Integrating Sequential and Relational Modeling for User Events: Datasets and Prediction Tasks

Anonymous Author(s)
Affiliation
Address
email

Abstract

User event modeling plays a central role in many machine learning applications, with use cases spanning e-commerce, social media, finance, cybersecurity, and other domains. User events can be broadly categorized into personal events, which involve individual actions, and relational events, which involve interactions between two users. These two types of events are typically modeled separately, using sequence-based methods for personal events and graph-based methods for relational events. Despite the need to capture both event types in real-world systems, prior work has rarely considered them together. This is often due to the convenient simplification that user behavior can be adequately represented by a single formalization, either as a sequence or a graph. To address this gap, there is a need for public datasets and prediction tasks that explicitly incorporate both personal and relational events. In this work, we introduce a collection of such datasets, propose a unified formalization, and empirically show that models benefit from incorporating both event types. Our results also indicate that current methods leave a notable room for improvements. We release these resources to support further research in unified user event modeling and encourage progress in this direction.

1 Introduction

Modeling user events is a central task in machine learning with broad applications across various domains [1–3]. In e-commerce, it is used to capture user preferences for personalized ranking and product recommendation [4, 5]. In social media platforms, event modeling supports feed optimization and engagement prediction by inferring user interests over time [6–8]. Financial systems leverage user behavior data for fraud detection, credit risk assessment, and behavioral profiling [9–12]. Online services such as search and streaming platforms rely on user event sequences for content recommendation under real-time constraints [13–16]. In cybersecurity, modeling user and system events is essential for detecting anomalies and preventing intrusions [17, 18]. These applications demonstrate the importance of building models that can effectively capture complex, context-dependent user behavior from event sequences.

User events can be broadly categorized into personal and relational events. Personal events involve only a single user and reflect individual actions, such as searching for content, viewing items, or posting updates. In contrast, relational events involve interactions between two or more users, such as following another user, co-editing a document, or exchanging messages. Traditionally, these two types of events are often modeled separately. Relational events are commonly modeled using graph-based approaches that capture structural dependencies and interaction patterns among users [19–22]. On the other hand, personal events are typically modeled as sequences using recurrent or attention-based architectures to capture temporal dependencies in personal event histories [23–29].

Submitted to 39th Conference on Neural Information Processing Systems (NeurIPS 2025). Do not distribute.

There have been efforts in the graph area to capture both structural and temporal dependencies using temporal graph formalizations (such as CTDG [30]) and models built on top of these formalizations (such as TGAT [31], TGN [32], and DyRep [33]). However, these approaches primarily focus on the temporal dependencies of relational events while neglecting personal events. For example, the formalization used in the Temporal Graph Benchmark (TGB) papers [34, 35] defines a temporal graph as a stream of triplets consisting of source, destination, and timestamp. Personal events that involve only a single entity cannot be directly represented under this formulation. One workaround is to convert all personal events into nodes and define personal events as triplets of user node, event node, and timestamp. However, this construction is not as straightforward for capturing temporal dependencies in personal event histories compared to sequence-based modeling.

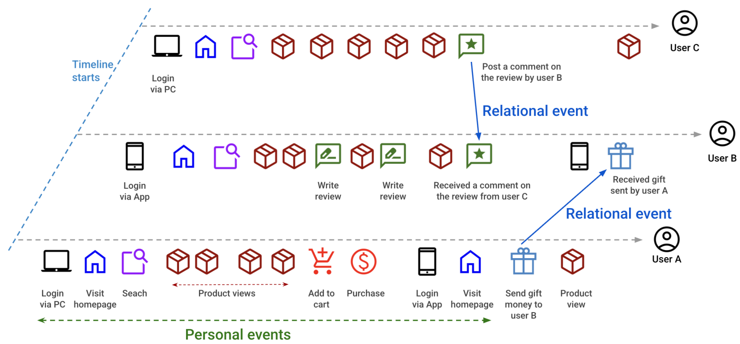


Figure 1: An illustration of personal and relational events in e-commerce. Personal events involve a single user, such as login, search, view, or purchase. Relational events involve interaction between two users, such as sending a gift or commenting on another user's review.

Going back to the personal and relational event category, in many application domains, the number of personal events is typically much larger than that of relational events. For example, in e-commerce platforms, as illustrated in Figure 1, users often view products, search for items, or add products to their cart, whereas relational interactions, such as referrals, sending gifts, or socially engaged reviews, are less frequent. In financial systems, customers routinely perform account queries, check balances, or initiate transactions, while peer-to-peer interactions such as money transfers or joint account actions are relatively infrequent. In cybersecurity systems, personal events may include actions like logging in, accessing files, or executing processes, while relational events, such as remote connections to other users, or file sharing between users, occur less frequently. Despite their higher volume, personal events are often underrepresented in existing graph-based formulations, which tend to prioritize relational structure. In practice, however, both personal and relational events carry complementary signals, and many predictive tasks, such as item recommendation, fraud detection, customer profiling, and behavior forecasting, benefit from capturing both types of information.

Even though there is a need to capture both personal and relational events in many application domains, prior work has rarely considered them together. Practitioners often simplify the complexity of user event modeling by adopting either a graph or a sequence formalization, as most machine learning models are developed within one of these frameworks. As a result, one type of event—typically the less convenient to represent—is often ignored entirely, leading to an incomplete view of user behavior. To build a more comprehensive understanding of user event modeling, there is a need for public datasets and benchmark tasks that explicitly incorporate both event types. Such resources would provide a foundation for developing and evaluating models that integrate these complementary signals.

Summary of Contributions. In this work, we aim to support the study of user event modeling that incorporates both personal and relational events. Our contributions are as follows:

- We curate, pre-process, and release a collection of public datasets and prediction tasks that explicitly include both personal and relational events.
- We introduce a new formalization for user event modeling that captures both personal and relational events.
- We empirically demonstrate that incorporating both personal and relational events improves performance on a range of prediction tasks.
- We show that existing models, originally developed for either sequential or relational data, are less well suited for this event modeling setting, leaving room for future improvements.
- We invite the research community to use these resources and help close the gap in unified user event modeling.

2 Related Works

Event sequence. Event sequence modeling is a broad topic that covers many different domains which share a similar goal of predicting future events from past histories. Temporal point processes (TPPs), such as the Poisson and Hawkes processes [36], model discrete events in continuous time using intensity functions. Neural extensions [37–39] incorporate RNNs or attention for more accurate timestamp prediction and are applied in finance, healthcare, and user modeling. However, TPPs often assume simple event structures and focus only on timing, which limits their ability to capture dependencies across users or networks.

Sequential recommendation. A closely related application domain is sequential recommendation, where the goal is to predict the next item a user will interact with based on their history. Early methods used Markov chains or matrix factorization on time-slided data [40, 41], while recent models such as GRU4Rec [42], SASRec [43], and BERT4Rec [26] apply deep sequence encoders. These models capture user preferences over time but typically treat users independently, without modeling user-to-user interactions.

Graph models. In parallel, graph-based models have advanced user interaction modeling, especially through GNNs. While static graphs lack temporal order, time-aware constructions such as time-windowed graphs have been used to encode the the dynamics [44], enabling tasks such as link prediction on constructed event graphs. GCN [45] introduced neighborhood aggregation, GraphSAGE [46] enabled inductive learning through sampling. GAT [47] added attention mechanisms, and HGT [48] extended GNNs to heterogeneous graphs. GNNs remain widely used for personal event modeling [19].

Temporal graph. Temporal graph methods fall into two main categories: discrete-time and continuous-time [49, 35]. Discrete-time methods support both homogeneous [50] and heterogeneous data [51–53]. Continuous-time methods preserve finer temporal detail and can be used to model event sequences as timestamped edges [54]. TGN [32] generalizes this setting and includes DyRep [33] as a special case. HTGN-BTW [55] and STHN [56] extend TGN to heterogeneous graphs. Beyond it, several methods have also been proposed for modeling temporal knowledge graphs [57–59].

Benchmark datasets. Benchmarks have been proposed across related areas. Temporal graph benchmarks include TGB [34], its heterogeneous and knowledge graph extension TGB 2.0 [35], and TGB-Seq [60], which adds a more complex sequence of edge dynamics. For static graphs, OGB [61] and OGB-LSC [62] are widely used. In recommendation, large-scale interaction benchmarks include MIND [63], TenRec [64], NineRec [65], and BARS [66]. For event sequences and temporal point processes, recent efforts include EBES [67], EasyTPP [68], and HOTPP [69].

Other research on graph and sequence. Several studies have explored different settings involving temporal and structural dynamics. Some models combine graph and time series data using spatio-temporal graphs [70–73]. Others merge the outputs of graph and sequence models in various application domains [74–76]. Recent works tokenize graphs and applies transformers or state space models (SSMs) for graph learning [77–83]. Additional efforts incorporate knowledge graphs into language models [84–86] and apply graph-augmented retrieval in text generation tasks [87, 88].

3 Problem Formalization

Notations. In our **P**ersonal and **R**elational User **E**vent **S**equence (PRES) modeling, we have a collection of event sequences, each representing the events that happen to a particular user (which can also be a customer, account, etc.). We denote the set of users as $\mathcal{U} = \{u_1, u_2, \cdots, u_N\}$, where N is the number of users. Each user has their own sequence of events that occur over time. For example, the sequence for user u_i is denoted as $\mathrm{Seq}(u_i) = [(e_1, t_1), (e_2, t_2), \cdots, (e_{M_i}, t_{M_i})]$, where e describes an event, t describes the time at which the event occurs, and M_i denotes the number of events for user u_i. Each user may have a different number of events in their event sequence. We denote the set of all user sequences by $\mathcal{S} = \{\mathrm{Seq}(u) \mid u \in \mathcal{U}\}$.

An event may come from two different event sets: the *personal event* set and the *relational event* set. The personal event set contains a set of events that can occur for an individual user; $p \in \mathcal{P} \triangleq \{1, 2, \cdots, |\mathcal{P}|\}$. The relational event set contains a set of all possible events $r \in \mathcal{R} \triangleq \{1, 2, \cdots, |\mathcal{R}|\}$, which involve a relation from one user to another. Thus, an event can be defined by a personal event $e = p$, or a relational event tuple $e = (r, v)$, where v is another user.

