Neural Temporal Walks: Motif-Aware Representation Learning on Continuous-Time Dynamic Graphs

Anonymous Author(s)
Affiliation
Address
email

Abstract

Continuous-time dynamic graphs naturally abstract many real-world systems, such as social and transactional networks. While the research on continuous-time dynamic graph representation learning has made significant advances recently, neither graph topological properties nor temporal dependencies has been well-considered and explicitly modeled in capturing dynamic patterns. In this paper, we introduce a novel method, *Neural Temporal Walks* (NeurTWs), for representation learning on continuous-time dynamic graphs. By considering not only time constraints but also structural and tree traversal properties, NeurTWs conducts spatiotemporal-biased random walks to retrieve a set of representative motifs, enabling temporal nodes to be characterized effectively. With a component based on neural ordinary differential equations, the extracted motifs allows for irregularly-sampled temporal nodes to be embedded explicitly over multiple interaction time intervals, enabling the capture of the underlying spatiotemporal dynamics. To enrich supervision signals, we further design a harder contrastive pretext task for model optimization. Our method demonstrates overwhelming superiority under both transductive and inductive settings on three real-world datasets.

1 Introduction

Continuous-time dynamic graphs (CTDGs) consist of temporal events with respect to nodes (e.g., node addition/deletion) and edges (i.e., temporal interactions), which naturally arise in many real-world systems such as social networks and knowledge graphs [11, 29]. Traditional studies in dynamic graph modeling extract expressive patterns manually that is beneficial for understanding the crucial laws behind [33]. For example, two people are likely to know each other if they have a common friend (Fig. 1). Such a dynamic graph motif describes how social connections are established [28]. Other expressive patterns such as feedforward control loops have also been investigated [18]. However, extracting motifs manually is expensive, time-consuming and requires domain knowledge, thus hinders learning dynamic graphs with more complicated laws [33].

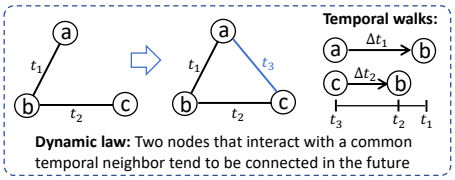


Figure 1: Temporal walks capture the law

The advent of graph neural networks (GNNs) makes it possible to understand more complicated graphs by automatically learning the laws behind. While GNNs have demonstrated great success in modeling static graphs, the research on dynamic graphs is still nascent. Current research on dynamic graph neural networks (DGNNs) faces two fundamental challenges. **Firstly**, the entangled spatial and temporal dependencies in real-world CTDGs typically need a specific design to model, preventing the direct use of off-the-shelf GNNs. To overcome this barrier, previous works simplify CTDGs to a series of static graph snapshots

Submitted to 36th Conference on Neural Information Processing Systems (NeurIPS 2022). Do not distribute.

with *uniform time intervals* (i.e., discrete-time dynamic graphs, DTDGs) [21, 26]. However, this approximation compromises the modeling precision. Although [13] and [31] propose to learn on CTDGs directly, the inductiveness of the patterns they captured is not guaranteed because node identities are directly involved in their modeling process. Some recent methods [35, 24, 33] attempt to alleviate this issue. However, they only consider time but not topological and tree traversal properties when sampling temporal neighbors, limiting their ability to extract diverse and expressive patterns from dynamic systems. **Secondly**, temporal events in CTDGs occur irregularly (e.g., nodes a and c interact with b at irregular timestamps in Fig. 1), resulting in a significant challenge in modeling temporal dependencies. Previous works typically bypass this challenge with the time encoding [35] to enable the use of message passing [35, 24] or sequence models [13, 33]. However, time encoding hurts model performance because temporal dependencies are modelled implicitly.

To tackle the above challenges, we propose the *Neural Temporal Walks* (`NeurTWs`) method for representation learning over CTDGs, which extracts and encodes representative dynamic graph motifs composed of irregularly-sampled temporal nodes. Specifically, motivated by [33], we propose *sptaiotemporal-biased random walks* to extract diverse and expressive patterns from a CTDG by not only considering time constrains but also topological properties and tree traversals, allowing the sampler to be better aware of the importance of temporal neighbors while maintaining the exploration and exploitation trade-off. To explicitly model temporal dependencies and capture essential dynamic laws more effectively, the extracted motifs consisting of irregularly-sampled temporal nodes are encoded with the proposed *continuous evolution* and *instantaneous activation* processes to learn time-variant node representations. The former process learns latent spatiotemporal dynamics across multiple interaction time intervals with an ordinary differential equation (ODE) function, which is regularized by the second process with those irregularly-sampled observations (temporal nodes).

On this basis, we acquire time-aware node embeddings by retrieving and encoding neighboring representative dynamic graph motifs, where a contrastive objective can be naturally designed to optimize `NeurTWs` by maximizing the mutual information between interacting temporal nodes. Compared with most existing works that use a simple link prediction objective, this harder contrastive pretext task helps enrich supervision signals, thus lifting the learning ability of `NeurTWs`.

On two benchmark datasets and a new dense e-commerce dataset, our method significantly and consistently outperforms all state-of-the-art methods. Specifically, it surpasses the strongest baselines by around 4% and 5% in all transductive and five out of six inductive tasks. In summary, our technical contributions are three-fold: **(1)** We propose novel spatiotemporal-biased random walks to extract diverse and expressive patterns from CTDGs by considering not only time constraints but also topological and tree traversal properties. **(2)** We introduce a new perspective on encoding dynamic graph motifs composed of irregularly-sampled temporal nodes, explicitly and better modeling the underlying spatiotemporal dynamics. **(3)** We integrate contrastive learning into dynamic graph modeling to enrich supervision signals, which lifts the learning ability of our model.

