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

Ming Jin, Yuan-Fang Li, and Shirui Pan

Monash University, Griffith University

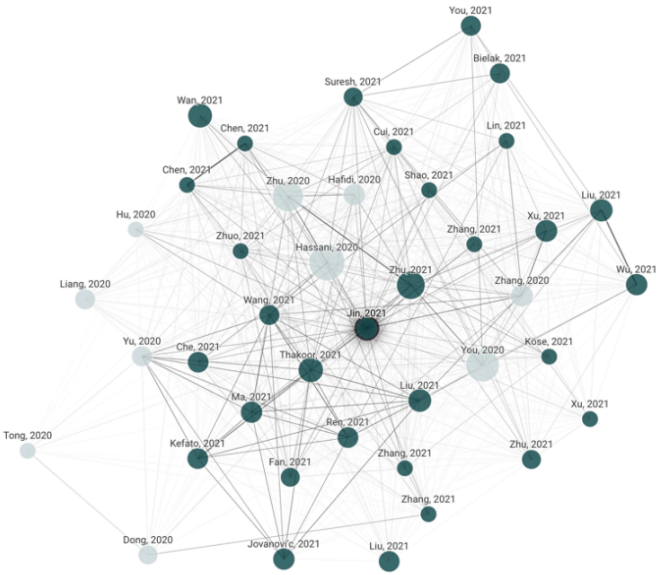
March 2, 2023

Code is available at:
https://github.com/KimMeen/Neural-Temporal-Walks

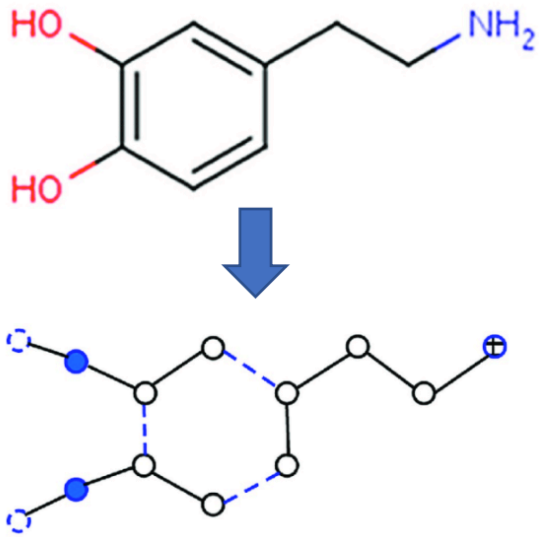
Contents

Background

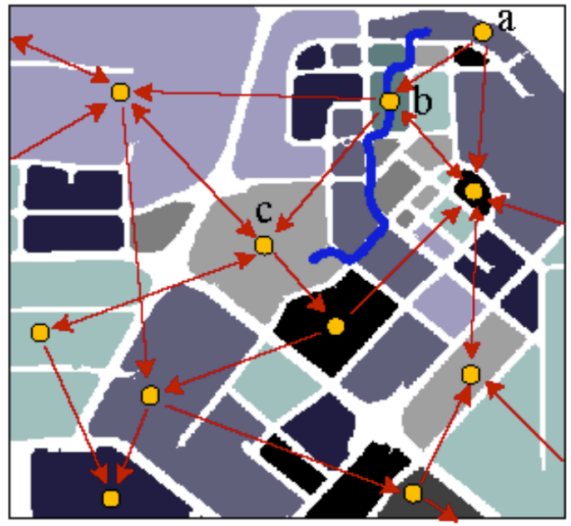
Graphs



(a) Citation Network



(b) Molecular Structure



(c) Traffic Network Topology

Figure: Three examples of real-world graphs

Dynamic Graphs

- Dynamic graph abstracts many real-world systems

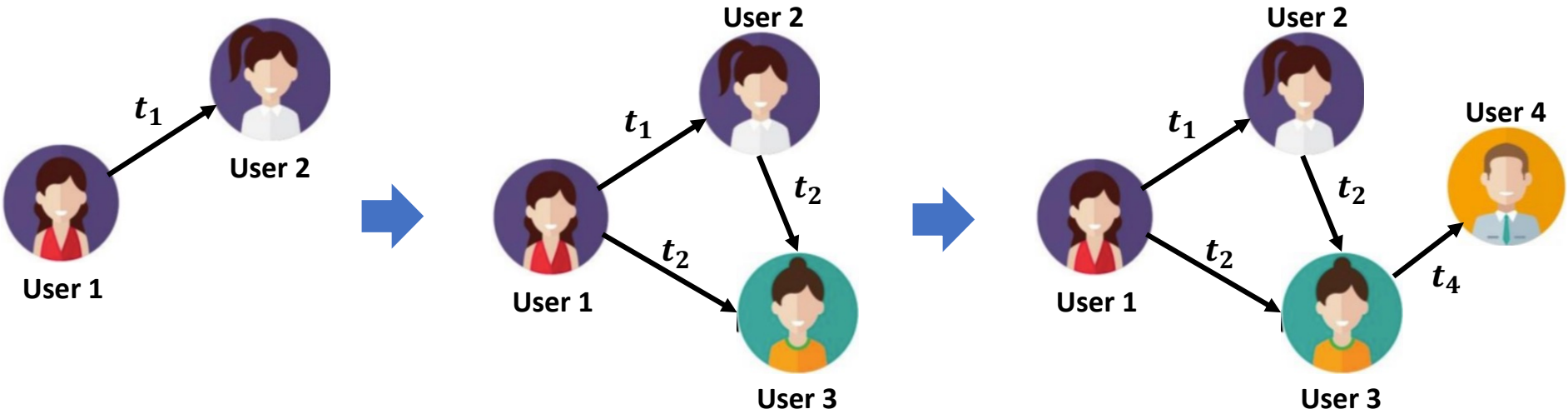
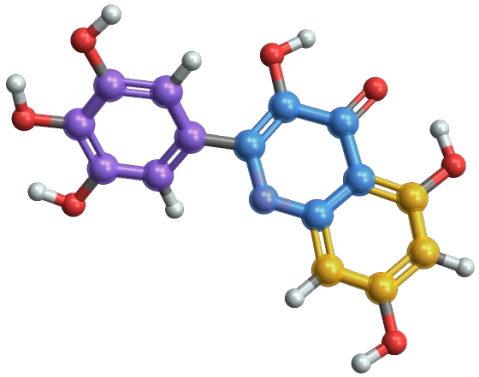