Table 1: Dataset Statistics

Properties	`brightkite`	`gowalla`	`az-clothing`	`az-electronics`	`github`
Personal Events Relational Events	check-in friendship	check-in friendship	product rating co-review	product rating co-review	github activity collaboration
# Users	58,228	196,591	185,986	254,064	3,669,079
# Events	5,130,866	8,342,943	1,591,947	2,938,178	102,878,895
# Personal Events	4,702,710	6,442,289	1,573,869	2,281,128	95,974,149
# Relational Events	428,156	1,900,654	18,078	657,050	6,904,746
# Unique Events	628,519	1,169,154	846,052	529,198	24
# Unique Timestamps	4,506,822	5,561,957	3,464	5,373	2,675,990
# Users w. pers. events	51,406	107,092	185,986	254,064	3,669,079
# Users w. rel. events	58,228	196,591	5,017	49,852	441,958
# Users w. both events	51,406	107,092	5,017	49,852	441,958

Difference from other well-known formalizations. Our formulation differs from graph-based representations in several ways. Static graphs aggregate interactions into a single structure, discarding temporal information. Temporal graphs introduce dynamic edges but focus on global structural changes rather than user-specific event sequences. In both cases, personal events are often omitted or encoded as nodes, limiting representational flexibility. In contrast, we model user-wise event sequences with preserved temporal order, explicitly capturing both personal and relational events. Our formulation also supports richer event representations, including decomposing events into sub-events, as shown in our experiments.

The PRES formulation also differs from the standard sequence-based approaches. Event sequence models typically treat user actions as flat sequences, without modeling interactions between users. Sequential recommendation focuses on item sequences per user and does not account for user-to-user interactions, while our formulation supports more flexible personal event representation, including decomposed sub-events, and explicitly models relational events. Temporal point process models capture event timing and types but are less suited for rich semantics or relational structure. In contrast, our formulation models both personal and relational events with their content and temporal order.

4 Datasets and Prediction Tasks

4.1 Dataset Information

We curated user event datasets from multiple domains and processed each according to our formalization in Section 3. The data is stored in CSV format with the columns: `uid`, `timestamp`, `event_set`, `event`, and `other_uid` (See Appendix B for details). The `uid` is a numerical user ID, whereas `event_set` indicates whether the event is *personal* or *relational*. For relational events, `other_uid` refers to the other user involved in the relation; for personal events, this column is null.

Dataset description. Here we describe each dataset in detail. Table 1 provides general statistics of each dataset. More details on collection, processing, and dataset license are available in Appendix A

`pres-brightkite`. This dataset contains location check-ins and friendship history of Brightkite users, a location-based social networking platform. It was originally collected by Cho et al. [89] and published in the SNAP Dataset Repository [90]. Personal events consist of sequences of location check-ins. We convert the original latitude and longitude coordinates into Geohash-8 representations [91, 92], short alphanumeric strings encoding geographic locations. Nearby locations share similar geohash prefixes, while distant ones differ. Example geohashes include `9v6kpmr1`, `gcpwkeq6`, and `u0yhxgm1`. Relational events capture friendship connections among users. The dataset includes 58,228 users and 5,130,866 events. Only personal events have timestamps; relational events do not.

`pres-gowalla`. The dataset also contains the location check-in and friendship history of another social network platform, Gowalla. It was also originally collected by Cho et al. [89] and published in the SNAP Repository [90]. We processed and formatted the data following the same approach used for `pres-brightkite`. The dataset contains personal events from geohash check-ins and relational events from friendship connections, totaling 8,342,943 events from 196,591 users.

pres-amazon-clothing. The dataset contains Amazon product reviews and ratings in the *Clothing, Shoes and Jewelry* category, spanning from May 1996 to July 2014. The raw data was originally collected by McAuley et al. [93]. In this dataset, we define personal events as sequences of product IDs and ratings reviewed by a user, for example: B000MLDCZ2:5 and B001OE3F08:3. Relational events represent co-review patterns, where two users have reviewed at least three of the same products. The dataset contains event sequences from 185,986 users, with a total of 1,591,947 events.

pres-amazon-electronics. The dataset contains Amazon product reviews and ratings in the *Electronics* category, originally collected by McAuley et al. [93]. As in pres-amazon-clothing, personal events are defined as sequences of product IDs and ratings, while relational events capture co-review patterns. In total, the dataset contains 2,938,178 events from 254,064 users.

pres-github. This dataset contains GitHub user activity from January 2025, collected from the GH Archive. Personal events include actions such as Push, CreateBranch, CreateRepository, PullRequestOpened, IssuesOpened, and Fork. Relational events represent project collaboration, where two users are linked if both contributed at least five commits or pull requests to the same repository. The dataset includes 102,878,895 events from 3,669,079 users. Only personal events include timestamps; relational events do not, similar to pres-brightkite and pres-gowalla.

Variability of the datasets. As shown in Table 1, the pres datasets vary significantly across multiple aspects. The number of users ranges from around 58 thousand in pres-brightkite to more than 3.5 million in pres-github. The number of events also varies, from approximately 1.5 million in pres-brightkite to over 100 million in pres-github. The ratio between relational and personal events ranges from around 1:3 in pres-gowalla to approximately 1:80 in pres-amazon-clothing. The number of unique events also differs widely, from just 24 in pres-github to more than 1 million in pres-gowalla. In addition, we observe variability in the number of users having personal events, relational events, and both. Some datasets have more users with relational events than with personal events (e.g., pres-brightkite, pres-gowalla), while others show the opposite trend (e.g., pres-amazon-clothing, pres-amazon-electronics, pres-github). These differences in dataset properties present distinct challenges for modeling user events in each dataset.

4.2 Prediction Tasks

From the pres datasets, we define two prediction tasks: one for relational events and one for personal events. These tasks are designed to enable fair comparisons between graph-based, sequence-based, and hybrid models. Relational event prediction focuses on predicting future or held-out subset of user-to-user interactions, similar to link prediction. Personal event prediction aims to predict the likelihood of future occurrence of personal events without requiring exact timestamps, for example, predicting the next 20 personal events given a user's first 100. In both tasks, observed events are compared against negative samples drawn from events not associated with the user. For reproducibility, pre-generated negative samples for validation and test sets are provided in the dataset repository.

Relational event prediction tasks. The corresponding tasks for pres-brightkite and pres-gowalla involve friend recommendation. We construct the training data by randomly splitting all relational events into 70% training, 10% validation, and 20% test sets. We also generate negative samples for the validation and test sets. Following Gastinger et al. [35], we adopt a *1-vs-1000* negative sampling scheme, in which 1,000 negative events are sampled for each relational event in the prediction set. Negative samples are drawn via uniform random sampling of users, excluding those who already have relational events with the target user in the training set.

For the pres-github dataset, the relational event prediction task is defined as collaboration prediction, which involves predicting which users collaborate with a given user. The train, validation, and test splits follow the same procedure as in pres-brightkite, including the sampling method. However, due to the large size of the dataset, we adopt a *1-vs-300* negative sampling scheme.

For the pres-amazon-clothing and pres-amazon-electronics datasets, the task is predicting co-review relationships, i.e., which users share at least three products they reviewed. Co-review patterns can reveal how one account may be related to another, which in some cases can help detect fraudulent review syndicates. In these datasets, relational events have timestamp information, i.e., the first time the co-review condition is met. As such, the train, validation, and test splits respect event timestamps. Specifically, we split each user's relational events by taking the last 20% for test, the previous 10% for validation, and the rest for training. To manage large histories of some users, we cap test and val sets at 20 and 10 events per user, respectively. Personal events are also split into ‘observed’ and ‘unobserved’ sets based on the timestamp cut-off in the relational event split, with only the observed set used for training. As in `pres-brightkite`, we adopt a *1-vs-1000* negative sampling scheme.

Personal event prediction tasks. The task for `pres-brightkite` and `pres-gowalla` is to predict the likelihood of a user checking in at a given geohash location in the future. We split each user’s personal events by taking the last 20% for test, the previous 10% for validation, and the rest for training. We also cap the number of events in the test and validation sets to at most 20 and 10 per user, respectively. Relational events are also split into ‘observed’ and ‘unobserved’ sets based on the timestamp cutoff from the personal event split, with only the observed set used in training. Since personal events are more frequent than relational ones, we adopt a *1-vs-500* negative sampling scheme. As geohash strings encode hierarchical spatial information (e.g., earlier characters represent broader regions), we apply stratified hierarchical sampling. Specifically, negatives are stratified by shared geohash prefixes, from matching the first five characters to none, ensuring a mix of nearby and distant locations.

For the `pres-amazon` datasets, the task is to predict future products a user will review and the corresponding ratings, as denoted in their personal event data. We adopt the same train/val/test split strategy as in `pres-brightkite`, along with a *1-vs-500* negative sampling scheme. Negative samples for each personal event (e.g., `B0010E3F08:3`) are drawn from three sources: (1) the same product with different ratings (e.g., `B0010E3F08:5`); (2) other personal events not in the user’s training data; and (3) samples from the second set with randomly perturbed ratings.

In the `pres-github` dataset, the number of unique events in the personal event set is only 24, corresponding to the list of possible GitHub activities. Thus, the task construction used in the previous datasets is not applicable to `pres-github`. We decided to omit this dataset from the set of datasets used for creating personal event prediction tasks.

Full event sequence. In addition to the datasets containing prediction tasks described above, we also publish a version of each dataset that includes all personal and relational events for all users, without any assigned tasks, train/val/test splits, or pre-specified negative samples. This is intended to facilitate future works that may wish to generate other prediction tasks not covered in this paper.

5 Experiments

5.1 Relational event prediction tasks

Experiment setup. We perform relational event prediction experiments on all five `pres` datasets, following the task setup described earlier. We evaluate several sets of baseline methods:

1. In the first set, we use only relational event data. We construct a user graph where edges represent relational events between two users, ignoring timestamp information. We then run static graph methods, GCN [45] and GAT [47], on this graph.
2. In the second set, we use a sequence model, BERT [94], to encode each user’s last 100 personal events from the training set. The resulting user embedding is added as input to the GCN and GAT models from the first set, denoted as GCN+S and GAT+S, respectively.
3. In the third set, we convert each unique personal event into a node and add it to the user graph from the first set, creating edges between users and their personal event nodes. As in the second set, we use only the last 100 personal events per user. We then run GCN and GAT on this graph, denoted as GCN-RP and GAT-RP.
4. Lastly, based on the graph containing user and personal event nodes from the third set, we add timestamp information to construct a temporal graph. For datasets that lack timestamps for relational events, we inject these events randomly into the sequence of personal events. We then run temporal graph models, TGN [32] and DyRep [33], on this graph.

The sequence model for capturing personal events in the second set is designed as a masked token prediction task using a BERT model with a masking probability of 0.3. A key benefit of using transformer-based models is flexibility in event tokenization. In `pres-brightkite` and `pres-gowalla`, personal events are 8-character geohash strings (e.g., `9q8yyk8y|9q8vzj5b|9q8vyzwk`). Since geohashes encode hierarchical geographic information, we apply hierarchical tokenization by splitting each into four two-character tokens with added prefixes (e.g., `gh12-9q`, `gh34-8y`, `gh12-yk`, `gh12-8y`). This roughly mimics hierarchical location modeling, such as identifying continent, country, city, and

Table 2: Performance results for relational event prediction tasks across various datasets.