2 Related Work

Dynamic graph neural networks (DGNNs) can be broadly classified into two categories based on their inputs. Discrete-time DGNNs operate over a sequence of evenly-sampled static graph snapshots, where different strategies are proposed to model spatial and temporal clues, e.g., combining GNNs with sequence models [27, 21, 3, 26]. Our work relates to the second category, continuous-time DGNNs, where time-dependent node or edge embeddings are learned directly on CTDGs. Among these works, an in-demand design updates latent node states by aggregating k-hop neighborhood information with temporal message passing. For instance, TGAT [35] samples a set of k-hop temporal neighbors and proposes trainable time encodings to preserve the time information during message passing. TGN [24] further equips TGAT with a node memory updater as in [13]. Another line of research leverages random walks to learn on CTDGs. Specifically, CTDNE [20] is the first to propose a CTDG embedding method with temporal walks. CAW [33] further extends it with anonymous temporal walks and uses a recurrent net to learn walk embeddings that are aggregated when computing interactive representations. Our method is different from prior temporal walk-based approaches in three aspects: (1) We propose a new perspective in sampling temporal walks. While prior arts only consider time constraints, `NeurTWs` leverages multidimensional information, allowing our model to explore diverse and expressive patterns from CTDGs; (2) We propose a novel and intuitive motif embedding method to model latent spatiotemporal dynamics among irregularly-sampled temporal

nodes on CTDGs and without using recurrent nets with time encodings, which allows temporal dependencies to be modeled explicitly; (3) We replace a common classification-based dynamic graph learning protocol with a harder contrastive pretext task, which provides stronger supervision signals.

Neural ordinary differential equations (NODEs) were proposed in [2] as a new paradigm that generalizes discrete deep neural networks by parameterizing the derivative of latent states. This concept has been applied in research areas including time series forecasting [10, 25] and computer vision [7, 1]. Recently, some works extend NODEs to the graph domain, where most of them consider building deeper GNNs while alleviating the negative impacts of over-smoothing [34, 22]. Notably, the time information is absent among those works. In dynamic graph learning, most ODE-based works focus on discrete-time settings [4, 6], and only a few extend NODEs to learn on CTDGs [5]. In this paper, inspired by the research on time series forecasting [25], we propose `NeurTWs` to encode extracted motifs with irregularly-sampled temporal nodes on CTDGs, which is fundamentally different from [5]: `NeurTWs` directly integrates over multiple interaction time intervals to explicitly model the latent spatiotemporal dynamics across different temporal nodes with an ODE function, while [5] relies on a time encoding-assisted message passer to learn from historical temporal events.

Graph contrastive learning has achieved great success in graph self-supervised learning [17]. Most existing works are on static graphs [32, 9, 38]. While some studies have explored the possibility of dynamic graph contrastive learning, many of them are on DTDGs. For example, STGCL [16] enhances the model's forecasting ability with DTDG augmentations and an auxiliary contrastive loss. A similar design also exists in [37, 36] and [23]. On CTDGs, Tian *et al.* [30] proposed to maximize the time-dependent agreement between time-aware node embeddings at different time points. In DySubC [8], the mutual information between a node and its surrounding temporal subgraphs is maximized. Different from these works, we design an effective pretext task for model optimization, where the mutual information between two nodes in an interaction is maximized. Meanwhile, we push nodes away in the embedding space if there are no temporal interactions between them.

3 Problem Formulation

We start by formally introducing the learning problem on CTDGs. A complete notation table is in Appendix A. This paper defines a CTDG as a stream of temporal interactions, i.e., $\mathcal{G} = \{(e_i, t_i)\}_{i=1}^{N}$, where each interaction has two nodes at a specific time, e.g., $(e_i, t_i) \coloneqq (\{u_i, v_i\}, t_i),\ t_i \in \mathbb{R}^+$. As many real-world CTDG datasets are unattributed and for simplicity, we first assume these temporal interactions are without node and edge attributes and later we will discuss how our method is extended to learn on attributed CTDGs. Facing the challenge of lacking label information, DGNNs are typically supervised by temporal interactions [39]. Thus, dynamic link prediction is a widely adopted testbed to evaluate how accurate a DGNN is in predicting future interactions with the observation of historical events. Specifically, Given two nodes u and v at time t in $\mathcal{G}$, we aim to learn their time-dependent embeddings $\overline{h}_u$ and $\overline{h}_v$, where the presence of an interaction between them can be predicted with a downstream classifier, i.e., $\widehat{y}_{u,v,t} = \text{clf}(\overline{h}_u, \overline{h}_v)$. The ground truth $y_{u,v,t} = 1$ if there exists an interaction between u and v at time t otherwise $y_{u,v,t} = 0$.

4 The Proposed Method: Neural Temporal Walks

4.1 Preliminaries: Temporal Walks and Dynamic Graph Motifs

Given a dynamic graph $\mathcal{G}$, we define a motif as a subset of temporal nodes with their interactions within a defined time range [12], e.g., $0 \leq t \leq q$. As those motifs reflect certain dynamic laws in a CTDG, it is desirable to characterize a temporal node with its surrounding motifs.

Definition 4.1.1 (Temporal Walk). Given a dynamic graph $\mathcal{G}$, we denote the interactions that are directly associated with a node u before a cut time t as $\mathcal{G}_{u,t} = \{(e, t') \mid t' < t,\ u \in e\ ,\ (e, t') \in \mathcal{G}\}$. A (time-reversed) temporal walk rooted at node u at time t is defined as W, which is a sequence of temporal nodes as in [33], i.e., node w_i at time t_i with $w_0 \coloneqq u$ and $t_0 \coloneqq t$:

$$W = \{(w_i, t_i) \mid i \in \mathbb{N}^+,\ 0 \leq i \leq l,\ t_0 > t_1 > \cdots > t_l,\ (\{w_{i-1}, w_i\}, t_i) \in \mathcal{G}_{w_i,t_i} \text{ for } i \geq 1\}, \quad (1)$$

where l is the walk length. We also use $W[i][0]$ and $W[i][1]$ (i.e. w_i and t_i in (w_i, t_i) respectively) to denote the specific node and time in the i-th step.

