs/2502.02975.

[35] Yingtong Dou, Zhiwei Liu, Li Sun, Yutong Deng, Hao Peng, and Philip S Yu. Enhancing graph neural network-based fraud detectors against camouflaged fraudsters. In *Proceedings of the 29th ACM international conference on information & knowledge management*, pages 315–324, 2020.

[36] Zihang Dai, Zhilin Yang, Yiming Yang, Jaime Carbonell, Quoc V. Le, and Ruslan Salakhutdinov. Transformer-xl: Attentive language models beyond a fixed-length context, 2019. URL https://arxiv.org/abs/1901.02860.

[37] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding, May 2019. URL http://arxiv.org/abs/1810.04805. arXiv:1810.04805 [cs].

[38] Sébastien Bubeck, Varun Chandrasekaran, Ronen Eldan, Johannes Gehrke, Eric Horvitz, Ece Kamar, Peter Lee, Yin Tat Lee, Yuanzhi Li, Scott Lundberg, Harsha Nori, Hamid Palangi, Marco Tulio Ribeiro, and Yi Zhang. Sparks of artificial general intelligence: Early experiments with gpt-4, 2023. URL https://arxiv.org/abs/2303.12712.

[39] Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Liò, and Yoshua Bengio. Graph Attention Networks, February 2018. URL http://arxiv.org/abs/1710.10903. arXiv:1710.10903 [stat].

[40] William L. Hamilton, Rex Ying, and Jure Leskovec. Inductive representation learning on large graphs, 2018. URL https://arxiv.org/abs/1706.02216.

[41] Thomas N. Kipf and Max Welling. Variational graph auto-encoders, 2016. URL https://arxiv.org/abs/1611.07308.

[42] Muhan Zhang and Yixin Chen. Link prediction based on graph neural networks, 2018. URL https://arxiv.org/abs/1802.09691.

[43] Rex Ying, Ruining He, Kaifeng Chen, Pong Eksombatchai, William L. Hamilton, and Jure Leskovec. Graph convolutional neural networks for web-scale recommender systems. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, KDD '18, page 974–983. ACM, July 2018. doi: 10.1145/3219819.3219890. URL http://dx.doi.org/10.1145/3219819.3219890.

[44] Xiang Wang, Xiangnan He, Yixin Cao, Meng Liu, and Tat-Seng Chua. Kgat: Knowledge graph attention network for recommendation. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, KDD '19, page 950–958. ACM, July 2019. doi: 10.1145/3292500.3330989. URL `http://dx.doi.org/10.1145/3292500.3330989`.

[45] Shiwen Wu, Fei Sun, Wentao Zhang, Xu Xie, and Bin Cui. Graph neural networks in recommender systems: A survey, 2022. URL `https://arxiv.org/abs/2011.02260`.

[46] Zhikai Chen, Haitao Mao, Hang Li, Wei Jin, Haifang Wen, Xiaochi Wei, Shuaiqiang Wang, Dawei Yin, Wenqi Fan, Hui Liu, and Jiliang Tang. Exploring the potential of large language models (llms)in learning on graphs. *ACM SIGKDD Explorations Newsletter*, 25:42 – 61, 2023. URL `https://api.semanticscholar.org/CorpusId:259375824`.

[47] Bryan Perozzi, Bahare Fatemi, Dustin Zelle, Anton Tsitsulin, Mehran Kazemi, Rami Al-Rfou, and Jonathan J. Halcrow. Let your graph do the talking: Encoding structured data for llms. *ArXiv*, abs/2402.05862, 2024. URL `https://api.semanticscholar.org/CorpusId:267547812`.

[48] Yijun Tian, Huan Song, Zichen Wang, Haozhu Wang, Ziqing Hu, Fang Wang, N. Chawla, and Panpan Xu. Graph neural prompting with large language models. In *AAAI Conference on Artificial Intelligence*, 2023. URL `https://arxiv.org/pdf/2309.15427.pdf`.

[49] Yuxiang Wang, Xinnan Dai, Wenqi Fan, and Yao Ma. Exploring graph tasks with pure llms: A comprehensive benchmark and investigation. *ArXiv*, abs/2502.18771, 2025. URL `https://api.semanticscholar.org/CorpusId:276617594`.

[50] Ruosong Ye, Caiqi Zhang, Runhui Wang, Shuyuan Xu, and Yongfeng Zhang. Language is all a graph needs. In *Findings*, 2023. URL `https://api.semanticscholar.org/CorpusId:260887732`.

