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

Ming Jin
Monash University
ming.jin@monash.edu

Yuan-Fang Li
Monash University
yuanfang.li@monash.edu

Shirui Pan *
Griffith University
s.pan@griffith.edu.au

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 have been well-considered and explicitly modeled in capturing dynamic patterns. In this paper, we introduce a new approach, *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, our method 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 allow for irregularly-sampled temporal nodes to be embedded explicitly over multiple different interaction time intervals, enabling the effective 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 six real-world datasets [1].

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 [13, 32]. Traditional studies in dynamic graph modeling manually extract expressive patterns that are beneficial for understanding the crucial laws behind [36]. For example, two people are likely to know each other if they have a common friend (Figure 1). Such a dynamic graph motif describes how social connections are established [31]. Other expressive patterns, such as feedforward control loops, have also been investigated [20]. However, manually extracting motifs is expensive, time-consuming, and requires domain knowledge, thus hindering the learning on dynamic graphs with more complicated laws [36].

The advent of graph neural networks (GNNs) makes it possible to understand more complicated graphs by automatically learning the laws behind [40, 42]. 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
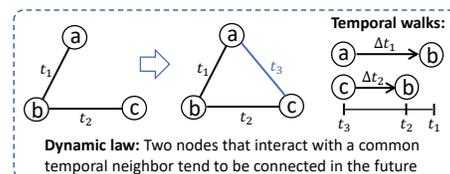

Figure 1: Temporal walks capture the law

---

off-the-shelf GNNs. To overcome this barrier, previous works simplify CTDGs to a series of static graph snapshots with *uniform time intervals* (i.e., discrete-time dynamic graphs, DTDGs) [24, 29]. However, this approximation compromises the modeling precision. Although [15] and [34] 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 [38, 27, 36] 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 different timestamps in Figure 1), resulting in a significant challenge in modeling temporal dependencies. Previous works typically bypass this challenge with the time encoding [38] to enable the use of message passing [38, 27] or sequence models [15, 36]. However, time encoding hurts model performance as temporal dependencies are modeled implicitly.

To tackle the above challenges, we propose the *Neural Temporal Walks* (`NeurTWs`) method for representation learning on CTDGs, which extracts and encodes informative dynamic graph motifs composed of irregularly-sampled temporal nodes. Specifically, motivated by [36], 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-aware node representations. The former process learns latent spatiotemporal dynamics across multiple interaction time intervals with an ordinary differential equation (ODE) function, and the second process regularizes the latent state trajectories with those irregularly-sampled observations.

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 five benchmark datasets and a new dense e-commerce dataset, our method significantly and consistently outperforms all state-of-the-art methods in general. Specifically, it surpasses the strongest baselines by around 3% and 5% in all transductive and seven out of eight inductive link prediction tasks. It also achieves the best or on-par performance on dynamic node classification 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 to encode 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). Existing 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 [30, 24, 5, 29]. 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 [38] samples a set of k-hop temporal neighbors and proposes learnable time encodings to preserve time information in message passing. TGN [27] further equips TGAT with a node memory update mechanism as in [15]. Another line of research leverages random walks to learn on CTDGs. Specifically, CTDNE [23] is the first to propose a CTDG embedding method with temporal walks. CAWs [36] extend this concept with anonymous temporal walks and uses a recurrent net to learn walk embeddings that are further aggregated when calculating interactive representations. Our method is different from prior walk-based approaches in three aspects: (1) We propose a new perspective on sampling temporal walks. While prior arts only consider time constraints, our method leverages multidimensional information,

allowing the 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 without relying on time encodings, which allows temporal dependencies to be modeled explicitly; (3) We replace the simple link prediction-based learning objective with a more challenging contrastive pretext task, which helps provide stronger supervision signals.