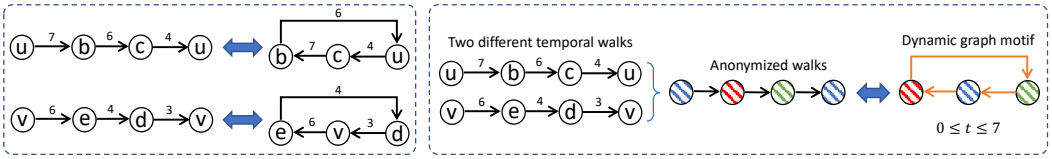
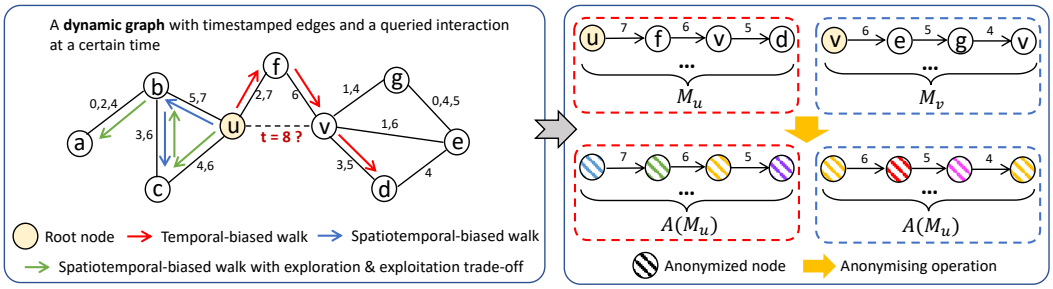


Figure 2: Quaternary closures and the dynamic graph motifs: Two example temporal walks form two different quaternary closures but represent the same motif within the time range $0 \le t \le 7$.



(a) Dynamic Graph Motifs Extraction (b) Walks Anonymization

Figure 3: Temporal walks sampling and anonymization. Given a CTDG and a queried interaction $(u, v, 8)$, we first extract surrounding temporal motifs by sampling a set of expressive temporal walks started from u and v, respectively, denoted as M_u and M_v. Then, the walks in two sets are anonymized by replacing nodes' identities with their position encodings.

Each walk W rooted at a temporal node can actually be regarded as one of its surrounding motifs if $t_1 - t_l$ is bounded within the defined motif time range. A concrete example is given in Fig. 2, where two walks in the leftmost side form two quaternary closures, which essentially represent the same motif on the rightmost side after an individual walk anonymization [19]. The necessity of anonymization is that it replaces original node identities in walks with their relative identities, which maps each walk to a particular pattern and thus connects temporal walks and dynamic graph motifs.

Definition 4.1.2 (Anonymous Walk). Given a temporal node w and a walk W, the anonymization operation $A(\cdot)$ is defined as follows [33]:

$$A(w; W) = |\{v_0, \cdots, v_{i^*} \mid v_i \in W\}|, \text{where } i^* \text{ is the smallest index s.t. } v_{i^*} = w. \tag{2}$$

4.2 Dynamic Graph Motif Extraction

Temporal walk sampling Existing walk-based methods mainly employ a temporal-biased sampling method when extracting dynamic graph motifs [20, 33]. Specifically, given a node u at time t, the probability of its neighbor a in $(\{a, u\}, t') \in \mathcal{G}_{u,t}$ to be sampled is proportion to $exp(\alpha(t' - t))$, which discounts stale neighbors and tends to sample more current nodes with timestamps closer to t. A larger α emphasizes more on this bias. Although more current neighbors are more likely to be informative, the underlying topological and tree traversal properties are not respected, which hinders the extraction of diverse and expressive patterns. Here, we propose *spatiotemporal-biased random walks* with the exploitation and exploration trade-off.

Our motivations are twofold: (1) Most-recent neighbors should be allocated a larger sampling probability $\propto exp(\alpha(t' - t))$ as they are typically more informative w.r.t. a root node at time t. In Fig. 3(a), given a root node u and its two temporal neighbors f and c, a temporal-biased sampling path is more likely to be $u \to f$ instead of $u \to c$; (2) Neighbors with higher connectivity need to be emphasized to allows exploring more diverse and potentially expressive motifs. Given a node a at time t', we use its degree $d_a = |\mathcal{G}_{a,t'}|$ to reify its connectivity, thus the proposed spatial-biased probability $\propto exp(-\beta/d_a)$ with a hyperparameter β to control the bias intensity. Alg. 1 illustrates the walk sampling procedures with the above considerations. Given a nodes u at time t, the probability of its temporal neighbor a to be sampled is the average of the following normalized probabilities:

$$Pr_t(a) = \frac{exp(\alpha(t_a - t))}{\sum_{a' \in \mathcal{G}_{u,t}} exp(\alpha(t_{a'} - t))} \tag{3} \qquad Pr_s(a) = \frac{exp(-\beta/d_a)}{\sum_{a' \in \mathcal{G}_{u,t}} exp(-\beta/d_{a'})} \tag{4}$$

Algorithm 1 Sampling Temporal Walks

Require: root node w_0, cut time t_0, $\mathcal{G}$, C, l

Initialize $\{W_c = \big((w_0, t_0)\big) \mid 1 \leq c \leq C\}$
for i in $1, 2, \cdots, l$ **do**
  for j in $1, 2, \cdots, C$ **do**
    $w_p, t_p := W_j[i][0], W_j[i][1]$
    Initialize $d_w = 0$ for all $w \in \mathcal{G}_{w_p,t_p}$
    for (e, t) in $\mathcal{G}_{w_p,t_p}$ **do**
      Let $e := \{w, w_p\}$, $d_w = |\mathcal{G}_{w,t}|$
    end for
    Sample one $(e, t) \in \mathcal{G}_{w_p,t_p}$ with prob. $\propto exp(\alpha(t - t_p))$ and $exp(-\beta/d_w)$
    Let $e := \{w, w_p\}$, $W_c = W_c || (w, t)$
  end for
end for
return $\{W_c \mid 1 \leq c \leq C\}$

Although Alg. 1 complements the temporal-biased schema by considering the additional topological information, it may overly encourage the depth-first search (DFS), which could misleadingly sample many homogeneous motifs with a limited budget. Take an extreme example, if $|M_u|$ is restricted to 3 in Fig. 3(a), paths $u \to b \to c \to u$ may be sampled three times, leaving no room to explore $u \to c \to b \to a$ and $u \to f \to v \to d$. Thus, we design an exploitation & exploration trade-off to regularize the walk sampling with another probability:

$$Pr_e(a) = \frac{exp(-\gamma s_a)}{\sum_{a' \in \mathcal{G}_{u,t}} exp(-\gamma s_{a'})}, \tag{5}$$

where s_a and γ denotes the traversal times of node a and the intensity of this regularization.

Our complete walk sampling algorithm and its complexity analysis are in Appendix B.1 and B.3. In a nutshell, given a temporal node, the probability of its neighbor to be sampled is the average of the probabilities defined in Eq. 3, 4 and 5. In `NeurTWs`, given a queried interaction between two temporal nodes u and v as shown in Fig. 3, we sample a set of C temporal walks rooted at each node with the length l, denoted as M_u and M_v.