Method	pres-brightkite					pres-gowalla				
Metric	MRR (%)	H@5 (%)	H@10 (%)	H@50 (%)	H@100 (%)	MRR (%)	H@5 (%)	H@10 (%)	H@50 (%)	H@100 (%)
Static graph models on relational event graph										
GCN	37.3±0.8	50.8±1.0	61.7±0.9	83.2±0.4	89.5±0.3	40.3±0.9	54.5±0.9	65.8±0.8	86.5±0.4	92.0±0.2
GAT	36.2±1.4	48.7±1.4	59.5±1.2	81.4±0.8	88.5±0.6	40.7±1.5	54.1±1.6	64.9±1.5	85.3±1.3	91.1±1.0
Static graph models on relational event graph + sequence embedding from personal event data										
GCN+S	43.9±0.7	57.8±0.8	67.8±0.8	**86.5±0.3**	**91.5±0.1**	**44.9±1.0**	**59.4±1.1**	**69.8±1.0**	**88.1±0.5**	**92.8±0.3**
GAT+S	**44.8±1.1**	**58.5±1.1**	**68.2±1.1**	86.2±0.5	91.5±0.4	44.9±0.9	58.8±0.6	69.0±0.4	87.0±0.4	92.0±0.5
Static graph models on relational event graph + personal event nodes										
GCN-RP	8.7±0.9	11.0±1.2	15.7±1.7	35.6±3.7	49.8±4.5	17.0±0.9	22.1±1.2	29.8±1.6	56.4±3.0	70.8±2.9
GAT-RP	10.7±1.0	13.5±1.2	18.2±1.4	35.6±2.3	47.8±2.8	14.9±1.4	19.0±1.6	25.8±2.0	50.7±3.1	66.2±3.2
Temporal graph models on relational event graph + personal event nodes										
TGN	12.2±0.7	15.9±0.9	23.5±1.0	50.2±1.3	63.5±1.3	15.4±2.6	20.6±3.7	27.8±4.6	51.8±5.6	64.9±5.4
DyRep	7.1±0.4	8.9±0.6	13.7±0.9	36.0±1.7	50.7±2.1	8.8±1.0	11.2±1.3	15.8±1.7	34.8±3.5	48.6±5.1

Method	pres-amazon-clothing					pres-amazon-electronics				
Metric	MRR (%)	H@5 (%)	H@10 (%)	H@50 (%)	H@100 (%)	MRR (%)	H@5 (%)	H@10 (%)	H@50 (%)	H@100 (%)
Static graph models on relational event graph										
GCN	6.1±1.6	7.4±2.1	10.0±2.5	23.4±3.0	35.3±1.3	13.1±0.6	15.9±0.7	21.5±0.6	45.9±1.3	60.6±1.6
GAT	7.2±2.5	7.8±2.7	10.2±2.9	23.8±3.6	38.4±1.9	13.2±0.7	15.5±0.9	20.7±1.0	45.2±1.2	61.0±1.5
Static graph models on relational event graph + sequence embedding from personal event data										
GCN+S	4.5±0.3	5.5±0.6	9.3±0.8	29.0±0.5	40.4±0.4	14.7±0.5	19.1±0.8	**27.2±1.5**	**57.9±1.9**	**70.6±1.5**
GAT+S	7.7±2.1	8.5±2.1	**12.0±1.8**	**31.3±1.3**	**46.3±0.6**	14.4±1.7	16.7±1.8	21.6±1.4	43.6±2.5	58.4±3.4
Static graph models on relational event graph + personal event nodes										
GCN-RP	**8.7±1.4**	**9.2±1.4**	10.5±1.4	18.1±1.2	25.8±2.5	7.5±0.6	8.3±0.6	10.9±0.8	21.8±2.3	29.0±3.2
GAT-RP	6.5±1.0	7.9±1.0	10.9±1.4	25.7±3.2	39.8±3.7	**15.5±0.5**	17.2±0.4	20.5±0.7	35.5±3.0	46.9±3.6
Temporal graph models on relational event graph + personal event nodes										
TGN	3.5±0.5	4.1±1.2	6.9±1.2	23.7±0.8	39.0±1.3	13.8±0.3	**19.2±0.6**	26.4±0.9	48.8±0.9	61.5±0.7
DyRep	2.9±0.6	3.0±1.1	5.8±1.6	22.8±2.6	39.6±2.4	6.8±0.7	8.9±1.0	13.8±1.3	33.4±2.3	47.5±3.0

neighborhood. For the `pres-amazon` datasets, we apply similar tokenization by splitting each event into three product tokens and one rating token. We do not apply token splitting for `pres-github`.

For performance evaluation, following prior benchmarks [34, 35, 60], we use ranking-based metrics: Mean Reciprocal Rank (MRR) and Hits@k, evaluated at various k depending on the number of negative samples. Each baseline is run five times with different random seeds, and we report the mean and standard deviation.

Table 3: Relational event predictions on `pres-github`.

Method	MRR (%)	H@3 (%)	H@5 (%)	H@10 (%)	H@30 (%)
GCN	54.0±7.2	62.9±7.5	69.6±5.1	75.2±2.3	80.1±0.4
GAT	69.3±3.1	73.6±2.2	76.1±1.3	78.1±0.4	80.4±0.3
GCN+S	70.8±0.8	75.1±0.2	76.9±0.0	78.5±0.1	80.9±0.4
GAT+S	**74.2±0.6**	**77.0±0.4**	**78.7±0.3**	**80.6±0.1**	**84.5±0.2**
GCN-RP	22.3±2.6	23.3±2.9	28.8±3.1	37.8±3.3	57.5±3.5
GAT-RP	33.1±4.1	35.7±5.4	43.9±5.6	57.2±4.5	76.7±0.7
TGN	Out of GPU Memory				
DyRep	Out of GPU Memory				

Experiment results. Table 2 and Table 3 show the experiment results (additional results are available in Appendix D). In each table, bold numbers indicate the best-performing model on a given metric, and underlined numbers indicate the second best. As each dataset has its own characteristics, the results vary across datasets. However, there are some emerging patterns in the results that we highlight below.

- In all datasets and across all metrics, the best and second-best models incorporate both relational and personal events as input to their architectures.
- The graph models with personal event sequence embeddings (GCN+S and GAT+S) consistently perform well across all datasets and metrics. On `pres-brightkite`, `pres-gowalla`, and `pres-github`, they clearly outperform other models, ranking either first or second in all metrics. In `pres-amazon-clothing`, GAT+S performs best on Hits@k for larger k (10, 50, 100), and second-best on Hits@5 and MRR. Similarly, in `pres-amazon-electronics`, GCN+S ranks first on Hits@10, Hits@50, and Hits@100, and second on Hits@5 and MRR.
- In many datasets, adding personal events as nodes into the relational event graph decreases predictive performance on the relational link prediction task, as shown by the results of GCN-RP and GAT-RP. Notable exceptions are `pres-amazon-clothing` and `pres-amazon-electronics`, where they perform relatively well on MRR and Hits@5, but not on Hits@k metrics with larger k.

Table 4: Performance results for personal event prediction tasks across various datasets.

Method	`pres-brightkite`					`pres-gowalla`				
Metric	MRR (%)	H@3 (%)	H@5 (%)	H@10 (%)	H@50 (%)	MRR (%)	H@3 (%)	H@5 (%)	H@10 (%)	H@50 (%)
Sequential models										
BERT	34.2±0.1	35.6±0.2	37.4±0.2	40.1±0.2	50.1±0.3	15.3±0.2	15.7±0.3	18.6±0.3	23.4±0.3	43.1±0.3
BERT-n2v-p	33.8±0.1	35.1±0.2	36.9±0.2	39.6±0.2	49.8±0.2	14.4±0.2	14.8±0.2	17.7±0.2	22.6±0.2	42.4±0.2
BERT-n2v-i	**34.4±0.1**	**35.9±0.1**	**37.6±0.1**	**40.3±0.1**	50.3±0.2	15.0±0.3	15.4±0.3	18.3±0.3	23.2±0.3	42.7±0.3
Graph models on personal event only graph										
GCN	24.9±1.2	27.1±1.5	31.8±1.7	38.8±1.9	55.8±1.0	28.2±3.2	29.7±3.3	34.2±3.0	41.5±2.3	63.8±0.9
GAT	19.0±1.4	20.3±1.7	24.9±1.9	32.1±2.0	52.3±1.6	15.4±1.2	15.4±1.4	20.0±1.5	28.4±1.6	59.3±1.1
TGN	23.5±0.2	24.5±0.3	28.9±0.4	37.1±0.8	54.5±1.1	10.7±0.4	10.6±0.6	14.4±1.1	21.5±1.8	42.7±4.9
DyRep	19.8±2.9	21.4±3.2	26.5±2.5	35.4±1.6	**57.2±1.7**	7.4±0.6	6.4±0.8	10.0±1.0	17.8±1.0	42.9±2.1
Graph models on personal and relational event graph										
GCN-PR	25.4±1.2	27.5±1.3	31.9±1.5	38.2±1.7	54.7±1.6	**30.3±5.1**	**32.0±5.4**	**36.8±5.2**	**44.2±4.4**	**65.4±1.2**
GAT-PR	18.8±0.5	20.3±0.6	25.2±0.6	32.9±0.6	53.3±0.6	16.0±0.6	16.0±0.7	20.5±0.8	28.6±0.9	59.2±0.9
TGN-PR	29.5±2.3	33.5±2.2	35.3±2.2	36.0±2.4	36.1±2.4	14.0±2.0	15.7±2.2	17.0±2.4	17.9±2.5	18.2±2.6
DyRep-PR	23.4±2.7	27.5±3.5	30.5±3.2	32.7±4.2	33.3±4.8	10.5±1.4	11.4±1.8	12.4±2.2	13.1±2.8	13.6±3.4