Neural Ordinary Differential Equations (NODEs). Chen *et al.* [2] propose 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 [12, 28] and computer vision [9, 1]. Recently, some works have extended NODEs to the graph domain, where most consider building deeper GNNs while alleviating the negative impacts of over-smoothing [37, 25]. Notably, the time information is absent among those works. In dynamic graph learning, most ODE-based works focus on discrete-time settings [12, 6, 8], and only a few extend NODEs to learn on CTDGs [7]. In this paper, inspired by the research on time series forecasting [28], we propose NeurTWs to encode extracted motifs with irregularly-sampled temporal nodes on CTDGs, which is fundamentally different from [7]: We directly integrate over multiple interaction time intervals to explicitly model the latent spatiotemporal dynamics across different temporal nodes with an ODE function, while [7] relies on a time encoding-assisted message passer to learn from historical temporal events.

Graph Contrastive Learning (GCL). Recently, GCL has achieved great success in graph self-supervised learning [19]. Most existing works are on static graphs [35, 43, 44, 11]. While some studies have explored the possibility of dynamic graph contrastive learning, many of them are on DTDGs. For example, STGCL [18] enhances the model's forecasting ability with DTDG augmentations and an auxiliary contrastive loss. A similar design also exists in [41, 39] and [26]. On CTDGs, Tian *et al.* [33] propose to maximize the agreement between time-aware node embeddings at different time points. In DySubC [10], 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.

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., $(e_i, t_i) := (\{u_i, v_i\}, t_i), \; t_i \in \mathbb{R}^+$. 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 [45]. 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-aware 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 $y_{u,v,t} = 0$. On this basis and with learned time-aware node representations, conducting other downstream tasks, such as dynamic node classification with another classifier, is also feasible.

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 [14], 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
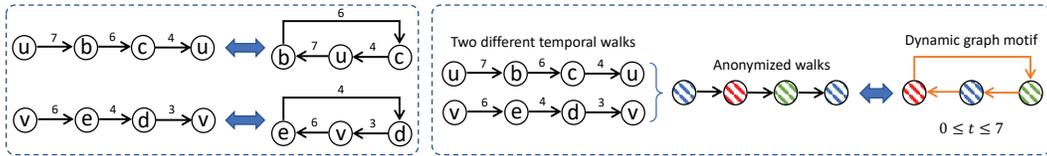
Figure 2: Triadic closures and the dynamic graph motifs: Two example temporal walks form two different triadic closures but represent the same motif within the time range $0 \leq t \leq 7$.
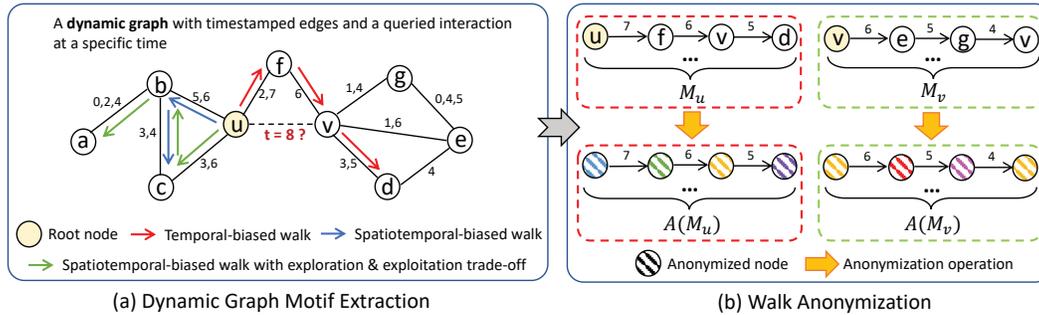

(a) Dynamic Graph Motif Extraction

(b) Walk Anonymization

Figure 3: Temporal walk sampling and anonymization. Given a CTDG and a queried interaction $(u, v, 8)$, we first extract surrounding temporal motifs by sampling a set of diverse and 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.

temporal nodes as in [36], i.e., node w_i at time t_i with $w_0 := u$ and $t_0 := t$:

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

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.