Anonymization Walk anonymization replaces node identities with position encodings (relative identities), which injects structural information while maintaining the inductiveness of `NeurTWs`. A drawback of Eq. 2 is that the position encoding of each node only depends on its specific walk, leading anonymous walks rooted at the same node sharing different name spaces [33]. Thus, we consider two practical solutions: *unitary* and *binary anonymization* to alleviate this problem. For a temporal node w in at least one walk rooted at node u, its unitary anonmization w.r.t. u considers the name space defined over M_u, the set of walks rooted at u:

$$A(w; M_u)[i] = |\{W \mid w = W[i][0],\ W \in M_u\}|, \text{where } i \in \{0, \cdots, l\}. \tag{6}$$

In Eq. 6, the identity of w is replaced by a vector $A(w; M_u)$ with length l, where the i-th element counts the number of walks that have node w appearing in position i.

While unitary anonymization anonymizes node w w.r.t. M_u as $A(w; M_u)$, for interacting root nodes u and v, $A(w; M_u)$ and $A(w; M_v)$ belong to different name spaces. Since DGNNs are typically supervised by temporal interactions, establishing the correlations between $W \in M_u \cup M_v$ may be beneficial. Thus, the binary anonymization is defined as follows [33]:

$$A(w; M_u, M_v) = A(w; M_u) \mid\mid A(w; M_v), \tag{7}$$

where $||$ denotes the concatenation operation to establish the connections among motifs in M_u and M_v. In the rest of the paper, we abbreviate the two anonymization strategies as $A(w)$ for simplicity and denote $\widehat{W} = \{\big(A(w_i), t_i\big) \mid (w_i, t_i) \in W \text{ for } i = 0, 1, \cdots, l\}$ as an anonymous temporal walk.

4.3 Neural Temporal Walk Encoding

A significant challenge to model CTDGs is that interactions occur irregularly. Previous work typically bypasses this challenge by concatenating node attributes with extra time encodings when aggregating the neighboring information [35, 24, 33], where temporal dependencies are modelled implicitly. A detailed discussion is in Appendix D.2. To encode a dynamic graph motif with irregularly-sampled temporal nodes, we explicitly integrate over multiple interaction time intervals to learn the latent spatiotemporal dynamics with those discrete observations. Fig. 4 and Alg. 2 describe our method in a nutshell, which consist of two interleaving steps: *Continuous evolution* and *instantaneous activation*.

Continuous evolution Given a series of temporal nodes at different time, i.e., $(A(w_i), t_i) \in \widehat{W}$ and ensuring $t_{i-1} < t_i$ by reversing the order of elements in $\widehat{W}$, the latent spatiotemporal dynamics among those nodes are modeled as follows:

$$h_i^{'} = h_{i-1} + \int_{t_{i-1}}^{t_i} f(h_t, \theta)\, dt, \tag{8}$$

Algorithm 2 Neural Temporal Walk Encoding

Require: An anonymous temporal walk $\widehat{W} = \{(A(w_i), t_i) \mid (w_i, t_i) \in W \text{ for } i = 0, 1, \cdots, l\}$

Reverse the order of elements in $\widehat{W}$
$t_{-1} = t_0$, $h_{-1} = \mathbf{0}$
for i in $0, 1, 2, \cdots, l$ **do**
  $h_i^{'} = \text{ODESolve}(h_{i-1}, f_\theta, t_{i-1}, t_i)$
  $A^{'}(w_i) = \text{MLP}_\psi(A(w_i))$
  $h_i = g_\phi(h_i^{'}, A^{'}(w_i))$
end for
return The walk embedding h_l

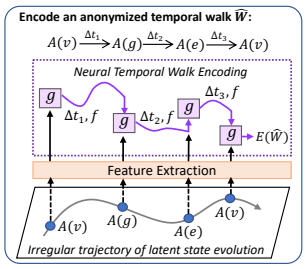


Figure 4: The spatiotemporal dynamics behind irregularly-sampled temporal nodes are modeled with `NeurTWs`.

where h_{i-1} is the latent states after encoding $(A(w_{i-1}), t_{i-1}) \in \widehat{W}$. We define the ODE function $f(h_t, \theta)$ as an autoregressive gated recurrent unit parameterized by θ. See Appendix B.2.

Instantaneous activation The latent state evolution in Eq. 8 conditions on a series of discrete observations. Thus, we define a function to activate latent state trajectories with instantaneous inputs:

$$h_i = g(h_i^{'}, A^{'}(w_i), \phi), \tag{9}$$

where $g(\cdot, \phi)$ can be a standard RNN cell parameterized by ϕ, and $A^{'}(w_i) = \text{MLP}(A(w_i), \psi)$ denotes the linear mapping of a discrete observation $A(w_i)$ in an anonymous temporal walk $\widehat{W}$.

Compared with [33], our method can straightforwardly model walks with irregularly-sampled temporal nodes, where the time information is explicitly reflected in `NeurTWs` encoding. On this basis, for a temporal node u, we obtain its time-dependent representation by pooling the embedding of walks in M_u, denoted as $\overline{h}_u$. In this paper, we simply use the sum pooling but a more advanced self-attention pooling [33] can also be applied.

4.4 Contrastive Optimization

A self-supervised pretext task is necessary to train DGNNs due to the scarcity of labeling information. Most current works formulate their learning problems as a binary classification task, where the existence of an interaction is predicted. From the contrastive point of view, a binary cross-entropy loss essentially forms a particular case of Jensen-Shannon estimator [17], where the number of negatives is one and thus provides limited supervision signals.

Here, we introduce a harder contrastive pretext task, where the mutual information between interacting temporal nodes (e.g., node u and v in Fig. 3) is maximized. We details the complete algorithm of our method in Appendix B.4. In brief, the following noise-contrastive loss is minimized:

$$\mathcal{L} = -\mathbb{E}\Big[log \frac{exp\big(\text{sim}(\overline{h}_u, \overline{h}_v)\big)}{exp\big(\text{sim}(\overline{h}_u, \overline{h}_v)\big) + \sum_{v' \in \mathcal{G}, v' \neq v} exp\big(\text{sim}(\overline{h}_u, \overline{h}_{v'})\big)}\Big]. \tag{10}$$

$\text{sim}(\cdot)$ measures the similarity between two entities, i.e., $\text{sim}(\overline{h}_u, \overline{h}_v) = \sigma\big(\text{MLP}(\overline{h}_u, \overline{h}_a, \xi)\big)$, where $\sigma(\cdot)$ and ξ are sigmoid activation and trainable parameters.