Method	`pres-amazon-clothing`					`pres-amazon-electronics`				
Metric	MRR (%)	H@3 (%)	H@5 (%)	H@10 (%)	H@50 (%)	MRR (%)	H@3 (%)	H@5 (%)	H@10 (%)	H@50 (%)
Sequential models										
BERT	3.3±0.0	2.3±0.0	3.5±0.1	6.1±0.1	22.4±0.2	8.1±0.2	7.7±0.2	10.7±0.3	16.1±0.3	38.1±0.2
BERT+n2v-p	3.4±0.0	2.4±0.0	3.6±0.0	6.2±0.1	22.7±0.2	8.1±0.1	7.7±0.1	10.7±0.2	16.1±0.2	38.2±0.1
BERT+n2v-i	3.3±0.0	2.3±0.0	3.5±0.1	6.1±0.1	22.6±0.2	8.1±0.1	7.8±0.1	10.7±0.0	16.1±0.1	38.0±0.2
Graph models on personal event only graph										
GCN	10.8±1.7	11.2±2.0	14.1±1.7	19.0±1.1	32.8±0.5	13.3±2.0	13.3±2.5	18.4±2.9	27.6±3.1	55.6±0.9
GAT	3.5±0.0	2.6±0.0	4.0±0.1	7.1±0.1	22.7±0.2	7.4±0.4	6.4±0.4	9.6±0.5	16.1±0.7	44.0±0.8
TGN	9.3±0.9	8.0±0.8	12.8±2.2	25.4±6.8	**44.4±3.4**	16.2±1.6	**17.4±2.0**	22.6±1.8	30.8±1.2	54.2±0.8
DyRep	8.9±1.3	8.1±2.4	13.5±4.1	25.1±6.7	43.5±5.7	11.4±0.4	10.6±0.6	15.3±0.8	25.8±1.0	55.1±0.8
Graph models on personal and relational event graph										
GCN-PR	**10.9±1.3**	**11.4±1.5**	14.3±1.3	19.1±0.8	32.8±0.6	**16.6±1.5**	17.3±1.9	22.2±2.1	30.6±2.0	**56.3±0.8**
GAT-PR	3.5±0.1	2.6±0.1	4.0±0.1	7.1±0.2	22.5±0.2	8.0±0.3	7.2±0.4	10.5±0.5	17.2±0.7	44.8±0.6
TGN-PR	9.3±0.7	7.8±1.1	13.9±3.3	30.4±3.7	41.7±1.9	15.6±0.6	16.8±0.8	**22.8±0.8**	**32.7±0.6**	54.0±1.0
DyRep-PR	10.5±0.2	9.7±0.5	**17.1±0.6**	**32.9±0.3**	41.3±0.4	14.3±0.5	15.0±0.7	21.2±0.9	32.3±1.4	53.3±2.8

- The performance of temporal graph methods (TGN and DyRep) on the relational link prediction task using graphs with personal event nodes is noticeably lower compared to static graph models on nearly all datasets. A notable exception is `pres-amazon-electronics`, where TGN performs relatively well. For the large dataset of `pres-github`, both TGN and DyRep suffer from GPU out of memory error, even when using small batch size.

Although GCN+S and GAT+S perform relational event prediction in two stages, where they first generate user embeddings from personal event sequences and then incorporate them into the graph learning process, they still perform well across datasets. In contrast, TGN and DyRep use a single-step approach that directly integrates temporal dynamics but operate on graph structures where personal events are represented as nodes. These differences highlight an opportunity for future exploration on how best to represent temporal dynamics of personal events within a user, while jointly modeling the full structure that includes user-to-user relational events in an end-to-end fashion.

5.2 Personal event prediction tasks

Experiment setup. We perform personal event prediction experiments on all `pres` datasets except `pres-github`. In these experiments, we evaluate several sets of baseline methods:

1. The first model is a sequential model that uses only personal event data. We use a BERT architecture with a prediction head to compute the likelihood of a user having a particular personal event in the future. For each user, we use the last 100 personal events in the training set to predict the likelihood of future events.
2. In the second set, we use node2vec [95] to learn the graph structure of relational events and generate a graph embedding for each user. We then incorporate the embedding into the BERT sequence model. We evaluate two versions of the model: (a) incorporating the graph embedding post transformer module and before the prediction head (`BERT-n2v-p`), and (b) using the embedding as a special input token to the transformer module (`BERT-n2v-i`).

3. In the third set, we use graph-based models on personal event–only data by creating a bipartite graph of user nodes and personal event nodes, based on the last 100 personal events per user. We run both static graph models (GCN and GAT) and temporal graph models (TGN and DyRep) on this graph.
4. In the last set, we augment the graph in the third set with relational event data by adding relational event edges between users. We then run GCN, GAT, TGN, and DyRep on this graph, denoted as GCN-PR, GAT-PR, TGN-PR, and DyRep-PR, respectively.

Similar to the sequence embedding used in relational event prediction tasks, we apply split tokenization for the BERT model in personal event prediction to allow more flexibility in modeling events. We use the same tokenization scheme for each dataset as described earlier. For evaluation, we report MRR and Hits@k at various values of k. Each baseline is run five times with different random seeds, and we report the mean and standard deviation.

Experiment results. Table 4 shows the results for the personal event prediction task. As in the relational event task, results vary across datasets due to their unique characteristics, with even more variations in this setting. We discuss some of the results as follows.

- In most cases, the best models incorporate both personal and relational events as input to their architectures.
- The sequence models perform well on `pres-brightkite` across all metrics. The base BERT model, which uses only personal event data, already shows strong performance. Adding relational event node2vec embeddings may either improve or degrade performance. In `pres-brightkite`, adding the embedding after the transformer module reduces performance, while using it as a special input token improves it. However, the changes are relatively minor but sufficient to make `BERT-n2v-i` the best-performing model on `pres-brightkite`. Similar minimal changes are observed in other datasets.
- The static graph model, GCN in particular, performs surprisingly well on `pres-gowalla`. The best performance is achieved by the GCN-PR model, which is trained on data containing both personal and relational events in a graph with user nodes and personal event nodes. GCN-PR also performs relatively well on `pres-amazon-clothing` and `pres-amazon-electronics`. However, the GAT-based models perform noticeably worse than their GCN counterparts.
- The temporal graph models perform relatively well on the `pres-amazon-clothing` and `pres-amazon-electronics` datasets, particularly on the Hits@5 and Hits@10 metrics. TGN and DyRep perform better on graphs that include both personal and relational events. A notable exception is the Hits@50 metric.

The results show that there is no single model that consistently performs best across all datasets. Some models work well on certain datasets but not on others. The only consistent pattern is that the best-performing models usually use both personal and relational events. This opens up opportunities for designing better models that can effectively integrate both types of information.

6 Conclusions and Limitations

In this work, we aim to advance user event modeling by introducing a unified framework that captures both personal and relational events. We curate and release a collection of public datasets with corresponding prediction tasks, all aligned under a formalization that integrates both event types to provide a more complete view of user behavior. Through empirical evaluation, we demonstrate that models leveraging both event types consistently outperform those using only one. We also show that existing methods, originally developed for either sequential or relational data, even with some adaptations to handle both (e.g., temporal graph models), are less effective across many of our prediction tasks. These findings highlight the need for further study of unified user event modeling.

A key challenge in this work is dataset curation, as many public datasets have already been collapsed into either graph-only or sequence-only formats, often discarding personal or relational events in the process. While we were able to gather and unify a set of datasets that include both event types, they may not fully capture the diversity and complexity of user event modeling across domains. Another limitation is that our current formulation does not support event-level or user-level features, presenting an opportunity for future work to extend the framework toward feature-aware modeling.

References

[1] Erasmo Purificato, Ludovico Boratto, and Ernesto William De Luca. User modeling and user profiling: A comprehensive survey. *arXiv preprint arXiv:2402.09660*, 2024.

[2] Alejandro García Martín, Raquel Martínez González, Andrés García, and Gabriel Villarrubia. A survey for user behavior analysis based on machine learning techniques: current models and applications. *Applied Intelligence*, 51:8110–8127, 2021.

[3] Songgaojun Deng, Maarten de Rijke, and Yue Ning. Advances in human event modeling: From graph neural networks to language models. In *Proceedings of the 30th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pages 6459–6469, 2024.

[4] Zhicheng He, Weiwen Liu, Wei Guo, Jiarui Qin, Yingxue Zhang, Yaochen Hu, and Ruiming Tang. A survey on user behavior modeling in recommender systems. *arXiv preprint arXiv:2302.11087*, 2023.

[5] Meng Zeng, Hong Cao, Min Chen, and Yujie Li. User behaviour modeling, recommendations, and purchase prediction during shopping festivals. *Electronic Markets*, 29(2):205–217, 2019.

[6] Ahmed Abdel-Hafez and Yanchun Xu. A survey of user modelling in social media websites. *Computer and Information Science*, 6(4):59–71, 2013.

[7] Guangyuan Piao and John G Breslin. Inferring user interests in microblogging social networks: A survey. *User Modeling and User-Adapted Interaction*, 28(3):277–329, 2018.

[8] Qixiang Fang, Zhihan Zhou, Francesco Barbieri, Yozen Liu, Leonardo Neves, Dong Nguyen, Daniel L Oberski, Maarten W Bos, and Ron Dotsch. General-purpose user modeling with behavioral logs: A snapchat case study. In *Proceedings of the 46th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 2431–2436, 2023.

[9] Luisa Hernandez Aros, Ximena Bustamante Molano, Francisco Gutierrez-Portela, and Juan Jose Moreno Hernandez. Financial fraud detection through the application of machine learning techniques: a literature review. *Palgrave Communications*, 11(1):1–15, 2024.

[10] Akib Mashrur, Wei Luo, Nayyar A Zaidi, and Antonio Robles-Kelly. Machine learning for financial risk management: A survey. *IEEE Access*, 8:203203–203223, 2020.

[11] Ahmet Murat Ozbayoglu, Mehmet Ugur Gudelek, and Omer Berat Sezer. Deep learning for financial applications: A survey. *Applied Soft Computing*, 93:106384, 2020.

[12] S Navid Hojaji, Mahmood Yahyazadehfar, and Bahareh Abedin. Machine learning in behavioral finance: A systematic literature review. *The Journal of Financial Data Science*, 4(3):129–146, 2022.

[13] Yan Zhao, Shoujin Wang, Yan Wang, and Hongwei Liu. Mbsrs: A multi-behavior streaming recommender system. *Information Sciences*, 631:1–17, 2023.

[14] Daniel Gayo-Avello. A survey on session detection methods in query logs and a proposal for future evaluation. *Information Sciences*, 179(12):1822–1843, 2009.

[15] Paul Covington, Jay Adams, and Emre Sargin. Deep neural networks for youtube recommendations. In *Proceedings of the 10th ACM Conference on Recommender Systems*, pages 191–198, 2016.

[16] Xavier Amatriain and Justin Basilico. Big & personal: data and models behind netflix recommendations. *ACM SIGKDD Explorations Newsletter*, 14(2):37–48, 2013.

[17] Ahmad H Lashkari, Meng Chen, and Ali A Ghorbani. A survey on user profiling model for anomaly detection in cyberspace. *Journal of Information Security and Applications*, 34:38–56, 2017.

[18] Gang Wang, Xinyang Zhang, Shaomei Tang, Christo Wilson, Haitao Zheng, and Ben Y Zhao. Clickstream user behavior models. *ACM Transactions on the Web (TWEB)*, 11(4):1–37, 2017.

[19] Chen Gao, Yu Zheng, Nian Li, Yinfeng Li, Yingrong Qin, Jinghua Piao, Yuhan Quan, Jianxin Chang, Depeng Jin, Xiangnan He, et al. A survey of graph neural networks for recommender systems: Challenges, methods, and directions. *ACM Transactions on Recommender Systems*, 1 (1):1–51, 2023.

[20] Qingyu Guo, Fuzhen Zhuang, Chuan Qin, Hengshu Zhu, Xing Xie, Hui Xiong, and Qing He. A survey on knowledge graph-based recommender systems. *IEEE Transactions on Knowledge and Data Engineering*, 34(8):3549–3568, 2020.