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 Figure 2, where two walks in the leftmost side form two triadic closures, which essentially represent the same motif on the rightmost side after an individual walk anonymization [21]. The necessity of anonymization is to replace 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 [36]:

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

4.2 Dynamic Graph Motif Extraction

Temporal Walk Sampling. Existing path-based methods mainly employ a temporal-biased sampling method when extracting dynamic graph motifs [23, 36]. 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 Figure 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 $u \rightarrow f$ instead of $u \rightarrow c$; (2) Neighbors with higher connectivity need to be emphasized to allow 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. Algorithm 1 illustrates the walk sampling procedures with the above considerations. Given a node u at time t, the probability of its temporal neighbor a to be sampled is the average of the following normalized probabilities:

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

Algorithm 1 Sampling Temporal Walks

Require: Root node w_0, cut time t_0, $\mathcal{G}$, C, l
1: Initialize $\{W_c = ((w_0, t_0)) \mid 1 \le c \le C\}$
2: **for** i in $1, 2, \cdots, l$ **do**
3:    **for** j in $1, 2, \cdots, C$ **do**
4:       $w_p, t_p := W_j[i][0], W_j[i][1]$
5:       Initialize $d_w = 0$ for all $w \in \mathcal{G}_{w_p, t_p}$
6:       **for** (e, t) in $\mathcal{G}_{w_p, t_p}$ **do**
7:          Let $e := \{w, w_p\}$, $d_w = |\mathcal{G}_{w,t}|$
8:       **end for**
9:       Sample one $(e, t) \in \mathcal{G}_{w_p, t_p}$ with prob. $\propto exp(\alpha(t - t_p))$ and $exp(-\beta/d_w)$
10:      Let $e := \{w, w_p\}$, $W_c = W_c || (w, t)$
11:    **end for**
12: **end for**
13: **return** $\{W_c \mid 1 \le c \le C\}$

Although Algorithm 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 Figure 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:

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

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

Our complete walk sampling algorithm and its complexity analysis are in Appendices 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 Equations 3, 4 and 5. In NeurTWs, given a queried interaction between two temporal nodes u and v as shown in Figure 3, we sample a set of C temporal walks rooted at each node with length l, denoted as M_u and M_v.

Anonymization. Walk anonymization replaces node identities with position encodings (aka relative identities), which injects structural information while maintaining the inductiveness of NeurTWs. A drawback of Equation 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 [36]. Thus, we consider two practical solutions: *unitary* and *binary anonymization* to address 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:

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

In Equation 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 [36]:

$$A(w; M_u, M_v) = A(w; M_u) \,||\, A(w; M_v), \quad (7)$$

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 $\widehat{W} = \{(A(w_i), t_i) \mid (w_i, t_i) \in W \text{ for } i = 0, 1, \cdots, l\}$ as an anonymous walk.

4.3 Neural Temporal Walk Encoding

A significant challenge to model CTDGs is that interactions occur irregularly. Previous works bypass this challenge by concatenating node attributes with extra time encodings when aggregating the neighbourhood information [38, 27, 36], where temporal dependencies are modeled implicitly. A detailed discussion is in Appendix D.2. To encode a motif with irregularly-sampled temporal nodes, we explicitly integrate over multiple interaction time intervals to learn the latent spatiotemporal dynamics with those discrete observations. Figure 4 and Algorithm 2 describe our method in a nutshell, which consists of two interleaving steps: *Continuous evolution* and *instantaneous activation*.

Algorithm 2 Neural Temporal Walk Encoding

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

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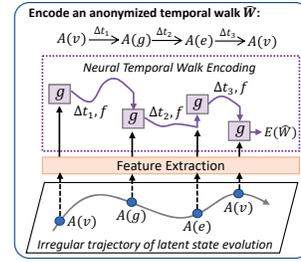8: **return** The walk embedding h_l

Figure 4: The spatiotemporal dynamics behind irregularly-sampled temporal nodes are explicitly modeled.

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:

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

where h_{i-1} denotes 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 Equation 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 walk $\widehat{W}$.

Compared with [36], we naturally model spatiotemporal dynamics behind walks with irregularly-sampled temporal nodes, where the time information has been explicitly reflected in this modeling process. On this basis, for a temporal node u, we obtain its time-aware representation by pooling the embedding of walks in M_u, denoted as $\overline{h}_u$. In this paper, we adopt the sum pooling for simplicity.