4.5 Extension and Discussion

Learning on attributed CTDGs Our method can be easily extended to model CTDGs with node and interaction attributes. To achieve this, we only need to slightly modify Eq. 9 as follows:

$$h_i = g(h_i^{'},\ A^{'}(w_i) \mid\mid X_{w_i} \mid\mid X_{e_i},\ \phi), \tag{11}$$

where X_{w_i} and X_{e_i} are linearly mapped features of node w_i and edge $e_i := \{w_{i-1}, w_i\}$ in $\widehat{W}$.

Batching and computational complexity One computational challenge is that each walk contains irregularly-sampled nodes at different times, requiring the model to separately solve C different ODEs to calculate the embedding of a root node, thus hindering training in batches. To solve all ODEs at once in a minibatch with B sampled interactions, we unify the integral time among all ODEs (see Appendix B.3) to a certain range, which results in a lower time complexity of $\mathcal{O}(l)$ instead of $\mathcal{O}(Bl)$. The detailed complexity analysis is in Appendix B.3.

Table 1: The dataset statistics. Average interaction intensity $\lambda = 2N/(|V|T)$ [14, 33] embodies the density of interactions in a fixed period, where T and $|V|$ are dataset duration and number of nodes.

Statistics	CollegeMsg	Enron	Taobao
# nodes & interactions	1,899 & 59,835	143 & 62,617	64,703 & 77,436
Duration (second)	16,621,303	72,932,520	36,000
# node & interaction attributes	0 & 0	0 & 0	0 & 4
Average interaction intensity λ	3.79×10^{-6}	1.2×10^{-5}	6.64×10^{-5}

5 Experiments

5.1 Experimental Setting

Baselines Our model is compared with five state-of-the-art methods in modeling CTDGs. They can be divided into two categories based on their intrinsic mechanisms: (1) message passing-based methods, including DyRep [31], TGAT [35], and TGN [24]; (2) sequential model-based methods, including JODIE [13] and CAWs [33]. More details can be found in Appendix C.2.

Datasets We evaluate model performance on three real-world datasets. CollegeMsg [15] is a social network dataset consisting of message sending activities. Enron [15] is an email communication network. Taobao [40] is an attributed user behavior dataset containing user-item interactions. The dataset statistics are summarized in Tab. 1 and their detailed descriptions are in Appendix C.1.

Evaluation protocols We follow the evaluation protocols of CAWs [33] and consider two different downstream tasks for evaluation: *transductive* and *inductive* link prediction.

In *transductive* link prediction, we sort and divide all N interactions in a dataset by time into three separate sets for training, validation, and testing. Specifically, the ranges of training, validation, and testing sets are $[0, N_{trn})$, $[N_{trn}, N_{val})$, $[N_{val}, N]$, where N_{trn}/N and N_{val}/N are 0.7 and 0.85.

For *inductive* link prediction, we use the same splits but mask a proportion of nodes (10%) and associated interactions during training, which are predicted during inference to evaluate the model inductiveness. Specifically, we first remove all interactions connected with masked nodes in the training set and then remove all interactions *not* associated with them in the validation and testing sets. Particularly, we have two specific settings: (1) *New-Old* tasks require the model to predict the interactions with one unobserved (i.e., masked) and one observed nodes; and (2) *New-New* tasks aim to predict the interactions between all unobserved nodes.

Training details We implement and train all models under a unified evaluation framework with the Adam optimizer. For hyperparameters, we mainly tune those related to walk sampling (e.g., α, β, γ in Eq. 3, 4 and 5) and contrastive learning (i.e., number of negatives). In solving ODEs, we use the Runge-Kutta 3/8 method [2] with the number of function evaluations set to 8 by default. For other general hyperparameters related to model training, we refer the settings in [33] for more details.

For fair comparisons and simplicity, we use sum-pooling when calculating node representations in both our method and CAWs. We also test `NeurTWs`†, which is equipped with the binary anonymization, while `NeurTWs` adopts the default unitary anonymization. All baselines are tuned thoroughly with a 2-layer perceptron to conduct downstream link prediction tasks, and we adopt the commonly used Area Under the ROC Curve (AUC) and Average Precision (AP) as the evaluation metrics. More implementation details can be found in Appendix C.3.

5.2 Results and Discussion

The performance of our method and baselines are summarized in Tab. 2 (w.r.t. AUC) and Appendix D.1 (w.r.t. AP). As can be seen from the table, our method achieves the best performance on different tasks and datasets. Notably, `NeurTWs` surpasses the state-of-the-art model, CAWs, significantly on both unattributed and attributed CTDGs: 7.76% and 4.78% on average on CollegeMsg and Taobao. Particularly, `NeurTWs` and `NeurTWs`† achieve a strong AUC result ≥ 0.95 on CollegeMsg, while all baselines have their AUCs ≤ 0.9, demonstrating the effectiveness of the proposed method.

Specifically, on both transductive and inductive tasks, we make the following observations. (1) Our method performs well on both attributed and unattributed datasets, while the effectiveness of baseline

Table 2: Transductive and inductive link prediction performances w.r.t. AUC. We use **bold font** and underline to highlight the best and second best performances. `NeurTWs`† is a vairant of our method with the binary anonymization.