[21] Shiwen Wu, Fei Sun, Wentao Zhang, Xu Xie, and Bin Cui. Graph neural networks in recommender systems: a survey. *ACM Computing Surveys*, 55(5):1–37, 2022.

[22] Zihao Li, Chao Yang, Yakun Chen, Xianzhi Wang, Hongxu Chen, Guandong Xu, Lina Yao, and Michael Sheng. Graph and sequential neural networks in session-based recommendation: A survey. *ACM Computing Surveys*, 57(2):1–37, 2024.

[23] Shu Chen, Zitao Xu, Weike Pan, Qiang Yang, and Zhong Ming. A survey on cross-domain sequential recommendation. *ArXiv*, abs/2401.04971, 2024.

[24] Liwei Pan, Weike Pan, Meiyan Wei, Hongzhi Yin, and Zhong Ming. A survey on sequential recommendation. *arXiv preprint arXiv:2412.12770*, 2024.

[25] Tesfaye Fenta Boka, Zhendong Niu, and Rama Bastola Neupane. A survey of sequential recommendation systems: Techniques, evaluation, and future directions. *Information Systems*, page 102427, 2024.

[26] Fei Sun, Jun Liu, Jian Wu, Changhua Pei, Xiao Lin, Wenwu Ou, and Peng Jiang. Bert4rec: Sequential recommendation with bidirectional encoder representations from transformer. In *Proceedings of the 28th ACM international conference on information and knowledge management*, pages 1441–1450, 2019.

[27] Yong Kiam Tan, Xinxing Xu, and Yong Liu. Improved recurrent neural networks for session-based recommendations. *Proceedings of the 1st Workshop on Deep Learning for Recommender Systems*, 2016.

[28] Wang-Cheng Kang and Julian McAuley. Self-attentive sequential recommendation. *2018 IEEE International Conference on Data Mining (ICDM)*, pages 197–206, 2018.

[29] Angelica Liguori, Luciano Caroprese, Marco Minici, Bruno Veloso, Francesco Spinnato, Mirco Nanni, Giuseppe Manco, and Joao Gama. Modeling events and interactions through temporal processes–a survey. *arXiv preprint arXiv:2303.06067*, 2023.

[30] 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.

[31] Da Xu, Chuanwei Ruan, Evren Korpeoglu, Sushant Kumar, and Kannan Achan. Inductive representation learning on temporal graphs. *arXiv preprint arXiv:2002.07962*, 2020.

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

[33] Rakshit Trivedi, Mehrdad Farajtabar, Prasenjeet Biswal, and Hongyuan Zha. Dyrep: Learning representations over dynamic graphs. In *International conference on learning representations*, 2019.

[34] 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. *Advances in Neural Information Processing Systems*, 36:2056–2073, 2023.

[35] Julia Gastinger, Shenyang Huang, Michael Galkin, Erfan Loghmani, Ali Parviz, Farimah Poursafaei, Jacob Danovitch, Emanuele Rossi, Ioannis Koutis, Heiner Stuckenschmidt, et al. Tgb 2.0: A benchmark for learning on temporal knowledge graphs and heterogeneous graphs. *Advances in neural information processing systems*, 37:140199–140229, 2024.

[36] Alan G Hawkes. Spectra of some self-exciting and mutually exciting point processes. *Biometrika*, 1971.

[37] Hongyuan Mei and Jason M. Eisner. The neural hawkes process: A neurally self-modulating multivariate point process. In *Advances in Neural Information Processing Systems*, volume 30, pages 6754–6764, 2017.

[38] Simiao Zuo, Haoming Jiang, Zitong Li, Tuo Zhao, and Hongyuan Zha. Transformer hawkes process. In *International Conference on Machine Learning*, pages 11692–11702, 2020.

[39] Qiang Zhang, Aldo Lipani, Omer Kirnap, and Emine Yilmaz. Self-attentive hawkes process. In *International conference on machine learning*, pages 11183–11193. PMLR, 2020.

[40] Steffen Rendle, Christoph Freudenthaler, and Lars Schmidt-Thieme. Factorizing personalized markov chains for next-basket recommendation. In *Proceedings of the 19th international conference on World wide web*, pages 811–820, 2010.

[41] Xiangnan He, Lizi Liao, Hanwang Zhang, Liqiang Nie, Xia Hu, and Tat-Seng Chua. Neural collaborative filtering. In *Proceedings of the 26th international conference on world wide web*, pages 173–182, 2017.

[42] Balázs Hidasi, Alexandros Karatzoglou, Linas Baltrunas, and Domonkos Tikk. Session-based recommendations with recurrent neural networks. *arXiv preprint arXiv:1511.06939*, 2015.

[43] Wang-Cheng Kang and Julian McAuley. Self-attentive sequential recommendation. In *2018 IEEE international conference on data mining (ICDM)*, pages 197–206. IEEE, 2018.

[44] Tong Zhao, Yozen Liu, Matthew Kolodner, Kyle Montemayor, Elham Ghazizadeh, Ankit Batra, Zihao Fan, Xiaobin Gao, Xuan Guo, Jiwen Ren, et al. Gigl: Large-scale graph neural networks at snapchat. *arXiv e-prints*, pages arXiv–2502, 2025.

[45] Thomas N Kipf and Max Welling. Semi-supervised classification with graph convolutional networks. In *International Conference on Learning Representations*, 2017.

[46] Will Hamilton, Zhitao Ying, and Jure Leskovec. Inductive representation learning on large graphs. *Advances in neural information processing systems*, 30, 2017.

[47] Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Liò, and Yoshua Bengio. Graph attention networks. In *International Conference on Learning Representations*, 2018.

[48] Ziniu Hu, Yuxiao Dong, Kuansan Wang, and Yizhou Sun. Heterogeneous graph transformer. In *Proceedings of the web conference 2020*, pages 2704–2710, 2020.

[49] Youngjoo Seo, Michaël Defferrard, Pierre Vandergheynst, and Xavier Bresson. Structured sequence modeling with graph convolutional recurrent networks. In *Neural information processing: 25th international conference, ICONIP 2018, Siem Reap, Cambodia, December 13-16, 2018, proceedings, part I 25*, pages 362–373. Springer, 2018.

[50] Zhen Han, Yunpu Ma, Yuyi Wang, Stephan Günnemann, and Volker Tresp. Graph hawkes neural network for forecasting on temporal knowledge graphs. *arXiv preprint arXiv:2003.13432*, 2020.

[51] Ying Yin, Li-Xin Ji, Jian-Peng Zhang, and Yu-Long Pei. Dhne: Network representation learning method for dynamic heterogeneous networks. *IEEE Access*, 7:134782–134792, 2019.

[52] Hansheng Xue, Luwei Yang, Wen Jiang, Yi Wei, Yi Hu, and Yu Lin. Modeling dynamic heterogeneous network for link prediction using hierarchical attention with temporal rnn. In *Machine Learning and Knowledge Discovery in Databases: European Conference, ECML PKDD 2020, Ghent, Belgium, September 14–18, 2020, Proceedings, Part I*, pages 282–298. Springer, 2021.

[53] Ranran Bian, Yun Sing Koh, Gillian Dobbie, and Anna Divoli. Network embedding and change modeling in dynamic heterogeneous networks. In *Proceedings of the 42nd international ACM SIGIR conference on research and development in information retrieval*, pages 861–864, 2019.

[54] Seyed Mehran Kazemi, Rishab Goel, Kshitij Jain, Ivan Kobyzev, Akshay Sethi, Peter Forsyth, and Pascal Poupart. Representation learning for dynamic graphs: A survey. *Journal of Machine Learning Research*, 21(70):1–73, 2020.

[55] Chongjian Yue, Lun Du, Qiang Fu, Wendong Bi, Hengyu Liu, Yu Gu, and Di Yao. Htgn-btw: Heterogeneous temporal graph network with bi-time-window training strategy for temporal link prediction. *arXiv preprint arXiv:2202.12713*, 2022.

[56] Ce Li, Rongpei Hong, Xovee Xu, Goce Trajcevski, and Fan Zhou. Simplifying temporal heterogeneous network for continuous-time link prediction. In *Proceedings of the 32nd ACM International Conference on Information and Knowledge Management*, pages 1288–1297, 2023.

[57] Zixuan Li, Xiaolong Jin, Wei Li, Saiping Guan, Jiafeng Guo, Huawei Shen, Yuanzhuo Wang, and Xueqi Cheng. Temporal knowledge graph reasoning based on evolutional representation learning. In *Proceedings of the 44th international ACM SIGIR conference on research and development in information retrieval*, pages 408–417, 2021.

[58] Woojeong Jin, Meng Qu, Xisen Jin, and Xiang Ren. Recurrent event network: Autoregressive structure inference over temporal knowledge graphs. *arXiv preprint arXiv:1904.05530*, 2019.

[59] Zixuan Li, Saiping Guan, Xiaolong Jin, Weihua Peng, Yajuan Lyu, Yong Zhu, Long Bai, Wei Li, Jiafeng Guo, and Xueqi Cheng. Complex evolutional pattern learning for temporal knowledge graph reasoning. *arXiv preprint arXiv:2203.07782*, 2022.

[60] 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. *arXiv preprint arXiv:2502.02975*, 2025.

[61] Weihua Hu, Matthias Fey, Marinka Zitnik, Yuxiao Dong, Hongyu Ren, Bowen Liu, Michele Catasta, and Jure Leskovec. Open graph benchmark: Datasets for machine learning on graphs. *Advances in neural information processing systems*, 33:22118–22133, 2020.

[62] Weihua Hu, Matthias Fey, Hongyu Ren, Maho Nakata, Yuxiao Dong, and Jure Leskovec. Ogb-lsc: A large-scale challenge for machine learning on graphs. *arXiv preprint arXiv:2103.09430*, 2021.

[63] Fangzhao Wu, Ying Qiao, Jiun-Hung Chen, Chuhan Wu, Tao Qi, Jianxun Lian, Danyang Liu, Xing Xie, Jianfeng Gao, Winnie Wu, et al. Mind: A large-scale dataset for news recommendation. In *Proceedings of the 58th annual meeting of the association for computational linguistics*, pages 3597–3606, 2020.

[64] Guanghu Yuan, Fajie Yuan, Yudong Li, Beibei Kong, Shujie Li, Lei Chen, Min Yang, Chenyun Yu, Bo Hu, Zang Li, et al. Tenrec: A large-scale multipurpose benchmark dataset for recommender systems. *Advances in Neural Information Processing Systems*, 35:11480–11493, 2022.

[65] Jiaqi Zhang, Yu Cheng, Yongxin Ni, Yunzhu Pan, Zheng Yuan, Junchen Fu, Youhua Li, Jie Wang, and Fajie Yuan. Ninerec: A benchmark dataset suite for evaluating transferable recommendation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2024.