4.4 Contrastive Optimization

A self-supervised pretext task is required to train DGNNs due to the scarcity of the 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 the Jensen-Shannon estimator [19], 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 Figure 3) is maximized. We detail the complete training algorithm of our method in Appendix B.4. In brief, the following noise-contrastive loss is minimized in the proposed approach:

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

$\text{sim}(\cdot)$ measures the similarity between two entities, i.e., $\text{sim}(\overline{h}_u, \overline{h}_v) = \sigma(\text{MLP}(\overline{h}_u, \overline{h}_a, \xi))$, 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 Equation 9 as follows:

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

Table 1: The dataset statistics. Average interaction intensity $\lambda = 2N/(|V|T)$ [16, 36] embodies the density of interactions in a fixed period, where T and $|V|$ are dataset duration and number of nodes.

Statistics	CollegeMsg	Enron	Taobao	MOOC	Wikipedia	Reddit
# Nodes & Interactions	1,899 & 59,835	143 & 62,617	64,703 & 77,436	7,144 & 411,749	9,227 & 157,474	10,984 & 672,447
Duration (second)	16,621,303	72,932,520	36,000	2,572,086	2,678,373	2,678,390
# Nodes & Interaction attributes	0 & 0	0 & 0	0 & 4	0 & 4	172 & 172	172 & 172
Average interaction intensity λ	3.79×10^{-6}	1.2×10^{-5}	6.64×10^{-5}	4.48×10^{-5}	1.27×10^{-5}	4.57×10^{-5}
# Dynamically labeled nodes	-	-	-	-	217	366

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 timestamps, 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 $\mathcal{O}(Bl)$. In Appendix B.3, we also provide detailed complexity analysis and discuss how to make solving Equation 8 tractable with very large time intervals.

5 Experiments

5.1 Experimental Setting

Baselines. Our model is compared with six 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 [34], TGAT [38], and TGN [27]; (2) sequential model-based methods, including CTDNE [23], JODIE [15], and CAWs [36]. More details can be found in Appendix C.2.

Datasets. We evaluate model performance on six real-world datasets. CollegeMsg [17] is a social network dataset consisting of message sending activities. Enron [17] is an email communication network. Taobao [46] is an attributed user behavior dataset containing user-item interactions. MOOC [17] is an attributed network consisting of student-course interactions. Wikipedia and Reddit [15] are two bipartite interaction networks consisting of editor-page and user-post activities with rich attributive information. The dataset statistics are summarized in Table 1, and their detailed descriptions are in Appendix C.1.

Evaluation Protocols. For link prediction, we follow the evaluation protocols of CAWs [36] 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.

In *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. In particular, 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.

For dynamic node classification, we follow the evaluation protocol in [27]. Specifically, we first obtain a model under the setting of *transductive* link prediction. Then, we train and test a separate classifier on top of this pre-trained model with the temporal nodes observed in $[0, N_{val})$ and $[N_{val}, N]$.

Training Details. We implement and train all models under a unified evaluation framework with the Adam optimizer. The tuning of primary hyperparameters is discussed in Appendix C.3. In solving ODEs, we use the Runge-Kutta method with the number of function evaluations set to 8 by default. 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 methods are tuned thoroughly with nonlinear 2-layer and 3-layer perceptrons to conduct downstream link