Task		Methods	CollegeMsg	Enron	Taobao
Transductive		JODIE	0.5846 ± 0.038	0.8714 ± 0.011	0.8477 ± 0.015
		DyRep	0.5297 ± 0.042	0.8632 ± 0.013	0.8462 ± 0.012
		TGAT	0.7528 ± 0.004	0.6592 ± 0.012	0.5400 ± 0.005
		TGN	0.8990 ± 0.003	0.8944 ± 0.015	0.8484 ± 0.029
		CAWs	0.9002 ± 0.002	0.9520 ± 0.002	0.8719 ± 0.001
		`NeurTWs`	0.9526 ± 0.002	0.9564 ± 0.005	$\mathbf{0.9100} \pm \mathbf{0.014}$
		`NeurTWs`†	$\mathbf{0.9750} \pm \mathbf{0.004}$	$\mathbf{0.9704} \pm \mathbf{0.012}$	0.8911 ± 0.014
Inductive	New-Old	JODIE	0.4589 ± 0.028	0.8182 ± 0.022	0.7626 ± 0.002
		DyRep	0.4486 ± 0.021	0.7241 ± 0.025	0.7641 ± 0.012
		TGAT	0.7240 ± 0.008	0.6131 ± 0.049	0.5537 ± 0.018
		TGN	0.8699 ± 0.007	0.7068 ± 0.116	0.8706 ± 0.008
		CAWs	0.8911 ± 0.015	$\mathbf{0.9612} \pm \mathbf{0.002}$	0.8744 ± 0.004
		`NeurTWs`	0.9575 ± 0.011	0.9525 ± 0.002	$\mathbf{0.9316} \pm \mathbf{0.018}$
		`NeurTWs`†	$\mathbf{0.9699} \pm \mathbf{0.010}$	0.9566 ± 0.007	0.9037 ± 0.013
	New-New	JODIE	0.5135 ± 0.048	0.7537 ± 0.025	0.7791 ± 0.004
		DyRep	0.5813 ± 0.066	0.7184 ± 0.061	0.7716 ± 0.017
		TGAT	0.7283 ± 0.029	0.6340 ± 0.032	0.5479 ± 0.025
		TGN	0.7745 ± 0.102	0.9217 ± 0.026	0.8701 ± 0.011
		CAWs	0.8974 ± 0.009	0.9777 ± 0.001	0.8762 ± 0.004
		`NeurTWs`	0.9649 ± 0.008	$\mathbf{0.9906} \pm \mathbf{0.007}$	$\mathbf{0.9242} \pm \mathbf{0.005}$
		`NeurTWs`†	$\mathbf{0.9768} \pm \mathbf{0.008}$	0.9858 ± 0.015	0.9140 ± 0.013

models varies significantly. For instance, JODIE and DyRep has enormous performance gaps between Taobao and CollegeMsg in all three tasks. It indicates that our `NeurTWs` method is better at capturing essential dynamic laws. This superiority can be attributed to our spatiotemporal-biased random walks and the associated encoding mechanism; (2) Our method is effective under both transductive and inductive settings. On the contrary, some baseline models cannot well generalize to predict interactions with unseen nodes (e.g., the performance of JODIE and DyRep drops on inductive tasks) because they rely on node identities. TGAT, TGN, and CAWs can yield better performances but leave room to improve. Notably, the simpler unitary anonymization (i.e., `NeurTWs`) and the more complex binary anonymization (i.e., `NeurTWs`†) yield similar performance. Our method with the default, simpler unitary anonymization (i.e., `NeurTWs`) performs significantly better than CAWs, which utilizes the binary anonymization, again demonstrating the superiority of our approach in modeling CTDGs without relying on sophisticated position encodings. (3) Comparing the results on Taobao, our method's improvements over CAWs are more significant than on CollegeMsg, e.g., 7.48% vs. 3.81% on transductive and 7.91% vs. 5.26% on inductive settings. This is because the CollegeMsg dataset has a lower average interaction intensity ($\lambda = 3.79 \times 10^{-6}$) compared with Taobao dataset ($\lambda = 6.64 \times 10^{-5}$). A lower λ indicates a larger average time span in a temporal walk, for which our proposed continuous evolution process is more effective.

5.3 Ablation Study

In Tab. 3, we report the results of our ablation study on predicting *inductive* interactions in both new-old and new-new settings. Specifically, Ab.1, Ab.2, and Ab.3 remove Eq. 3, 4, and 5 when sampling temporal walks. Ab.4 disables Eq. 8 when encoding an anonymous walk. Ab.5 replaces our contrastive loss (Eq. 10) with a binary cross entropy loss. From the above variants, we see performances degradation when retrieving motifs without considering the temporal or spatial information, demonstrating the effectiveness of spatiotemporal-biased walk sampler. Moreover, disabling the exploitation & exploration trade-off also hurts performances. When removing the continuous evolution, more severe performance drops are found on CollegeMsg, further demonstrating that `NeurTWs` excels on CTDGs with more sparse interactions where existing works usually fail to model them well. The performance drops also exist when our contrastive pretext task is disabled, suggesting the superiority of our design in optimizing DGNNs. A specific study related to continuous evolution is in Appendix D.2.

Table 3: Ablation study with the proposed `NeurTWs` method. The performance in predicting *all inductive* interactions is reported.

No.	Configuration	CollegeMsg		Taobao	
		AUC	AP	AUC	AP
0	Full model	**0.958 ± 0.01**	**0.966 ± 0.01**	**0.938 ± 0.02**	**0.933 ± 0.02**
1	w/o T-biased probability	0.918 ± 0.02	0.928 ± 0.02	0.932 ± 0.03	0.927 ± 0.01
2	w/o S-biased probability	0.949 ± 0.02	0.958 ± 0.02	0.915 ± 0.01	0.915 ± 0.01
3	w/o E&E-biased probability	0.957 ± 0.01	0.965 ± 0.01	0.926 ± 0.01	0.927 ± 0.01
4	w/o continuous evolution	0.868 ± 0.02	0.898 ± 0.01	0.860 ± 0.05	0.901 ± 0.02
5	w/o contrasitve learning	0.954 ± 0.01	0.962 ± 0.01	0.935 ± 0.01	0.932 ± 0.01

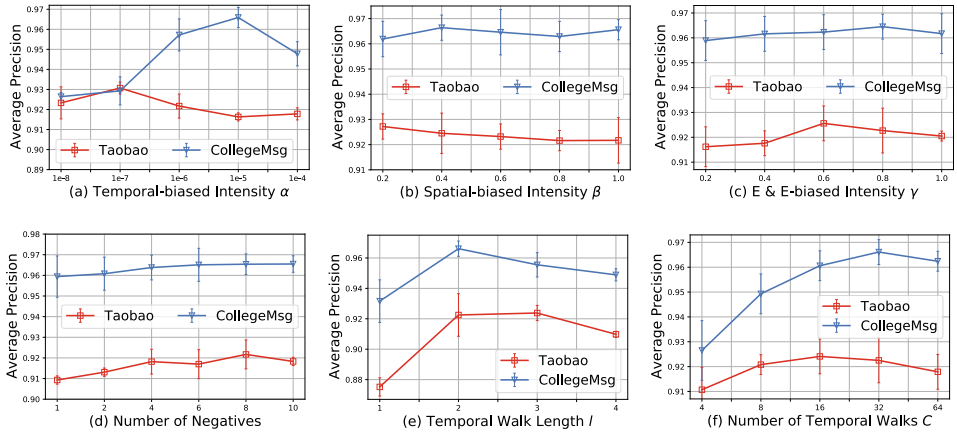


Figure 5: Hyperparameter sensitivity study on the proposed `NeurTWs` method. The performance in predicting *all inductive* interactions is reported.