[66] Jieming Zhu, Quanyu Dai, Liangcai Su, Rong Ma, Jinyang Liu, Guohao Cai, Xi Xiao, and Rui Zhang. Bars: Towards open benchmarking for recommender systems. In *Proceedings of the 45th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 2912–2923, 2022.

[67] Dmitry Osin, Igor Udovichenko, Viktor Moskvoretskii, Egor Shvetsov, and Evgeny Burnaev. Ebes: Easy benchmarking for event sequences. *arXiv preprint arXiv:2410.03399*, 2024.

[68] Siqiao Xue, Xiaoming Shi, Zhixuan Chu, Yan Wang, Hongyan Hao, Fan Zhou, Caigao Jiang, Chen Pan, James Y Zhang, Qingsong Wen, et al. Easytpp: Towards open benchmarking temporal point processes. *arXiv preprint arXiv:2307.08097*, 2023.

[69] Ivan Karpukhin, Foma Shipilov, and Andrey Savchenko. Hotpp benchmark: Are we good at the long horizon events forecasting? *arXiv preprint arXiv:2406.14341*, 2024.

[70] Bing Yu, Haoteng Yin, and Zhanxing Zhu. Spatio-temporal graph convolutional networks: A deep learning framework for traffic forecasting. *arXiv preprint arXiv:1709.04875*, 2017.

[71] Cunjun Yu, Xiao Ma, Jiawei Ren, Haiyu Zhao, and Shuai Yi. Spatio-temporal graph transformer networks for pedestrian trajectory prediction. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XII 16*, pages 507–523. Springer, 2020.

[72] Lei Bai, Lina Yao, Can Li, Xianzhi Wang, and Can Wang. Adaptive graph convolutional recurrent network for traffic forecasting. *Advances in neural information processing systems*, 33:17804–17815, 2020.

[73] 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*, pages 753–763, 2020.

[74] Yan Hai, Dongyang Wang, Zhizhong Liu, Jitao Zheng, and Chengrui Ding. A study of recommendation methods based on graph hybrid neural networks and deep crossing. *Electronics*, 13(21):4224, 2024.

[75] Lei Shi, Yifan Zhang, Jian Cheng, and Hanqing Lu. Two-stream adaptive graph convolutional networks for skeleton-based action recognition. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 12026–12035, 2019.

[76] Kelong Mao, Xi Xiao, Tingyang Xu, Yu Rong, Junzhou Huang, and Peilin Zhao. Molecular graph enhanced transformer for retrosynthesis prediction. *Neurocomputing*, 457:193–202, 2021.

[77] Jiawei Zhang, Haopeng Zhang, Congying Xia, and Li Sun. Graph-bert: Only attention is needed for learning graph representations. arxiv 2020. *arXiv preprint arXiv:2001.05140*, 2001.

[78] Chloe Wang, Oleksii Tsepa, Jun Ma, and Bo Wang. Graph-mamba: Towards long-range graph sequence modeling with selective state spaces. *arXiv preprint arXiv:2402.00789*, 2024.

[79] Ali Behrouz and Farnoosh Hashemi. Graph mamba: Towards learning on graphs with state space models. In *Proceedings of the 30th ACM SIGKDD conference on knowledge discovery and data mining*, pages 119–130, 2024.

[80] Yinan Huang, Siqi Miao, and Pan Li. What can we learn from state space models for machine learning on graphs? *arXiv preprint arXiv:2406.05815*, 2024.

[81] Zifeng Ding, Yifeng Li, Yuan He, Antonio Norelli, Jingcheng Wu, Volker Tresp, Yunpu Ma, and Michael Bronstein. Dygmamba: Efficiently modeling long-term temporal dependency on continuous-time dynamic graphs with state space models. *arXiv preprint arXiv:2408.04713*, 2024.

[82] Dongyuan Li, Shiyin Tan, Ying Zhang, Ming Jin, Shirui Pan, Manabu Okumura, and Renhe Jiang. Dyg-mamba: Continuous state space modeling on dynamic graphs. *arXiv preprint arXiv:2408.06966*, 2024.

[83] Ali Behrouz, Ali Parviz, Mahdi Karami, Clayton Sanford, Bryan Perozzi, and Vahab Mirrokni. Best of both worlds: Advantages of hybrid graph sequence models. *arXiv preprint arXiv:2411.15671*, 2024.

[84] Weijie Liu, Peng Zhou, Zhe Zhao, Zhiruo Wang, Qi Ju, Haotang Deng, and Ping Wang. K-bert: Enabling language representation with knowledge graph. In *Proceedings of the AAAI conference on artificial intelligence*, volume 34, pages 2901–2908, 2020.

[85] Michihiro Yasunaga, Hongyu Ren, Antoine Bosselut, Percy Liang, and Jure Leskovec. Qa-gnn: Reasoning with language models and knowledge graphs for question answering. In *North American Chapter of the Association for Computational Linguistics (NAACL)*, 2021.

[86] Luyang Huang, Lingfei Wu, and Lu Wang. Knowledge graph-augmented abstractive summarization with semantic-driven cloze reward. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 5094–5107, 2020.

[87] Boci Peng, Yun Zhu, Yongchao Liu, Xiaohe Bo, Haizhou Shi, Chuntao Hong, Yan Zhang, and Siliang Tang. Graph retrieval-augmented generation: A survey. *arXiv preprint arXiv:2408.08921*, 2024.

[88] Zulun Zhu, Tiancheng Huang, Kai Wang, Junda Ye, Xinghe Chen, and Siqiang Luo. Graph-based approaches and functionalities in retrieval-augmented generation: A comprehensive survey. *arXiv preprint arXiv:2504.10499*, 2025.

[89] 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*, pages 1082–1090, 2011.

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

[91] Gustavo Niemeyer. Geohash. *Geohash*, 2008.

[92] Guy M Morton. A computer oriented geodetic data base and a new technique in file sequencing. *IBM*, 1966.

[93] 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*, pages 43–52, 2015.

[94] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 conference of the North American chapter of the association for computational linguistics: human language technologies, volume 1 (long and short papers)*, pages 4171–4186, 2019.

[95] Aditya Grover and Jure Leskovec. node2vec: Scalable feature learning for networks. In *Proceedings of the 22nd ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 855–864, 2016.

A Dataset Documentation

All datasets presented in this paper are intended for academic research purposes, and their corresponding licenses are listed in this section. They are constructed from publicly available resources described below. In all cases, we perform anonymization by removing any personally identifiable information when appropriate. User IDs in the original data are replaced with auto-incremented ID numbers.

Download links. The datasets and tasks described in this paper are available for download from the following links:

- Dataset and task website: `https://redacted-for-double-blind-review/`
- Dataset documentation: `https://redacted-for-double-blind-review/`
- Code for dataset preparation: `https://redacted-for-double-blind-review/`
- Code for running experiments: `https://redacted-for-double-blind-review/`

Dataset source and license information. Below, we describe how the source data was obtained and provide license information for each dataset:

- `pres-github`. This dataset is based on GitHub data collected from the GH Archive website (`https://www.gharchive.org/`) using its HTTP JSON download link. It contains GitHub user activity from January 2025, and user IDs have been anonymized. Content from GH Archive is released under the CC-BY-4.0 license, while the associated code is released under the MIT license.
- `pres-amazon-clothing` and `pres-amazon-electronics`. These datasets contain Amazon product reviews and ratings in their respective categories. Both are based on Amazon review data collected by McAuley et al. [93] and hosted at: `https://cseweb.ucsd.edu/~jmcauley/datasets/amazon/links.html`. The Amazon review content is licensed under the Amazon license:

  > By accessing the Amazon Customer Reviews Library ("Reviews Library"), you agree that the Reviews Library is an Amazon Service subject to the Amazon.com Conditions of Use and you agree to be bound by them, with the following additional conditions:
  >
  > In addition to the license rights granted under the Conditions of Use, Amazon or its content providers grant you a limited, non-exclusive, non-transferable, non-sublicensable, revocable license to access and use the Reviews Library for purposes of academic research. You may not resell, republish, or make any commercial use of the Reviews Library or its contents, including use of the Reviews Library for commercial research, such as research related to a funding or consultancy contract, internship, or other relationship in which the results are provided for a fee or delivered to a for-profit organization. You may not (a) link or associate content in the Reviews Library with any personal information (including Amazon customer accounts), or (b) attempt to determine the identity of the author of any content in the Reviews Library. If you violate any of the foregoing conditions, your license to access and use the Reviews Library will automatically terminate without prejudice to any of the other rights or remedies Amazon may have.

- `pres-gowalla`. This dataset contains user activity on the (now defunct) social network Gowalla. It was originally collected by Cho et al. [89] using the platform's public API and published in the SNAP Dataset Repository [90] (`https://snap.stanford.edu/data/loc-Gowalla.html`). No specific license information is provided by the curator.
- `pres-brightkite`. This dataset contains user activity on the (also now defunct) social network Brightkite. It was also originally collected by Cho et al. [89] using the platform's public API and published in the SNAP Dataset Repository [90] (`https://snap.stanford.edu/data/loc-brightkite.html`). No specific license information is provided by the curator.

B Dataset Contents

Examples of dataset contents. To illustrate the structure of the curated datasets, we provide examples of user event sequences from several `pres` datasets. Each table includes both personal and relational events, showing how different types of user activity are represented in our format.

- `pres-brightkite` and `pres-gowalla`

Table 5: Example of user event sequence in `pres-brightkite` and `pres-gowalla`.

uid	timestamp	event_set	event	other_uid
39	1206596784	personal	9xj6hwkm	<NA>
39	1206596838	personal	9xj3fynm	<NA>
39	1206596871	personal	9xj3fynm	<NA>
39	1235862855	personal	9xj65423	<NA>
39	1250883230	personal	9xj65423	<NA>
39	1254535157	personal	9xj5skbn	<NA>
39	1254535193	personal	9xj5sm00	<NA>
39	1283443369	personal	9q8yyyhs	<NA>
39	<NA>	relational	friendship	0
39	<NA>	relational	friendship	30
39	<NA>	relational	friendship	105
39	<NA>	relational	friendship	1190

- `pres-amazon-clothing` and `pres-amazon-electronics`

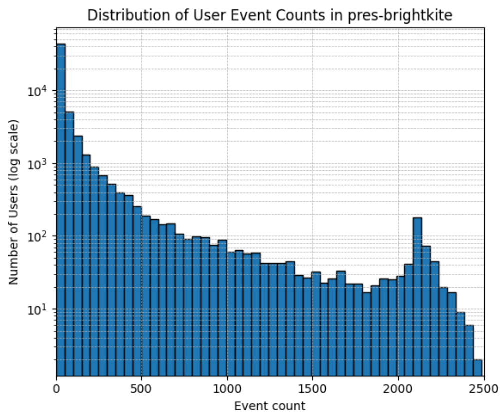
Table 6: Example of user event sequence in `pres-amazon-clothing` and `pres-amazon-electronics`.

uid	timestamp	event_set	event	other_uid
254057	1375401600	personal	B000A6PPOK:3	<NA>
254057	1377302400	personal	B003TMPHOU:5	<NA>
254057	1377302400	personal	B004A81PJI:4	<NA>
254057	1377302400	personal	B0054R4AXW:5	<NA>
254057	1377302400	personal	B005CPGHAA:5	<NA>
254057	1377302400	personal	B007FNXMEQ:5	<NA>
254057	1377302400	personal	B007IV7KRU:5	<NA>
254057	1377302400	personal	B007WAWHD4:5	<NA>
254057	1377302400	personal	B008AST7R6:5	<NA>
254057	1377302400	personal	B008R56H4S:5	<NA>
254057	1404086400	relational	co-review_product	107741

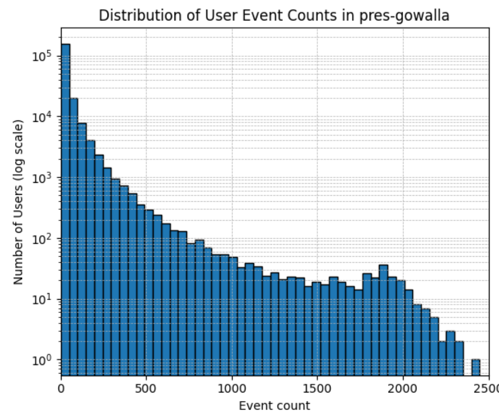
- `pres-github`

Table 7: Example of user event sequence in `pres-github`.