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		Method	CollegeMsg	Enron	Taobao	MOOC
Transductive		JODIE [15]	0.5846 ± 0.038	0.8714 ± 0.011	0.8477 ± 0.015	0.6815 ± 0.014
		DyRep [34]	0.5297 ± 0.042	0.8632 ± 0.013	0.8462 ± 0.012	0.6195 ± 0.018
		TGAT [38]	0.7528 ± 0.004	0.6592 ± 0.012	0.5400 ± 0.005	0.6750 ± 0.035
		TGN [27]	0.8990 ± 0.003	0.8944 ± 0.015	0.8484 ± 0.029	$\underline{0.7703 \pm 0.032}$
		CAWs [36]	0.9002 ± 0.002	0.9520 ± 0.002	0.8719 ± 0.001	0.6948 ± 0.053
		`NeurTWs`	$\underline{0.9526 \pm 0.002}$	$\underline{0.9564 \pm 0.005}$	$\mathbf{0.9100 \pm 0.014}$	$\mathbf{0.7756 \pm 0.031}$
		`NeurTWs`†	$\mathbf{0.9750 \pm 0.004}$	$\mathbf{0.9704 \pm 0.012}$	$\underline{0.8911 \pm 0.014}$	0.7470 ± 0.028
Inductive	New-Old	JODIE [15]	0.4589 ± 0.028	0.8182 ± 0.022	0.7626 ± 0.002	0.6304 ± 0.006
		DyRep [34]	0.4486 ± 0.021	0.7241 ± 0.025	0.7641 ± 0.012	0.5504 ± 0.010
		TGAT [38]	0.7240 ± 0.008	0.6131 ± 0.049	0.5537 ± 0.018	0.6410 ± 0.024
		TGN [27]	0.8699 ± 0.007	0.7068 ± 0.116	0.8706 ± 0.008	0.6968 ± 0.008
		CAWs [36]	0.8911 ± 0.015	$\mathbf{0.9612 \pm 0.002}$	0.8744 ± 0.004	0.7479 ± 0.023
		`NeurTWs`	$\underline{0.9575 \pm 0.011}$	0.9525 ± 0.002	$\mathbf{0.9316 \pm 0.018}$	$\mathbf{0.7822 \pm 0.004}$
		`NeurTWs`†	$\mathbf{0.9699 \pm 0.010}$	$\underline{0.9566 \pm 0.007}$	$\underline{0.9037 \pm 0.013}$	$\underline{0.7772 \pm 0.006}$
	New-New	JODIE [15]	0.5135 ± 0.048	0.7537 ± 0.025	0.7791 ± 0.004	0.8243 ± 0.007
		DyRep [34]	0.5813 ± 0.066	0.7184 ± 0.061	0.7716 ± 0.017	0.5288 ± 0.021
		TGAT [38]	0.7283 ± 0.029	0.6340 ± 0.032	0.5479 ± 0.025	0.6365 ± 0.014
		TGN [27]	0.7745 ± 0.102	0.9217 ± 0.026	0.8701 ± 0.011	0.6448 ± 0.053
		CAWs [36]	0.8974 ± 0.009	$\underline{0.9777 \pm 0.001}$	0.8762 ± 0.004	0.7558 ± 0.036
		`NeurTWs`	$\underline{0.9649 \pm 0.008}$	$\mathbf{0.9906 \pm 0.007}$	$\mathbf{0.9242 \pm 0.005}$	$\mathbf{0.8329 \pm 0.010}$
		`NeurTWs`†	$\mathbf{0.9768 \pm 0.008}$	0.9858 ± 0.015	$\underline{0.9140 \pm 0.013}$	$\underline{0.8302 \pm 0.007}$

prediction and node classification 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 link prediction performance of our method and state-of-the-art baselines are summarized in Table 2 (w.r.t. AUC) and Appendix D.1 (w.r.t. AP). As can be seen from this table, our method achieves the best performance on different tasks and datasets in general. Notably, `NeurTWs` surpasses the strongest model, CAWs, significantly on both unattributed and attributed CTDGs, e.g., 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.

Table 3: Dynamic node classification performance w.r.t. AUC. We use **bold font** and underline to highlight the best and second best performances. The baseline results are taken from [27].

Method	Wikipedia	Reddit
CTDNE [23]	0.7589 ± 0.005	0.5943 ± 0.006
JODIE [15]	0.8484 ± 0.012	0.6183 ± 0.027
DyRep [34]	0.8459 ± 0.022	0.6291 ± 0.024
TGAT [38]	0.8369 ± 0.007	0.6556 ± 0.007
TGN [27]	$\underline{0.8781 \pm 0.003}$	$\mathbf{0.6706 \pm 0.009}$
`NeurTWs`	$\mathbf{0.8851 \pm 0.003}$	$\underline{0.6652 \pm 0.022}$