5.4 Parametric Sensitivity

We study the primary hyperparameters in `NeurTWs`. Specifically, we have the following observations: (1) For each dataset, there exist sweet spots in balancing three intensities (i.e., α, β, and γ). Specifically, different datasets with different λ prefer different temporal-biased intensities (e.g., a relatively large α benefits on CollegeMsg but hurts the performance on Taobao as shown in Fig. 5(a)). On both datasets, overly emphasizing the topology information with a large β hurts the performance (Fig. 5(b)) because the temporal information is overly neglected. However, setting a relatively large γ seems beneficial (Fig. 5(c)), indicating the importance of balancing exploitation and exploration in temporal walk sampling. (2) We also investigate the choosing of walk length l and number of walks C (Fig. 5(e) and 5(f)). On both datasets, sampling 16 or 32 walks with the length 2 or 3 seems sufficient to characterize a temporal node. (3) For efficiency, we limit the number of negatives in Eq. 10 instead of calculating $\overline{h}_{v'}$ for all $v' \in \mathcal{G}$ where $v' \neq v$. Specifically, we find that a large number is usually beneficial on both datasets as shown in Fig. 5(d), where the performance increases to converge with an increasing number of negatives. In our method, we normally do not have to tune continuous evolution-related hyperparameters, but we also provide a study in Appendix D.2.

6 Conclusion

We propose a novel and competitive method for modeling CTDGs, which has advantages in three aspects. Firstly, `NeurTWs` complements existing temporal neighborhood sampling algorithms by simultaneously considering temporal, topological, and tree traversal properties, enabling diverse yet expressive dynamic graph motifs to be retrieved. Secondly, `NeurTWs` allows these motifs with irregularly-sampled temporal nodes to be better encoded, where temporal dependencies are now explicitly modeled. Thirdly, `NeurTWs` forms a harder contrastive pretext task to enrich supervision signals. Consequently, `NeurTWs` outperforms SOTA methods by large margins, revealing significant application prospects in many industrial systems, which will be an exciting focus in future work.

References

[1] Eric Z Chen, Terrence Chen, and Shanhui Sun. Mri image reconstruction via learning optimization using neural odes. In *International Conference on Medical Image Computing and Computer-Assisted Intervention*, pages 83–93. Springer, 2020.

[2] Ricky T. Q. Chen, Yulia Rubanova, Jesse Bettencourt, and David Duvenaud. Neural ordinary differential equations. *Advances in Neural Information Processing Systems*, 2018.

[3] Palash Goyal, Sujit Rokka Chhetri, and Arquimedes Martinez Canedo. Capturing network dynamics using dynamic graph representation learning, March 5 2020. US Patent App. 16/550,771.

[4] Zhen Han, Zifeng Ding, Yunpu Ma, Yujia Gu, and Volker Tresp. Temporal knowledge graph forecasting with neural ode. *arXiv preprint arXiv:2101.05151*, 2021.

[5] Zijie Huang, Yizhou Sun, and Wei Wang. Learning continuous system dynamics from irregularly-sampled partial observations. *Advances in Neural Information Processing Systems*, 33:16177–16187, 2020.

[6] Zijie Huang, Yizhou Sun, and Wei Wang. Coupled graph ode for learning interacting system dynamics. In *27th ACM SIGKDD Conference on Knowledge Discovery and Data Mining, KDD 2021*, pages 705–715, 2021.

[7] Boyan Jiang, Yinda Zhang, Xingkui Wei, Xiangyang Xue, and Yanwei Fu. Learning compositional representation for 4d captures with neural ode. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5340–5350, 2021.

[8] Linpu Jiang, Ke-Jia Chen, and Jingqiang Chen. Self-supervised dynamic graph representation learning via temporal subgraph contrast. *arXiv preprint arXiv:2112.08733*, 2021.

[9] Ming Jin, Yizhen Zheng, Yuan-Fang Li, Chen Gong, Chuan Zhou, and Shirui Pan. Multi-scale contrastive siamese networks for self-supervised graph representation learning. In *International Joint Conference on Artificial Intelligence*, pages 1477–1483, 2021.

[10] Ming Jin, Yu Zheng, Yuan-Fang Li, Siheng Chen, Bin Yang, and Shirui Pan. Multivariate time series forecasting with dynamic graph neural odes. *arXiv preprint arXiv:2202.08408*, 2022.

[11] Seyed Mehran Kazemi. Dynamic graph neural networks. In *Graph Neural Networks: Foundations, Frontiers, and Applications*, pages 323–349. Springer, 2022.

[12] Lauri Kovanen, Márton Karsai, Kimmo Kaski, János Kertész, and Jari Saramäki. Temporal motifs in time-dependent networks. *Journal of Statistical Mechanics: Theory and Experiment*, 2011(11):P11005, 2011.

[13] Srijan Kumar, Xikun Zhang, and Jure Leskovec. Predicting dynamic embedding trajectory in temporal interaction networks. In *Proceedings of the 25th ACM SIGKDD international conference on knowledge discovery & data mining*, pages 1269–1278, 2019.

[14] Günter Last and Mathew Penrose. *Lectures on the Poisson Process*. Institute of Mathematical Statistics Textbooks. Cambridge University Press, 2017.

[15] Jure Leskovec and Andrej Krevl. Snap datasets: Stanford large network dataset collection; 2014, 2016.

[16] Xu Liu, Yuxuan Liang, Yu Zheng, Bryan Hooi, and Roger Zimmermann. Spatio-temporal graph contrastive learning. *arXiv preprint arXiv:2108.11873*, 2021.

[17] Yixin Liu, Ming Jin, Shirui Pan, Chuan Zhou, Yu Zheng, Feng Xia, and Philip Yu. Graph self-supervised learning: A survey. *IEEE Transactions on Knowledge and Data Engineering*, 2022.

[18] Shmoolik Mangan and Uri Alon. Structure and function of the feed-forward loop network motif. *Proceedings of the National Academy of Sciences*, 100(21):11980–11985, 2003.