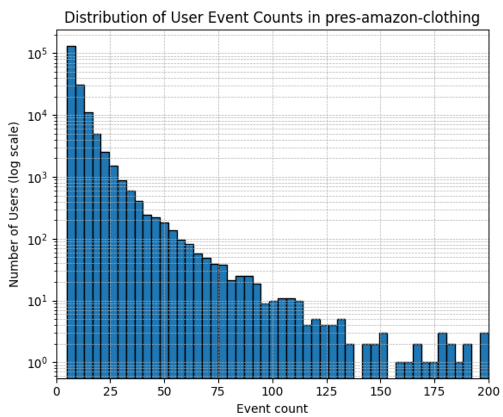
uid	timestamp	event_set	event	other_uid
3669059	1738288160	personal	PullRequestReviewCreated	<NA>
3669059	1738288191	personal	PullRequestReviewCreated	<NA>
3669059	1738288198	personal	PullRequestClosed	<NA>
3669059	1738288200	personal	Push	<NA>
3669059	1738288206	personal	PullRequestClosed	<NA>
3669059	1738288207	personal	Push	<NA>
3669059	1738288217	personal	DeleteBranch	<NA>
3669059	1738288219	personal	DeleteBranch	<NA>
3669059	<NA>	relational	collaborate	824409
3669059	<NA>	relational	collaborate	3126262



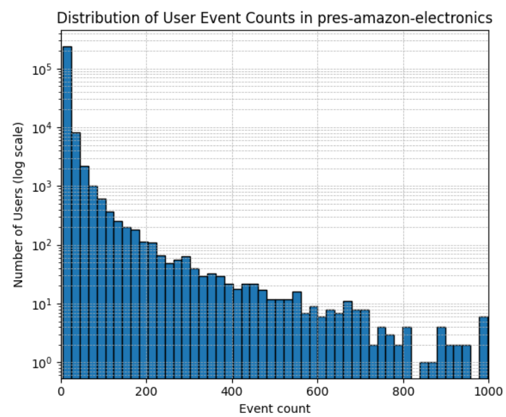
(a) `pres-brightkite`



(b) `pres-gowalla`



(c) `pres-amazon-clothing`



(d) `pres-amazon-electronics`

Figure 2: Histogram of the number of events per user in each dataset.

Event statistics. To characterize user events, we include histograms in Figure 2 and Figure 3 showing the distribution of event counts per user in each dataset. These histograms are constructed by computing the number of events associated with each user and aggregating how many users fall into each count bucket. The y-axis is log-scaled to highlight the long-tailed nature of user behavior, where the majority of users generate only a small number of events, while a much smaller group contributes disproportionately large volumes of activity. This skew is common across datasets and presents both challenges and opportunities for modeling.

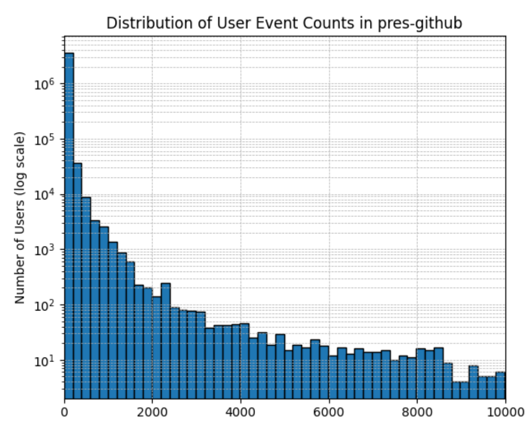


Figure 3: Histogram of the number of event per user in `pres-github`.

C Experiment Details

C.1 Hyperparameters

Personal event prediction task. In Table 8, we present the hyperparameters used during the training of various models for personal event prediction tasks. We use the following notations: **Emb Dim** denotes the dimensionality of token embeddings; **Heads** is the number of attention heads; **Layers**

Table 8: Hyperparameter configurations for personal event prediction tasks

Model Name	Learning Rate	Batch Size	Epochs	Emb Dim	Heads	Layers	Channels	Max Events	Max Examples	Num Neg Samples	Num Neighbors
BERT	3e-4	1024	10	64	4	4	–	100	50	10	–
BERT-n2v-p	3e-4	1024	10	64	4	4	–	100	50	10	–
BERT-n2v-i	3e-4	1024	10	64	4	4	–	100	50	10	–
GCN	1e-3	1024	10	128	–	2	128	100	–	5	10
GCN-PR	1e-3	1024	10	128	–	2	128	100	–	5	10
GAT	1e-3	1024	10	128	2	2	64	100	–	5	10
GAT-PR	1e-3	1024	10	128	2	2	64	100	–	5	10
TGN	1e-3	4096	10	16/32	–	–	–	100	–	5	10
TGN-PR	1e-3	4096	10	16/32	–	–	–	100	–	5	10
DyRep	1e-3	4096	10	32/64	–	–	–	100	–	5	10
DyRep-PR	1e-3	4096	10	32/64	–	–	–	100	–	5	10

refers to the number of hidden layers; **Channels** indicates the number of hidden channels per layer in GAT and GCN models; **Max Examples** is the maximum number of training samples generated per user; **Num Neg Samples** represents the number of negative samples for each (positive) sample; and **Num Neighbors** is the number of neighbors sampled per layer for GNN models. Additionally, due to GPU memory limitations, we reduce the embedding dimensions for the TGN and DyRep models to 16 and 32, respectively, for the `pres-brightkite` and `pres-gowalla` datasets, and to 32 and 64 for `pres-amazon-clothing` and `pres-amazon-electronics`.

Relational event prediction task. In Table 9, we present the hyperparameters used across all models for relational event prediction tasks. Due to memory and time constraints, batch size, number of epochs, and embedding dimensions were adjusted per dataset. All datasets used a batch size of 4096, except for `pres-github`, which used 512. The number of training epochs was set to 5 for `pres-github`, 20 for `pres-gowalla` and `pres-amazon-electronics`, 100 for `pres-amazon-clothing`, and 1000 for `pres-brightkite`. The model checkpoint with the best validation MRR was saved and used for testing. As shown in our results, TGN and DyRep could not be run on `pres-github`. For the remaining datasets, the embedding dimension for TGN and DyRep was 128, except for `pres-gowalla`, which used 64 to avoid GPU out-of-memory errors.

Table 9: Hyperparameter configurations for relational event prediction tasks

Model Name	Learning Rate	Batch Size	Epochs	Emb Dim	Heads	Layers	Channels	Num Neg Samples	Num Neighbors
GCN	1e-3	512/4096	5-1000	128	–	2	128	5	10
GCN-PR	1e-3	512/4096	5-1000	128	–	2	128	5	10
GCN+S	1e-3	512/4096	5-1000	128	–	2	128	5	10
GAT	1e-3	512/4096	5-1000	128	2	2	128	5	10
GAT-PR	1e-3	512/4096	5-1000	128	2	2	128	5	10
GAT+S	1e-3	512/4096	5-1000	128	2	2	128	5	10
TGN	1e-3	4096	20-1000	64/128	–	–	128	5	10
DyRep	1e-3	4096	20-1000	64/128	–	–	128	5	10

C.2 Computing Resources

We conducted all experiments on a server equipped with 8 NVIDIA Ampere A10G GPUs (24 GB each), 16 CPU cores, and a RAM upper limit of 512 GB. To fully leverage all resources, we parallelized the training runs so that each experiment used a single GPU. Each experiment is designed to be run on a single-GPU machine. Table 10 summarizes the average training time (in hours) and standard deviation for each model across five datasets, categorized by task type. For relational event prediction tasks, lightweight GCN and GAT variants exhibit minimal computational overhead, with training times generally under one hour except on the GitHub dataset. In contrast, temporally expressive models such as TGN and DyRep incur significantly higher costs, especially on large-scale datasets like Gowalla. In personal event prediction tasks, training times increase across the board,

Table 10: Computational Time (in hours) for Different Models and Datasets

Method	Time (h)				
	`amazon-clothing`	`amazon-electronics`	`brightkite`	`gowalla`	`github`
Relational event prediction tasks					
GCN	0.06±0.00	0.05±0.00	0.60±0.00	0.26±0.00	8.38±0.11
GCN-RP	0.10±0.01	0.17±0.00	0.40±0.00	1.98±0.03	7.39±0.06
GCN+S	0.07±0.00	0.05±0.00	0.61±0.00	0.29±0.00	8.58±0.12
GAT	0.07±0.01	0.08±0.02	0.61±0.00	0.29±0.00	8.41±0.12
GAT-RP	0.15±0.03	0.18±0.01	0.49±0.06	2.07±0.02	7.40±0.06
GAT+S	0.09±0.02	0.05±0.00	0.62±0.00	0.32±0.00	2.52±0.20
TGN	0.49±0.03	0.32±0.00	1.06±0.01	4.62±0.10	–
DyRep	0.49±0.02	0.31±0.00	1.03±0.01	4.43±0.07	–
Personal event prediction tasks					
GCN	5.81±0.10	7.14±0.29	1.73±0.02	8.01±0.71	–
GCN-PR	5.80±0.11	7.21±0.29	1.73±0.03	7.45±1.82	–
GAT	5.83±0.10	7.17±0.28	1.73±0.03	7.33±1.82	–
GAT-PR	5.82±0.10	7.24±0.30	1.76±0.02	7.51±1.79	–
TGN	3.94±0.40	4.10±0.10	0.33±0.01	3.12±0.75	–
TGN-PR	4.38±0.39	5.75±0.19	0.81±0.03	7.89±1.78	–
DyRep	2.03±0.38	3.23±0.34	0.38±0.01	4.11±1.03	–
DyRep-PR	4.88±0.10	5.96±0.20	0.78±0.03	7.85±1.86	–
BERT	4.67±0.06	6.30±0.15	2.65±0.02	9.21±1.11	–
BERT+n2v-i	3.41±0.14	4.43±0.19	2.54±0.01	6.40±0.19	–
BERT+n2v-p	3.60±0.22	4.78±0.14	2.52±0.01	6.38±0.20	–

with most models exceeding 7 hours on larger datasets, again highlighting the computational demands of modeling fine-grained temporal dynamics.