[51] Zirui Guo, Lianghao Xia, Yanhua Yu, Yuling Wang, Zixuan Yang, Wei Wei, Liang Pang, Tat-Seng Chua, and Chao Huang. Graphedit: Large language models for graph structure learning. *ArXiv*, abs/2402.15183, 2024. URL `https://api.semanticscholar.org/CorpusId:267897533`.

[52] Yanchao Tan, Hang Lv, Pengxiang Zhan, Shiping Wang, and Carl Yang. Graph-oriented instruction tuning of large language models for generic graph mining. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2024. URL `https://api.semanticscholar.org/CorpusId:268296710`.

[53] Xubin Ren, Jiabin Tang, Dawei Yin, Nitesh V. Chawla, and Chao Huang. A survey of large language models for graphs. *Proceedings of the 30th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, 2024. URL `https://api.semanticscholar.org/CorpusId:269761536`.

[54] Chao Huang, Xubin Ren, Jiabin Tang, Dawei Yin, and Nitesh V. Chawla. Large language models for graphs: Progresses and directions. *Companion Proceedings of the ACM Web Conference 2024*, 2024. URL `http://dl.acm.org/citation.cfm?id=3641251`.

[55] Jiajin Liu, Dongzhe Fan, Jiacheng Shen, Chuanhao Ji, D. Zha, and Qiaoyu Tan. Graph-mllm: Harnessing multimodal large language models for multimodal graph learning. *ArXiv*, abs/2506.10282, 2025. URL `https://api.semanticscholar.org/CorpusId:279318592`.

[56] Huachi Zhou, Jiahe Du, Chuang Zhou, Chang Yang, Yilin Xiao, Yuxuan Xie, and Xiao Huang. Each graph is a new language: Graph learning with llms. In *Annual Meeting of the Association for Computational Linguistics*, 2025. URL `https://api.semanticscholar.org/CorpusId:275757427`.

## A Additional Related Works

### A.1 Sequence Modeling

Transformer-based sequence models have delivered state-of-the-art performance across a wide range of language and vision tasks by learning contextual representations over token sequences [8, 36–38]. Given an input sequence, these models map each token to a continuous embedding and use self-attention to capture local and long-range dependencies within the sequence. While highly effective for modeling the internal structure of a single sequence, this paradigm typically treats each sequence in isolation and does not leverage information from *related* sequences.

### A.2 Graph Modeling

Graph models capture relational structures among entities (nodes). In particular, graph neural networks (GNNs) propagate and aggregate information over local neighborhoods to learn node, edge, or graph level embeddings [25, 27, 39]. These methods have demonstrated strong performance across tasks such as node classification[25, 40], link prediction [41, 42], and recommendation [43–45]. Standard GNNs model each node in the graph as a single feature vector and do not account for sequential information within nodes. Bridging graph-based relational modeling with sequence-aware representations is therefore an important step toward richer models that capture both intra-sequence patterns and inter-sequence relations.

### A.3 Graph Large Language Models

An emerging line of research seeks to leverage the generalization capabilities of foundation models for graph learning. *Graph large language models (Graph LLMs)* achieve this by integrating pretrained LLMs with graph-structured data through prompting [46–48], instruction tuning [49–52], or hybrid architectures [53–55]. The promise of Graph LLMs lies in bridging reasoning over relational data with the flexibility of natural language interfaces. In line with this view, Zhou et al. [56] even conceptualizes that each graph can be seen as a new language for the LLMs to learn. Our approach, however, differs: rather than adapting pretrained LLMs to graphs, we introduce a unified architecture that integrates sequence modeling with GNNs to jointly capture temporal dynamics and relational structure.

## B Limitations and Discussion

While BRIDGE provides a unified way to learn from sequences and graphs, several limitations remain.

**Use of timestamps.** BRIDGE preserves event order but does not inject absolute timestamps or inter–event gaps. Likewise, TOKENXATTN attends over tokens without explicit time encoding. Tasks where absolute time or time interval between events are informative may benefit from adding relative or absolute time features, which we leave to future work.

**Complexity of TOKENXATTN.** For preserving the granularity of event-level information, the complexity of TOKENXATTN is quadratic with respect to the sequence length. Some potential efficiency improvements can be applying a more efficient attention mechanism, such as linear attention, or reducing the sequence length through patching.