[19] Silvio Micali and Zeyuan Allen Zhu. Reconstructing markov processes from independent and anonymous experiments. *Discrete Applied Mathematics*, 200:108–122, 2016.

[20] 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.

[21] Aldo Pareja, Giacomo Domeniconi, Jie Chen, Tengfei Ma, Toyotaro Suzumura, Hiroki Kanezashi, Tim Kaler, Tao Schardl, and Charles Leiserson. Evolvegcn: Evolving graph convolutional networks for dynamic graphs. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 5363–5370, 2020.

[22] Michael Poli, Stefano Massaroli, Junyoung Park, Atsushi Yamashita, Hajime Asama, and Jinkyoo Park. Graph neural ordinary differential equations. *arXiv preprint arXiv:1911.07532*, 2019.

[23] Rui Qian, Tianjian Meng, Boqing Gong, Ming-Hsuan Yang, Huisheng Wang, Serge Belongie, and Yin Cui. Spatiotemporal contrastive video representation learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6964–6974, 2021.

[24] 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.

[25] Yulia Rubanova, Ricky TQ Chen, and David K Duvenaud. Latent ordinary differential equations for irregularly-sampled time series. *Advances in neural information processing systems*, 32, 2019.

[26] Aravind Sankar, Yanhong Wu, Liang Gou, Wei Zhang, and Hao Yang. Dysat: Deep neural representation learning on dynamic graphs via self-attention networks. In *Proceedings of the 13th International Conference on Web Search and Data Mining*, pages 519–527, 2020.

[27] Youngjoo Seo, Michaël Defferrard, Pierre Vandergheynst, and Xavier Bresson. Structured sequence modeling with graph convolutional recurrent networks. In *International Conference on Neural Information Processing*, pages 362–373. Springer, 2018.

[28] Georg Simmel. *The sociology of georg simmel*, volume 92892. Simon and Schuster, 1950.

[29] Joakim Skarding, Bogdan Gabrys, and Katarzyna Musial. Foundations and modeling of dynamic networks using dynamic graph neural networks: A survey. *IEEE Access*, 9:79143–79168, 2021.

[30] Sheng Tian, Ruofan Wu, Leilei Shi, Liang Zhu, and Tao Xiong. Self-supervised representation learning on dynamic graphs. In *Proceedings of the 30th ACM International Conference on Information & Knowledge Management*, pages 1814–1823, 2021.

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

[32] Petar Velickovic, William Fedus, William L Hamilton, Pietro Liò, Yoshua Bengio, and R Devon Hjelm. Deep graph infomax. *International Conference on Learning Representations*, 2(3):4, 2019.

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

[34] Louis-Pascal Xhonneux, Meng Qu, and Jian Tang. Continuous graph neural networks. In *International Conference on Machine Learning*, pages 10432–10441. PMLR, 2020.

[35] Da Xu, Chuanwei Ruan, Evren Korpeoglu, Sushant Kumar, and Kannan Achan. Inductive representation learning on temporal graphs. In *International Conference on Learning Representations*, 2020.

[36] Yiwen Zhan, Yuchen Chen, Pengfei Ren, Haifeng Sun, Jingyu Wang, Qi Qi, and Jianxin Liao. Spatial temporal enhanced contrastive and pretext learning for skeleton-based action representation. In *Asian Conference on Machine Learning*, pages 534–547. PMLR, 2021.

[37] Yujia Zhang, Lai-Man Po, Xuyuan Xu, Mengyang Liu, Yexin Wang, Weifeng Ou, Yuzhi Zhao, and Wing-Yin Yu. Contrastive spatio-temporal pretext learning for self-supervised video representation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, 2022.

[38] Yizhen Zheng, Ming Jin, Shirui Pan, Yuan-Fang Li, Hao Peng, Ming Li, and Zhao Li. Towards graph self-supervised learning with contrastive adjusted zooming. *arXiv preprint arXiv:2111.10698*, 2021.

[39] Hongkuan Zhou, Da Zheng, Israt Nisa, Vasileios Ioannidis, Xiang Song, and George Karypis. Tgl: A general framework for temporal gnn training on billion-scale graphs. *arXiv preprint arXiv:2203.14883*, 2022.

[40] Han Zhu, Xiang Li, Pengye Zhang, Guozheng Li, Jie He, Han Li, and Kun Gai. Learning tree-based deep model for recommender systems. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pages 1079–1088, 2018.

Checklist

1. For all authors...
   (a) Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope? [Yes]
   (b) Did you describe the limitations of your work? [Yes] See Appendix D.4
   (c) Did you discuss any potential negative societal impacts of your work? [N/A]
   (d) Have you read the ethics review guidelines and ensured that your paper conforms to them? [Yes]
2. If you are including theoretical results...
   (a) Did you state the full set of assumptions of all theoretical results? [Yes]
   (b) Did you include complete proofs of all theoretical results? [Yes] See Appendix B.3 for the detailed justification of our batching mechanism discussed in Sec. 4.5.
3. If you ran experiments...
   (a) Did you include the code, data, and instructions needed to reproduce the main experimental results (either in the supplemental material or as a URL)? [Yes] We upload the code as a part of the supplemental material.
   (b) Did you specify all the training details (e.g., data splits, hyperparameters, how they were chosen)? [Yes] See Appendix C.3
   (c) Did you report error bars (e.g., with respect to the random seed after running experiments multiple times)? [Yes]
   (d) Did you include the total amount of compute and the type of resources used (e.g., type of GPUs, internal cluster, or cloud provider)? [Yes] See Appendix C.3
4. If you are using existing assets (e.g., code, data, models) or curating/releasing new assets...
   (a) If your work uses existing assets, did you cite the creators? [Yes]
   (b) Did you mention the license of the assets? [Yes]
   (c) Did you include any new assets either in the supplemental material or as a URL? [No]
   (d) Did you discuss whether and how consent was obtained from people whose data you're using/curating? [Yes] We discuss how to generate datasets in Appendix C.1
   (e) Did you discuss whether the data you are using/curating contains personally identifiable information or offensive content? [N/A]
5. If you used crowdsourcing or conducted research with human subjects...
   (a) Did you include the full text of instructions given to participants and screenshots, if applicable? [N/A]
   (b) Did you describe any potential participant risks, with links to Institutional Review Board (IRB) approvals, if applicable? [N/A]
   (c) Did you include the estimated hourly wage paid to participants and the total amount spent on participant compensation? [N/A]