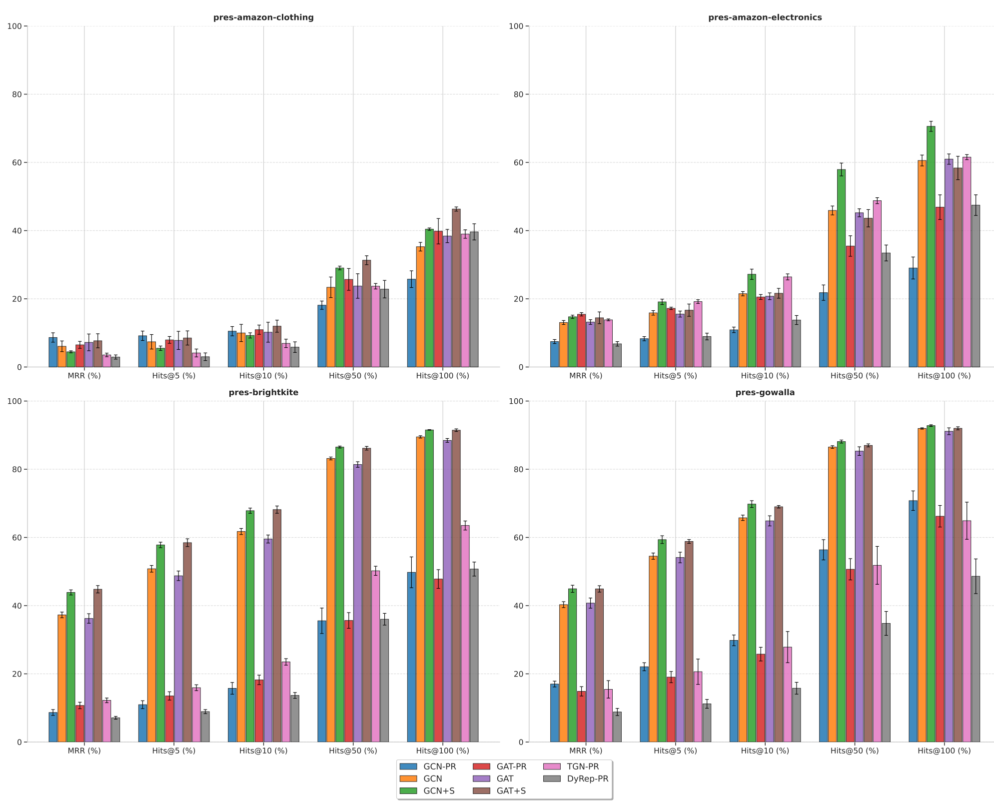


Figure 4: Comparison of relational event predictions across different datasets.

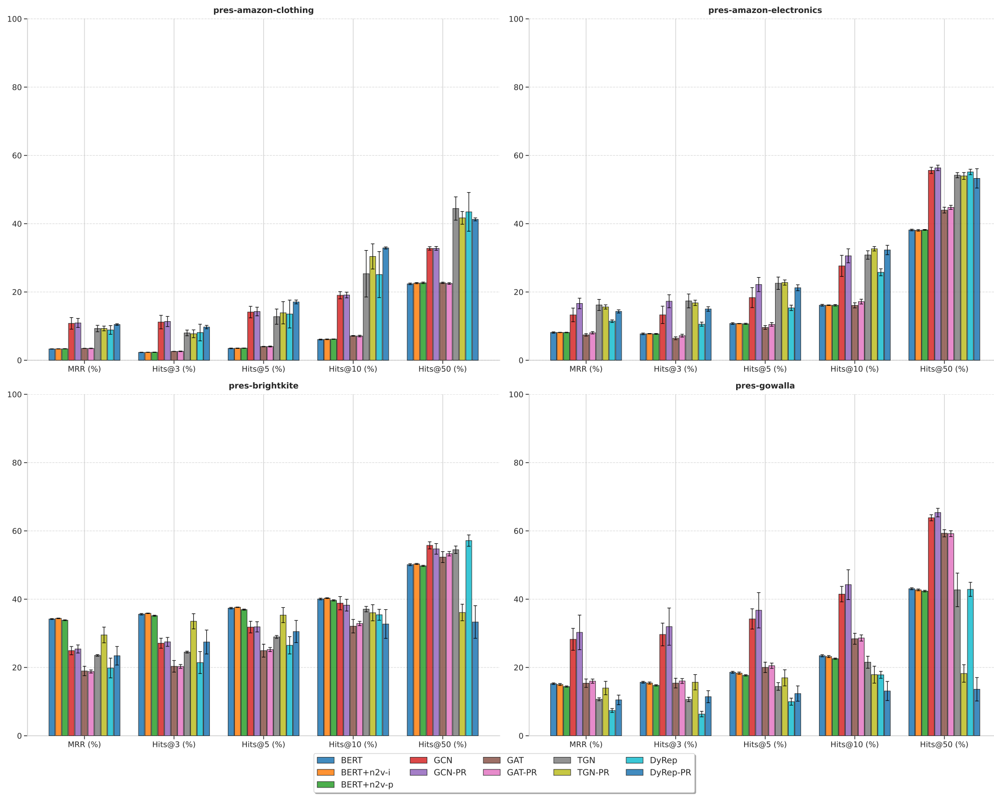


Figure 5: Comparison of personal event predictions across different datasets.

D Additional Experimental Results

In Figures 4 and 5, we present the results from the main paper in a more visual format to facilitate comparison across methods. In the relational event prediction tasks, across all datasets and metrics, static GNNs augmented with personal event sequence embeddings (GCN+S and GAT+S) consistently achieve the best or second-best results. This highlights the benefit of integrating both personal and relational signals. Temporal methods (TGN-PR, DyRep-PR) underperform, particularly when personal event nodes are included. For personal event prediction tasks, BERT+n2v-i offers slight improvements over regular BERT. In particular, BERT-based models exhibit competitive performance in some cases, most prominently on the Brightkite dataset, where they outperform GNN-based counterparts at MMR and lower hit rate thresholds such as Hits@3, Hits@5, and Hits@10.

E Broader Impacts

Broader impact of our paper The datasets and prediction tasks we release may support future research on user event modeling, particularly in settings that involve both personal and relational events. Researchers can build models on top of these resources and evaluate them in a consistent way. This can help accelerate empirical progress and facilitate more comparable results. This has potential impact in a range of industry applications where modeling user behavior is critical, such as recommendation, fraud detection, and user interaction analysis.

Potential negative impact The datasets we release may not cover all use cases of user event modeling, and may reflect only a subset of real-world scenarios. This could introduce bias in model development or evaluation, especially if models are tuned specifically for the structure or properties of our datasets. As a result, there is a risk that future methods may overfit to our datasets and generalize less effectively to other domains or applications.

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: We empirically verify the main claims stated in the abstract and introduction (summary of contributions) through the experiments presented in the experiments section. In addition, we openly release our datasets and formalization (used in the dataset format) as part of our contributions.

   Guidelines:

   - The answer NA means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A No or NA answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: We discuss the limitation of our work in the Conclusions and Limitations section.

   Guidelines:

   - The answer NA means that the paper has no limitation while the answer No means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

Answer: [NA]

Justification: The paper does not include theoretical results.

Guidelines:

- The answer NA means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

Answer: [Yes]

Justification: We provide several components to ensure the reproducibility of our results. First, we include pre-computed negative samples for the validation and test sets for every task and dataset. We also provide experiment details in both the Experiments section and Appendix C. Lastly, we release the code for running all experiments.

Guidelines:

- The answer NA means that the paper does not include experiments.
- If the paper includes experiments, a No answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
- If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
- Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
- While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
  (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
  (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
  (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
  (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in

some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

   Answer: [Yes]

   Justification: We release the datasets and the experiment codes.

   Guidelines:

   - The answer NA means that paper does not include experiments requiring code.
   - Please see the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - While we encourage the release of code and data, we understand that this might not be possible, so "No" is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
   - The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
   - The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
   - At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
   - Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer, etc.) necessary to understand the results?

   Answer: [Yes]

   Justification: The experiment details are available in the Experiments section and Appendix C.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [Yes]

   Justification: We report the mean and standard deviation of our experimental results in all result tables in the paper.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The authors should answer "Yes" if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.

- The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
- The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
- The assumptions made should be given (e.g., Normally distributed errors).
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g. negative error rates).
- If error bars are reported in tables or plots, The authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

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

   Answer: [Yes]

   Justification: The information about computing resources is available in Appendix C

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification: We have reviewed the ethic code and make sure our research conform with the code of ethics.

   Guidelines:

   - The answer NA means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer No, they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [Yes]

    Justification: We discuss the broader impacts in Appendix E

    Guidelines:

    - The answer NA means that there is no societal impact of the work performed.

- If the authors answer NA or No, they should explain why their work has no societal impact or why the paper does not address societal impact.
- Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pretrained language models, image generators, or scraped datasets)?

    Answer: [NA]

    Justification: We do not identify any potential misuse risks associated with our paper, code, or datasets.

    Guidelines:

    - The answer NA means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
    - Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
    - We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [Yes]

    Justification: We discuss the license of the source data in Appendix A

    Guidelines:

    - The answer NA means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.
    - The name of the license (e.g., CC-BY 4.0) should be included for each asset.
    - For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.

- If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
- For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [Yes]

    Justification: Our pre-processed datasets are derived from publicly available source data, and we discuss their licenses in Appendix A. Our code for processing the datasets and running experiments is released under the MIT license.

    Guidelines:

    - The answer NA means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [NA]

    Justification: The paper does not involve crowdsourcing nor research with human subjects.

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
    - According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [NA]

    Justification: The paper does not involve crowdsourcing nor research with human subjects.

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.

- Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
- We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
- For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

    Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does not impact the core methodology, scientific rigorousness, or originality of the research, declaration is not required.

    Answer: [NA]

    Justification: The core method development in this research does not involve LLMs in any important, original, or non-standard components. We use LLMs only as writing and coding assistants.

    Guidelines:

    - The answer NA means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
    - Please refer to our LLM policy (`https://neurips.cc/Conferences/2025/LLM`) for what should or should not be described.