Figure: An example of social network evolving

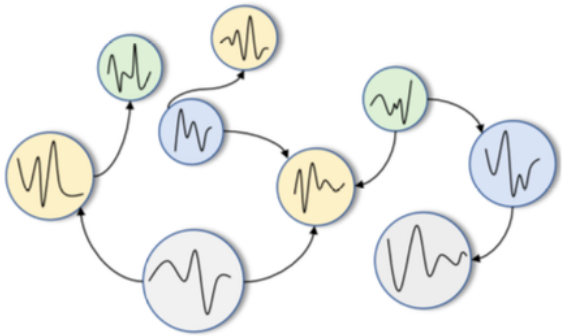
- Examples: Social platforms (e.g., user-user interactions) and online shopping websites (e.g., user-item interactions)

Dynamic Graphs: Taxonomy

- **Static graphs**: No temporal information involved, i.e., fixed structural and attributive information

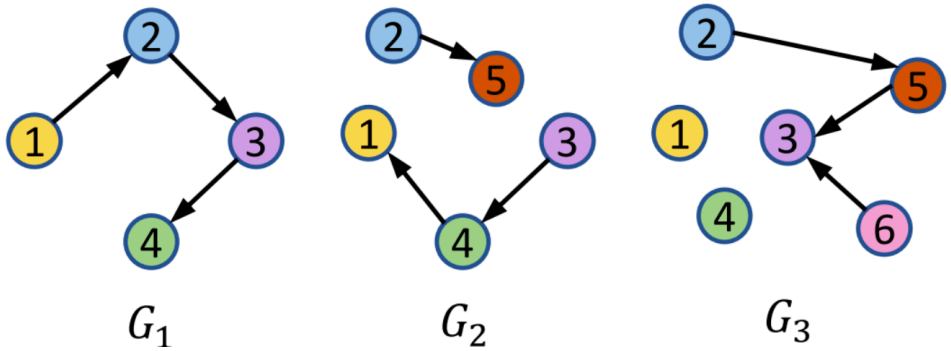


- **Edge weighted graphs**: Temporal information on edges and/or nodes (e.g., attributes) of a static graph, e.g., dynamic traffic volumes in metro networks with a fixed station topology

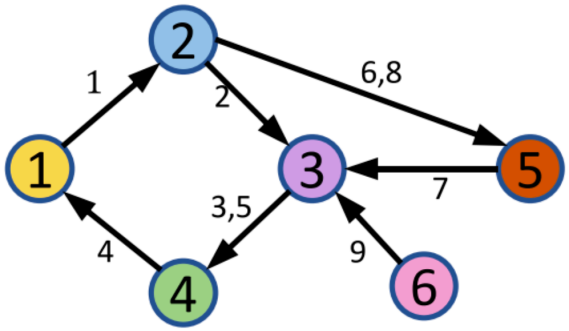


Dynamic Graphs: Taxonomy

- **Discrete-time dynamic graphs (DTDGs)**: A sequence of regularly-sampled (static) graph snapshots



- **Continuous-time dynamic graphs (CTDGs)**: A graph consisting of temporal events that are irregularly-sampled, such as the insertion or deletion of an edge at a specific time



Message-Passing Graph Neural Networks

- MP-GNN is a popular simplified paradigm in modeling graph-structured data

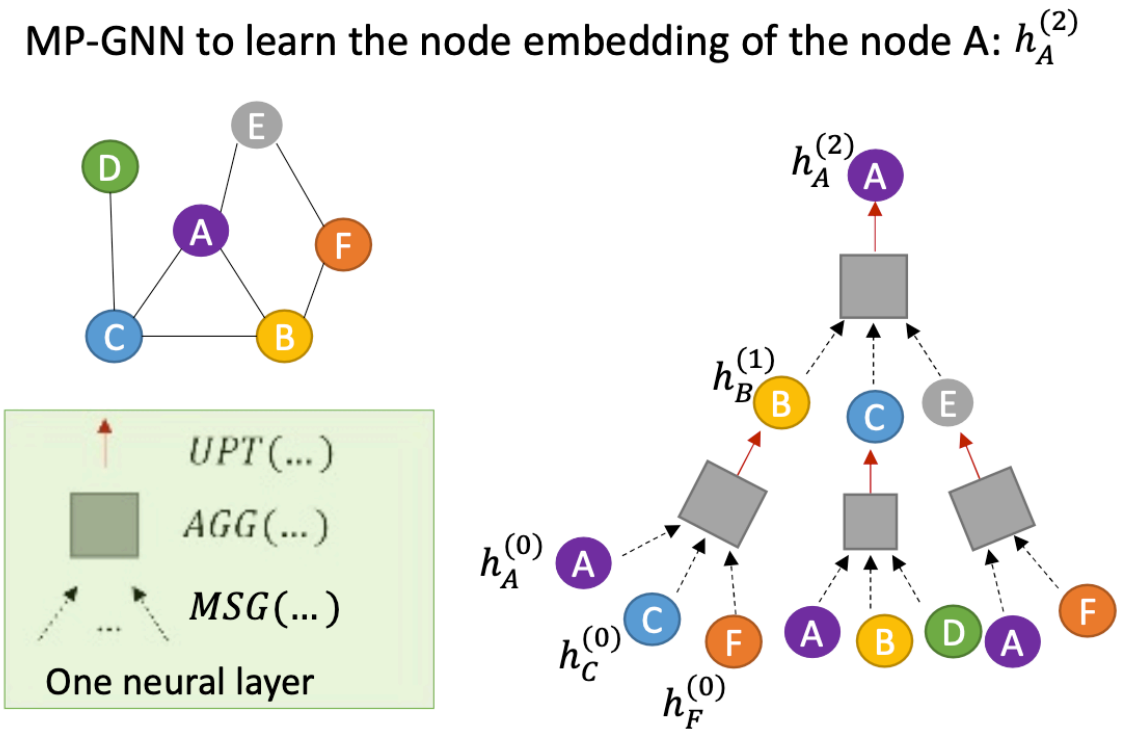


Figure: The computational flow of MP-GNN to calculate a node embedding

Message-Passing Graph Neural Networks

- Initialize node representations with native attributes: $\boldsymbol{h}_v^{(0)} \leftarrow \boldsymbol{X}_v, \forall v \in \mathcal{V}$
- Update each node representation over the graph structure:

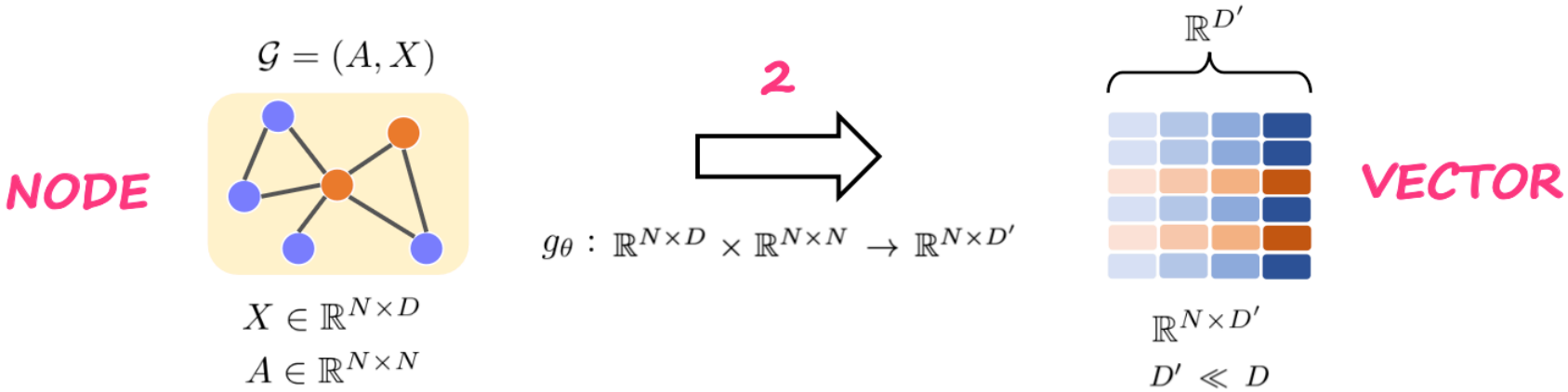
$$\textbf{Message: } \boldsymbol{m}_{vu}^{(l)} \leftarrow \mathsf{MSG}(\boldsymbol{h}_v^{(l-1)}, \boldsymbol{h}_u^{(l-1)}), \forall (u,v) \in \mathcal{E}$$

$$\textbf{Aggregation: } \boldsymbol{a}_v^{(l)} \leftarrow \mathsf{AGG}(\{\boldsymbol{m}_{vu}^{(l)} \mid u \in \mathcal{N}_v\}), \forall v \in \mathcal{V}$$

$$\textbf{Update: } \boldsymbol{h}_v^{(l)} \leftarrow \mathsf{UPT}(\boldsymbol{h}_v^{(l-1)}, \boldsymbol{a}_v^{(l)}), \forall v \in \mathcal{V}$$

Graph Representation Learning

- **"Node2Vec":** We aim to learn low-dimensional node embedding vectors



- The parameterized transformation g_θ can be a MP-GNN

Graph Representation Learning

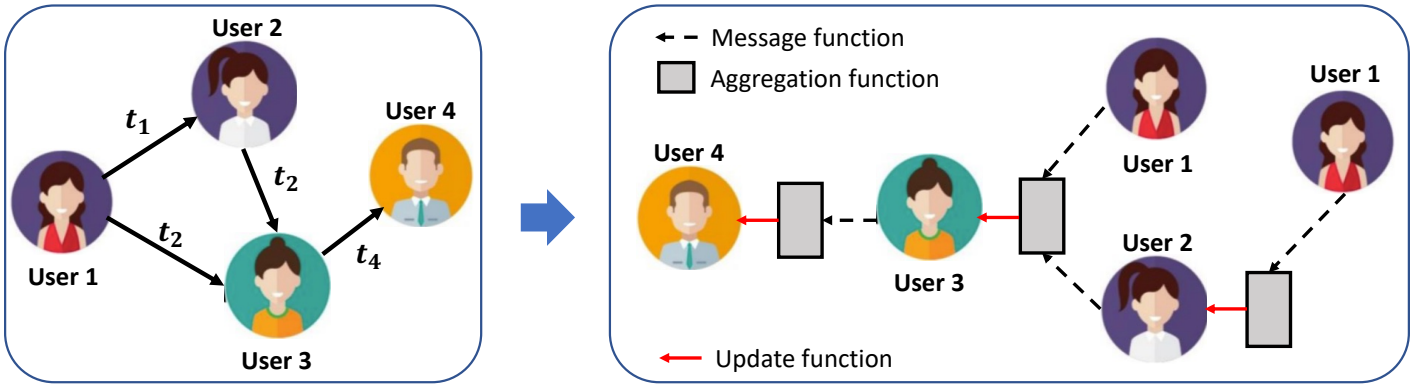
Question

How can conventional graph neural networks be extended to model continuous-time dynamic graphs?

Related Work

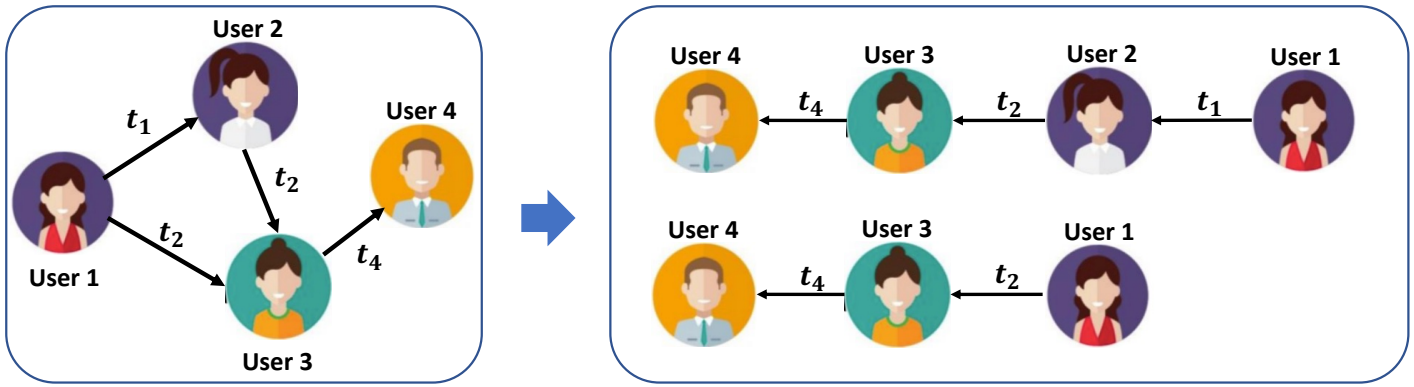
Dynamic Graph Neural Networks: Taxonomy

- **Node-based**: Extending the concept of message passing to aggregate the temporal neighborhood information