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 models varies significantly. For instance, JODIE and DyRep have 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 walks and 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. It is worth noting that simple unitary anonymization (i.e., `NeurTWs`) and more complex binary anonymization (i.e., `NeurTWs`†) yield similar performance in most cases. Our method with the default, simpler unitary anonymization performs significantly better than CAWs, which utilizes binary anonymization, again demonstrating the superiority of our approach in modeling

Table 4: 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 (`NeurTWs`)	**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

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. Similar observations can also be found when compared on MOOC and CollegeMsg. This is because the CollegeMsg dataset has a lower average interaction intensity (i.e., $\lambda = 3.79 \times 10^{-6}$) compared with Taobao and MOOC datasets (i.e., $\lambda = 6.64 \times 10^{-5}$ and 4.48×10^{-5}). A lower λ indicates a larger average time span in a temporal walk, for which our proposed continuous evolution process is more effective.

In addition, we compare the performance of our method (i.e., `NeurTWs`) on dynamic node classification with state-of-the-art baselines as shown in Table 3, where our approach achieves the best or on-par performances, further confirming the effectiveness of the proposal. We do not report the results of `NeurTWs`† because node classification mainly considers properties of temporal nodes rather than interactions.

5.3 Ablation Study

In Table 4, we report the results of our ablation study on predicting *inductive* interactions in both new-old and new-new settings. Specifically, ablations 1, 2, and 3 remove Equations 3, 4, and 5 when sampling temporal walks. Ablation 4 disables Equation 8 when encoding an anonymous walk. Ablation 5 replaces our contrastive loss 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 the proposed 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` excel 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. A specific study related to continuous evolution is in Appendix D.2.

5.4 Parametric Sensitivity

We study the important settings in `NeurTWs` and have the following observations: (1) For each dataset, there are sweet spots in balancing three intensities (i.e., α, β, and γ). Specifically, different datasets with different average interaction intensities, namely λ, prefer different temporal-biased intensities. For example, a relatively large α benefits on CollegeMsg but hurts the performance on Taobao as shown in Figure 5(a). On both datasets, overly emphasizing the topology information with a large β hurts the performance (see Figure 5(b)) because the temporal information is overly neglected. However, setting a relatively large γ seems beneficial (see Figure 5(c)), indicating the importance of balancing exploitation and exploration in temporal walk sampling; (2) We also investigate the choice of walk length l and number of walks C as shown in Figures 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 Equation 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 Figure 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 provide a detailed study in Appendix D.2.
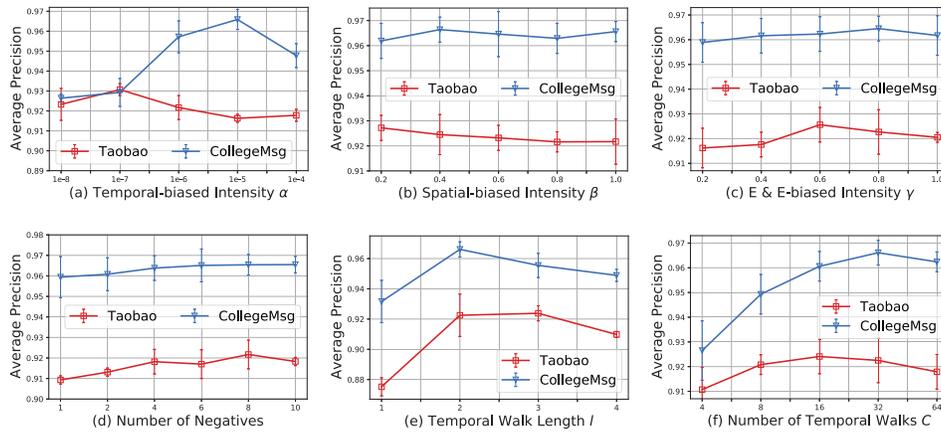
Figure 5: Study on important settings of `NeurTWs`. The performance in predicting *all inductive* interactions is reported.