(a) Dynamic Graph (b) Temporal Message Passing

- **Subgraph-based**: Generalizing the path-based static network embedding methods to model dynamic graphs



(a) Dynamic Graph (b) Temporal Random Walks

Category 1. Temporal Message Passing

- Initialize time-aware node representations with raw attributes:
  $\boldsymbol{h}_{v,t}^{(0)} \leftarrow \boldsymbol{X}_{v,t}, \forall v \in \mathcal{V}, t \in \mathbb{R}^+$

- Update time-aware node representations w.r.t. the dynamic graph structural and attributive information:

$$\textbf{Message: } \boldsymbol{m}_{vu,t}^{(l)} \leftarrow \text{MSG}(\boldsymbol{h}_{v,t}^{(l-1)}, \boldsymbol{h}_{u,t_k}^{(l-1)}, \boldsymbol{z}^{(t-t_k)}), \forall (u,v,t) \in \mathcal{E}; t, t_k \in \mathbb{R}^+$$

$$\textbf{Aggregation: } \boldsymbol{a}_{v,t}^{(l)} \leftarrow \text{AGG}(\{\boldsymbol{m}_{vu,t}^{(l)} \mid u \in \mathcal{N}_{v,t}\}), \forall v \in \mathcal{V}, t \in \mathbb{R}^+$$

$$\textbf{Update: } \boldsymbol{h}_{v,t}^{(l)} \leftarrow \text{UPT}(\boldsymbol{h}_{v,t}^{(l-1)}, \boldsymbol{a}_{v,t}^{(l)}), \forall v \in \mathcal{V}, t \in \mathbb{R}^+$$

- **"Time2Vec"**: Taking important time gap information into account

$$\boldsymbol{z}_i^{(t)} = \begin{cases} \omega_i t + \phi_i, & \text{if } i = 0, \\ \sin \omega_i t + \phi_i, & \text{if } 1 \leq i \leq D'. \end{cases}$$

- Both *TGAT* and *TGN* build on this paradigm with some upgrades

Category 1. Temporal Message Passing

- Initialize time-aware node representations with raw attributes:
  $\boldsymbol{h}_{v,t}^{(0)} \leftarrow \boldsymbol{X}_{v,t}, \forall v \in \mathcal{V}, t \in \mathbb{R}^+$

- Update time-aware node representations w.r.t. the dynamic graph structural and attributive information:

$$\textbf{Message: } \boldsymbol{m}_{vu,t}^{(l)} \leftarrow \mathsf{MSG}(\boldsymbol{h}_{v,t}^{(l-1)}, \boldsymbol{h}_{u,t_k}^{(l-1)}, \boldsymbol{z}^{(t-t_k)}), \forall (u,v,t) \in \mathcal{E}; t, t_k \in \mathbb{R}^+$$

$$\textbf{Aggregation: } \boldsymbol{a}_{v,t}^{(l)} \leftarrow \mathsf{AGG}(\{\boldsymbol{m}_{vu,t}^{(l)} \mid u \in \mathcal{N}_{v,t}\}), \forall v \in \mathcal{V}, t \in \mathbb{R}^+$$

$$\textbf{Update: } \boldsymbol{h}_{v,t}^{(l)} \leftarrow \mathsf{UPT}(\boldsymbol{h}_{v,t}^{(l-1)}, \boldsymbol{a}_{v,t}^{(l)}), \forall v \in \mathcal{V}, t \in \mathbb{R}^+$$

- **"Time2Vec"**: Taking important time gap information into account

$$\boldsymbol{z}_i^{(t)} = \begin{cases} \omega_i t + \phi_i, & \text{if } i = 0, \\ \sin \omega_i t + \phi_i, & \text{if } 1 \leq i \leq D'. \end{cases}$$

- Both *TGAT* and *TGN* build on this paradigm with some upgrades

Category 1. Temporal Message Passing

- Initialize time-aware node representations with raw attributes:
  $\boldsymbol{h}_{v,t}^{(0)} \leftarrow \boldsymbol{X}_{v,t}, \forall v \in \mathcal{V}, t \in \mathbb{R}^+$

- Update time-aware node representations w.r.t. the dynamic graph structural and attributive information:

$$\textbf{Message: } \boldsymbol{m}_{vu,t}^{(l)} \leftarrow \text{MSG}(\boldsymbol{h}_{v,t}^{(l-1)}, \boldsymbol{h}_{u,t_k}^{(l-1)}, \boldsymbol{z}^{(t-t_k)}), \forall (u,v,t) \in \mathcal{E}; t, t_k \in \mathbb{R}^+$$

$$\textbf{Aggregation: } \boldsymbol{a}_{v,t}^{(l)} \leftarrow \text{AGG}(\{\boldsymbol{m}_{vu,t}^{(l)} \mid u \in \mathcal{N}_{v,t}\}), \forall v \in \mathcal{V}, t \in \mathbb{R}^+$$

$$\textbf{Update: } \boldsymbol{h}_{v,t}^{(l)} \leftarrow \text{UPT}(\boldsymbol{h}_{v,t}^{(l-1)}, \boldsymbol{a}_{v,t}^{(l)}), \forall v \in \mathcal{V}, t \in \mathbb{R}^+$$

- **"Time2Vec"**: Taking important time gap information into account

$$\boldsymbol{z}_i^{(t)} = \begin{cases} \omega_i t + \phi_i, & \text{if } i = 0, \\ \sin \omega_i t + \phi_i, & \text{if } 1 \leq i \leq D'. \end{cases}$$

- Both *TGAT* and *TGN* build on this paradigm with some upgrades

Category 1. Temporal Message Passing

- Initialize time-aware node representations with raw attributes:
  $\boldsymbol{h}_{v,t}^{(0)} \leftarrow \boldsymbol{X}_{v,t}, \forall v \in \mathcal{V}, t \in \mathbb{R}^+$

- Update time-aware node representations w.r.t. the dynamic graph structural and attributive information:

$$\textbf{Message: } \boldsymbol{m}_{vu,t}^{(l)} \leftarrow \text{MSG}(\boldsymbol{h}_{v,t}^{(l-1)}, \boldsymbol{h}_{u,t_k}^{(l-1)}, \boldsymbol{z}^{(t-t_k)}), \forall (u,v,t) \in \mathcal{E}; t, t_k \in \mathbb{R}^+$$

$$\textbf{Aggregation: } \boldsymbol{a}_{v,t}^{(l)} \leftarrow \text{AGG}(\{\boldsymbol{m}_{vu,t}^{(l)} \mid u \in \mathcal{N}_{v,t}\}), \forall v \in \mathcal{V}, t \in \mathbb{R}^+$$

$$\textbf{Update: } \boldsymbol{h}_{v,t}^{(l)} \leftarrow \text{UPT}(\boldsymbol{h}_{v,t}^{(l-1)}, \boldsymbol{a}_{v,t}^{(l)}), \forall v \in \mathcal{V}, t \in \mathbb{R}^+$$

- **"Time2Vec"**: Taking important time gap information into account

$$\boldsymbol{z}_i^{(t)} = \begin{cases} \omega_i t + \phi_i, & \text{if } i = 0, \\ \sin \omega_i t + \phi_i, & \text{if } 1 \leq i \leq D'. \end{cases}$$

- Both *TGAT* and *TGN* build on this paradigm with some upgrades

Category 2. Temporal Random Walk

- **Temporal walk**: A walk from v_1 to v_k in a continuous-time dynamic graph is a sequence of vertices $[v_1, v_2, \cdots, v_k]$, where $\left(v_i, v_{i+1}, \mathcal{T}(v_i, v_{i+1})\right) \in \mathcal{E}$ for $1 \leq i < k$, and $\mathcal{T}(v_i, v_{i+1}) \leq \mathcal{T}(v_{i+1}, v_{i+2})$ for $1 \leq i < (k-1)$
- **Initialization:** Given a CTDG, we sample an initial edge $e := (v, u)$ with the time $t_* = \mathcal{T}(e) := \mathcal{T}(v, u)$ from a distribution $\mathbb{F}_s$

$$Pr(e) = \frac{\exp[\mathcal{T}(e) - t_{min}]}{\sum_{e' \in \mathcal{E}} \exp[\mathcal{T}(e') - t_{min}]}$$

- **Walk construction:** We sample the rest of nodes in a walk from another distribution $\mathbb{F}_r$

$$Pr(w) = \frac{\exp[\tau(w) - \tau(u)]}{\sum_{w' \in \mathcal{N}_{u,\tau u}} \exp[\tau(w') - \tau(u)]}$$

- Given as set of walks S, learn the function $f : \mathcal{V} \to \mathbb{R}^{N \times D'}$ as follows:

$$\max_f \log Pr(W_T = \{v_{i-w}, \cdots, v_{i+w}\} \setminus v_i \mid f(v_i))$$

- Both *CTDNE* and *CAW* are based on the concept of temporal walks

Category 2. Temporal Random Walk

- **Temporal walk**: A walk from v_1 to v_k in a continuous-time dynamic graph is a sequence of vertices $[v_1, v_2, \cdots, v_k]$, where $\big(v_i, v_{i+1}, \mathcal{T}(v_i, v_{i+1})\big) \in \mathcal{E}$ for $1 \leq i < k$, and $\mathcal{T}(v_i, v_{i+1}) \leq \mathcal{T}(v_{i+1}, v_{i+2})$ for $1 \leq i < (k-1)$
- **Initialization:** Given a CTDG, we sample an initial edge $e := (v, u)$ with the time $t_* = \mathcal{T}(e) := \mathcal{T}(v, u)$ from a distribution $\mathbb{F}_s$

$$Pr(e) = \frac{\exp[\mathcal{T}(e) - t_{min}]}{\sum_{e' \in \mathcal{E}} \exp[\mathcal{T}(e') - t_{min}]}$$

- **Walk construction:** We sample the rest of nodes in a walk from another distribution $\mathbb{F}_r$

$$Pr(w) = \frac{\exp[\tau(w) - \tau(u)]}{\sum_{w' \in \mathcal{N}_{u,\tau u}} \exp[\tau(w') - \tau(u)]}$$

- Given as set of walks S, learn the function $f : \mathcal{V} \to \mathbb{R}^{N \times D'}$ as follows:

$$\max_f \log Pr(W_T = \{v_{i-w}, \cdots, v_{i+w}\} \setminus v_i \mid f(v_i))$$

- Both *CTDNE* and *CAW* are based on the concept of temporal walks

Category 2. Temporal Random Walk

- **Temporal walk**: A walk from v_1 to v_k in a continuous-time dynamic graph is a sequence of vertices $[v_1, v_2, \cdots, v_k]$, where $\big(v_i, v_{i+1}, \mathcal{T}(v_i, v_{i+1})\big) \in \mathcal{E}$ for $1 \leq i < k$, and $\mathcal{T}(v_i, v_{i+1}) \leq \mathcal{T}(v_{i+1}, v_{i+2})$ for $1 \leq i < (k-1)$
- **Initialization:** Given a CTDG, we sample an initial edge $e := (v, u)$ with the time $t_* = \mathcal{T}(e) := \mathcal{T}(v, u)$ from a distribution $\mathbb{F}_s$

$$Pr(e) = \frac{\exp[\mathcal{T}(e) - t_{min}]}{\sum_{e' \in \mathcal{E}} \exp[\mathcal{T}(e') - t_{min}]}$$