6 Discussion and Conclusion

Limitations. There are a few limitations for our method. Firstly, a more sophisticated time interval normalization is required. In real-world dynamic graphs, there may be some large time intervals between temporal nodes when constructing temporal walks. Although we propose a simple solution based on logarithmic transformations (see Appendix B.3), there is no theoretical guarantee of stability when solving the continuous evolution process by this normalization trick. Secondly, the calculation of spatial-biased probabilities is computationally heavy. While it is practical to limit the number of spanned temporal neighbors to alleviate the computational burden (see Appendix D.4), we resort to finding a more efficient implementation in the future.

Social Impacts. Continuous-time dynamic graph representation learning benefits a wide range of real-world applications, including but not limited to recommender systems, social network mining, and industrial process modeling. However, there are also some potential negative impacts. For example, `NeurTWs` may learn skewed representations if there are biased patterns in the training data, which may result stereotyped predictions. Also, powerful dynamic graph models may augment harmful activities (e.g., attacking and phishing) on real-world dynamic systems.

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

https://doi.org/10.52202/068431-1445

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] Ricky TQ Chen, Brandon Amos, and Maximilian Nickel. Neural spatio-temporal point processes. In *International Conference on Learning Representations*, 2020.

[4] Kyunghyun Cho, Bart van Merriënboer, Dzmitry Bahdanau, and Yoshua Bengio. On the properties of neural machine translation: Encoder–decoder approaches. In *Proceedings of SSST-8, Eighth Workshop on Syntax, Semantics and Structure in Statistical Translation*, pages 103–111, 2014.

[5] 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.

[6] Zhen Han, Zifeng Ding, Yunpu Ma, Yujia Gu, and Volker Tresp. Temporal knowledge graph forecasting with neural ode. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, page 8352–8364, 2021.

[7] 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.

[8] 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.

[9] 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.

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

[11] Di Jin, Luzhi Wang, Yizhen Zheng, Xiang Li, Fei Jiang, Wei Lin, and Shirui Pan. Cgmn: A contrastive graph matching network for self-supervised graph similarity learning. In *Proceedings of the Thirty-First International Joint Conference on Artificial Intelligence*, pages 2101–2107, 2022.

[12] 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.

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

[14] 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.

[15] 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.

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

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

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

[19] 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.

[20] 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.

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

[22] Michael C Mozer, Denis Kazakov, and Robert V Lindsey. Discrete event, continuous time rnns. *arXiv preprint arXiv:1710.04110*, 2017.

[23] 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.

[24] 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.

[25] Michael Poli, Stefano Massaroli, Junyoung Park, Atsushi Yamashita, Hajime Asama, and Jinkyoo Park. Graph neural ordinary differential equations. In *International Workshop on Deep Learning on Graphs*, 2019.

[26] 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.

[27] 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.

[28] 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.

[29] 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.

[30] 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.

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

[32] 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.

[33] 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.

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

[35] Petar Velickovic, William Fedus, William L Hamilton, Pietro Liò, Yoshua Bengio, and R Devon Hjelm. Deep graph infomax. In *International Conference on Learning Representations*, 2019.

[36] 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.

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

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

[39] 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.

[40] He Zhang, Bang Wu, Xingliang Yuan, Shirui Pan, Hanghang Tong, and Jian Pei. Trustworthy graph neural networks: Aspects, methods and trends. *arXiv preprint arXiv:2205.07424*, 2022.

[41] 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.

[42] Xin Zheng, Miao Zhang, Chunyang Chen, Chaojie Li, Chuan Zhou, and Shirui Pan. Multi-relational graph neural architecture search with fine-grained message passing. In *IEEE International Conference on Data Mining*, 2022.

[43] 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. *IEEE Transactions on Neural Networks and Learning Systems*, 2022.

[44] Yizhen Zheng, Shirui Pan, Vincent Cs Lee, Yu Zheng, and Philip S Yu. Rethinking and scaling up graph contrastive learning: An extremely efficient approach with group discrimination. *Advances in Neural Information Processing Systems*, 2022.

[45] 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. *Proceedings of the VLDB Endowment*, 15(8):1572–1580, 2022.

[46] 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.