- **Walk construction:** We sample the rest of nodes in a walk from another distribution $\mathbb{F}_r$

$$Pr(w) = \frac{\exp[\tau(w) - \tau(u)]}{\sum_{w' \in \mathcal{N}_{u,\tau u}} \exp[\tau(w') - \tau(u)]}$$

- Given as set of walks S, learn the function $f : \mathcal{V} \to \mathbb{R}^{N \times D'}$ as follows:

$$\max_f \log Pr(W_T = \{v_{i-w}, \cdots, v_{i+w}\} \setminus v_i \mid f(v_i))$$

- Both *CTDNE* and *CAW* are based on the concept of temporal walks

Category 2. Temporal Random Walk

- **Temporal walk**: A walk from v_1 to v_k in a continuous-time dynamic graph is a sequence of vertices $[v_1, v_2, \cdots, v_k]$, where $\big(v_i, v_{i+1}, \mathcal{T}(v_i, v_{i+1})\big) \in \mathcal{E}$ for $1 \leq i < k$, and $\mathcal{T}(v_i, v_{i+1}) \leq \mathcal{T}(v_{i+1}, v_{i+2})$ for $1 \leq i < (k-1)$
- **Initialization:** Given a CTDG, we sample an initial edge $e := (v, u)$ with the time $t_* = \mathcal{T}(e) := \mathcal{T}(v, u)$ from a distribution $\mathbb{F}_s$

$$Pr(e) = \frac{\exp[\mathcal{T}(e) - t_{min}]}{\sum_{e' \in \mathcal{E}} \exp[\mathcal{T}(e') - t_{min}]}$$

- **Walk construction:** We sample the rest of nodes in a walk from another distribution $\mathbb{F}_r$

$$Pr(w) = \frac{\exp[\tau(w) - \tau(u)]}{\sum_{w' \in \mathcal{N}_{u,\tau u}} \exp[\tau(w') - \tau(u)]}$$

- Given as set of walks S, learn the function $f : \mathcal{V} \to \mathbb{R}^{N \times D'}$ as follows:

$$\max_f \log Pr(W_T = \{v_{i-w}, \cdots, v_{i+w}\} \setminus v_i \mid f(v_i))$$

- Both *CTDNE* and *CAW* are based on the concept of temporal walks

Category 2. Temporal Random Walk

- **Temporal walk**: A walk from v_1 to v_k in a continuous-time dynamic graph is a sequence of vertices $[v_1, v_2, \cdots, v_k]$, where $\big(v_i, v_{i+1}, \mathcal{T}(v_i, v_{i+1})\big) \in \mathcal{E}$ for $1 \leq i < k$, and $\mathcal{T}(v_i, v_{i+1}) \leq \mathcal{T}(v_{i+1}, v_{i+2})$ for $1 \leq i < (k-1)$
- **Initialization:** Given a CTDG, we sample an initial edge $e := (v, u)$ with the time $t_* = \mathcal{T}(e) := \mathcal{T}(v, u)$ from a distribution $\mathbb{F}_s$

$$Pr(e) = \frac{\exp[\mathcal{T}(e) - t_{min}]}{\sum_{e' \in \mathcal{E}} \exp[\mathcal{T}(e') - t_{min}]}$$

- **Walk construction:** We sample the rest of nodes in a walk from another distribution $\mathbb{F}_r$

$$Pr(w) = \frac{\exp[\tau(w) - \tau(u)]}{\sum_{w' \in \mathcal{N}_{u,\tau u}} \exp[\tau(w') - \tau(u)]}$$

- Given as set of walks S, learn the function $f : \mathcal{V} \to \mathbb{R}^{N \times D'}$ as follows:

$$\max_f \log Pr(W_T = \{v_{i-w}, \cdots, v_{i+w}\} \setminus v_i \mid f(v_i))$$

- Both *CTDNE* and *CAW* are based on the concept of temporal walks

Our Proposal

Motivations

- **Challenge 1.** The entangled spatial and temporal dependencies in real-world CTDGs require a specific paradigm to model
  - This prevents the direct use of off-the-shelf GNNs
  - Most of existing works simplify CTDGs to a series of static graph snapshots with *uniform time intervals*, i.e., DTDGs
  - Some works propose to directly learn on CTDGs, e.g., JODIE and DyRep, but the *inductiveness* of the patterns they captured is not guaranteed
  - Although some recent advantages attempt to alleviate this issue, e.g., TGAT and CAW, they usually fail to explore *diverse and expressive patterns* from real-world CTDGs

Motivations

- **Challenge 1.** The entangled spatial and temporal dependencies in real-world CTDGs require a specific paradigm to model
  - This prevents the direct use of off-the-shelf GNNs
  - Most of existing works simplify CTDGs to a series of static graph snapshots with *uniform time intervals*, i.e., DTDGs
  - Some works propose to directly learn on CTDGs, e.g., JODIE and DyRep, but the *inductiveness* of the patterns they captured is not guaranteed
  - Although some recent advantages attempt to alleviate this issue, e.g., TGAT and CAW, they usually fail to explore *diverse and expressive patterns* from real-world CTDGs

Motivations

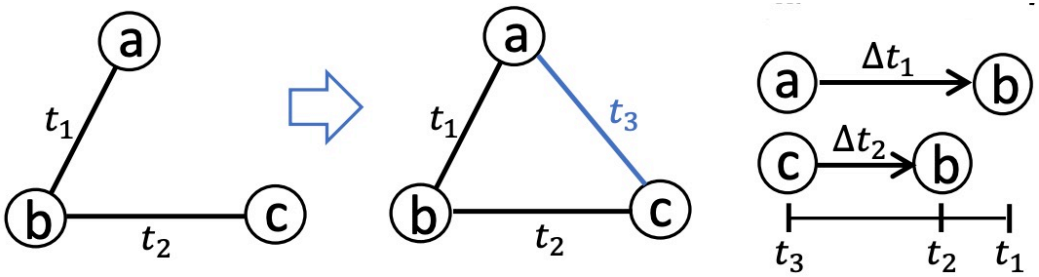
- **Challenge 1.** The entangled spatial and temporal dependencies in real-world CTDGs require a specific paradigm to model
  - This prevents the direct use of off-the-shelf GNNs
  - Most of existing works simplify CTDGs to a series of static graph snapshots with *uniform time intervals*, i.e., DTDGs
  - Some works propose to directly learn on CTDGs, e.g., JODIE and DyRep, but the *inductiveness* of the patterns they captured is not guaranteed
  - Although some recent advantages attempt to alleviate this issue, e.g., TGAT and CAW, they usually fail to explore *diverse and expressive patterns* from real-world CTDGs

Motivations

- **Challenge 1.** The entangled spatial and temporal dependencies in real-world CTDGs require a specific paradigm to model
  - This prevents the direct use of off-the-shelf GNNs
  - Most of existing works simplify CTDGs to a series of static graph snapshots with *uniform time intervals*, i.e., DTDGs
  - Some works propose to directly learn on CTDGs, e.g., JODIE and DyRep, but the *inductiveness* of the patterns they captured is not guaranteed
  - Although some recent advantages attempt to alleviate this issue, e.g., TGAT and CAW, they usually fail to explore *diverse and expressive patterns* from real-world CTDGs

Motivations

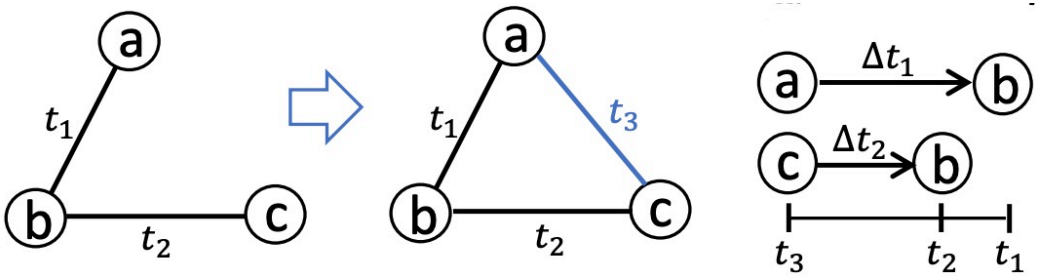
- **Challenge 2.** Temporal events in CTDGs occur *irregularly*, resulting in a significant challenge in modeling temporal dependencies
  - E.g., nodes a and c interact with b at different time (i.e., $\Delta t_1 \neq \Delta t_2$)



  - Previous works typically bypasses this challenge with the *time encoding* to enable the use of message passing or sequence models
    - The important time dependencies are modeled implicitly (as a part of attributive information)
    - We empirically find that this hurts the performance

Motivations

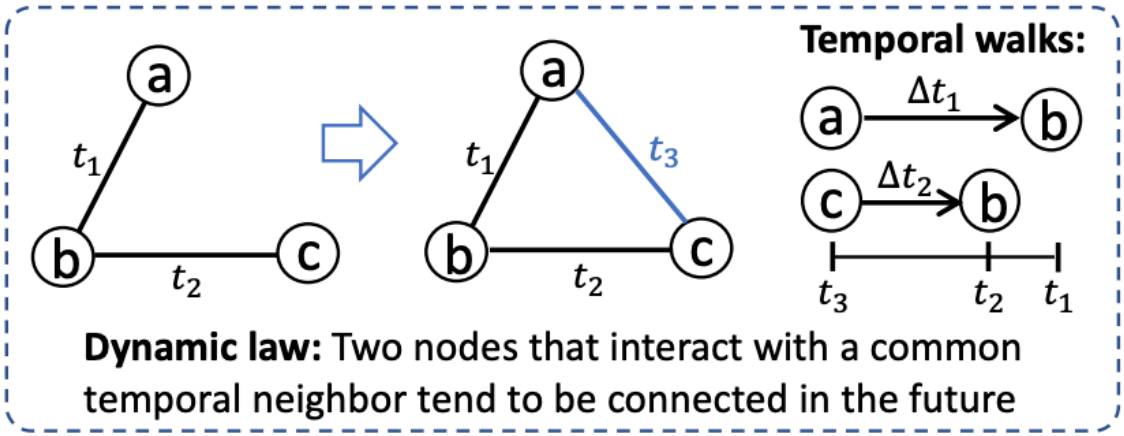
- **Challenge 2.** Temporal events in CTDGs occur *irregularly*, resulting in a significant challenge in modeling temporal dependencies
  - E.g., nodes a and c interact with b at different time (i.e., $\Delta t_1 \neq \Delta t_2$)



  - Previous works typically bypasses this challenge with the *time encoding* to enable the use of message passing or sequence models
    - The important time dependencies are modeled implicitly (as a part of attributive information)
    - We empirically find that this hurts the performance

Dynamic Graph Motif

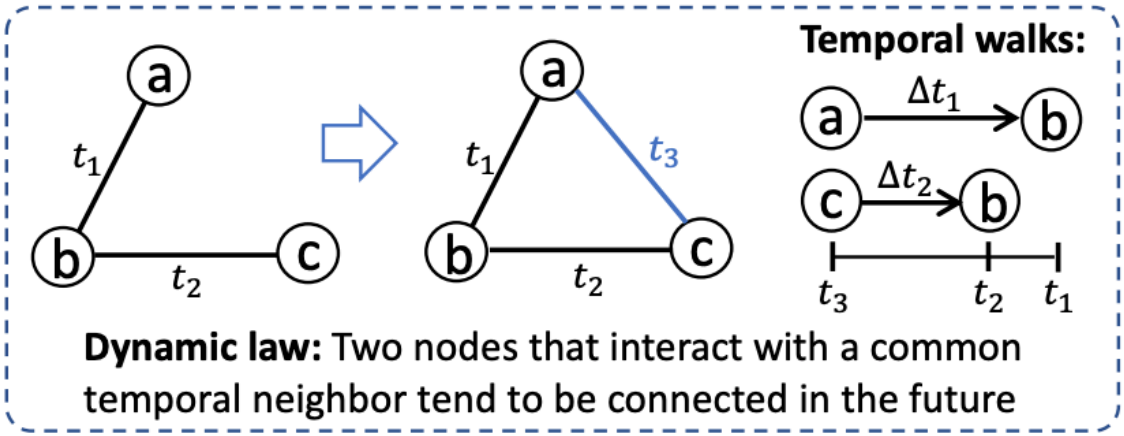
- Dynamic graph learning is about modeling *dynamic graph motifs*, which reflect essential *dynamic laws*
  - E.g., two people are likely to know each other if they have a common friend



  - In this case, ♠ → ★ → ♣ within the time range $0 \leq t \leq t_3$ is a dynamic graph motif that describe the law that ♠ is likely to interact with ♣ at a certain time
  - In this work, we mainly focus on answering two research questions: (1) How to extract diverse and expressive motifs and (2) how to encode these motifs to learn effective node representations

Dynamic Graph Motif

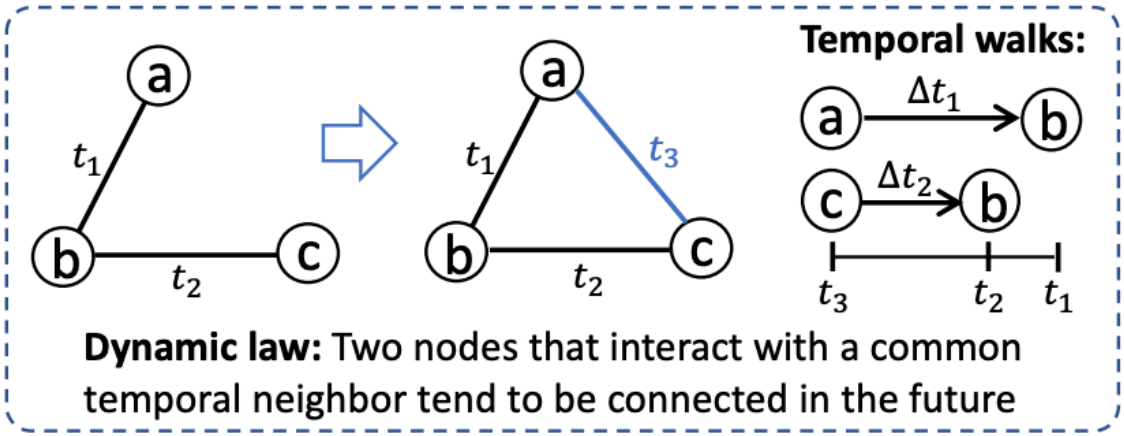
- Dynamic graph learning is about modeling *dynamic graph motifs*, which reflect essential *dynamic laws*
  - E.g., two people are likely to know each other if they have a common friend



  - In this case, ♠ → ★ → ♣ within the time range $0 \leq t \leq t_3$ is a dynamic graph motif that describe the law that ♠ is likely to interact with ♣ at a certain time
  - In this work, we mainly focus on answering two research questions: (1) How to extract diverse and expressive motifs and (2) how to encode these motifs to learn effective node representations

Dynamic Graph Motif

- Dynamic graph learning is about modeling *dynamic graph motifs*, which reflect essential *dynamic laws*
  - E.g., two people are likely to know each other if they have a common friend



  - In this case, ♠ $\rightarrow$ ★ $\rightarrow$ ♣ within the time range $0 \leq t \leq t_3$ is a dynamic graph motif that describe the law that ♠ is likely to interact with ♣ at a certain time
  - In this work, we mainly focus on answering two research questions: (1) How to extract diverse and expressive motifs and (2) how to encode these motifs to learn effective node representations

Preliminaries

- We define a continuous-time dynamic graph as a stream of temporal interactions
  - For simplicity, we assume these temporal interactions are without node and edge attributes in the following slides
  - Our method can be easily extended to learn on attributed CTDGs

Continuous-Time Dynamic Graph

A CTDG is defined as $\mathcal{G} = \{(e_i, t_i)\}_{i=1}^{N}$, where each interaction has two nodes at a specific time, e.g., $(e_i, t_i) := (\{u_i, v_i\}, t_i)$, $t_i \in \mathbb{R}^+$

Preliminaries

- As dynamic graph motifs reflect certain dynamic laws in a CTDG, it is desirable to characterize a temporal node with its surrounding motifs

Dynamic Graph Motif

Given a CTDG $\mathcal{G}$, we define a motif as a subset of temporal nodes with their interactions within a defined time range, i.e., $0 \leq t \leq q$.

Preliminaries

- *Temporal walks* rooted at a node can be regarded as its surrounding dynamic graph motifs after *walk anonymization*

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, i.e., node w_i at time t_i with $w_0 := u$ and $t_0 := t$:

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

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

Preliminaries

- *Temporal walks* rooted at a node can be regarded as the surrounding dynamic graph motifs of this node after the *walk anonymization*

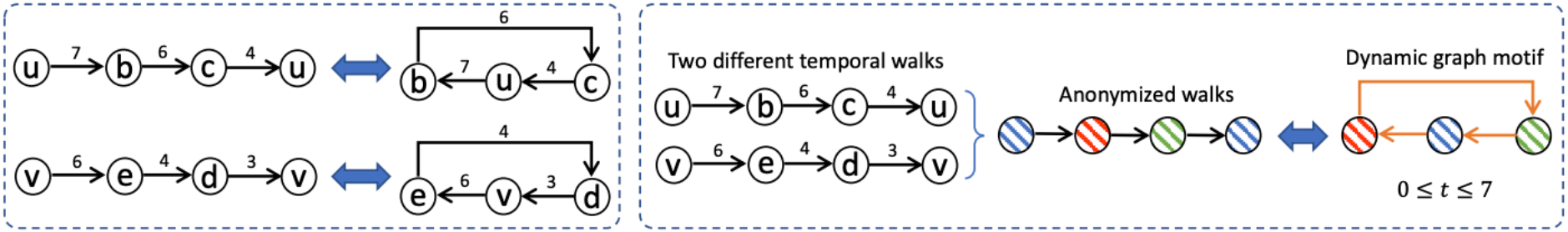


Figure: Two example temporal walks form two different triadic closures but represent the same motif within the time range $0 \le t \le 7$.

Remark 1

A dynamic graph motif has one or more instantiations, which are temporal walks

Preliminaries

- *Temporal walks* rooted at a node can be regarded as its surrounding dynamic graph motifs after *walk anonymization*

(Simplified) Walk Anonymization

Given a temporal node w and a walk W, the anonymization operator $A(\cdot)$ is defined as follows:

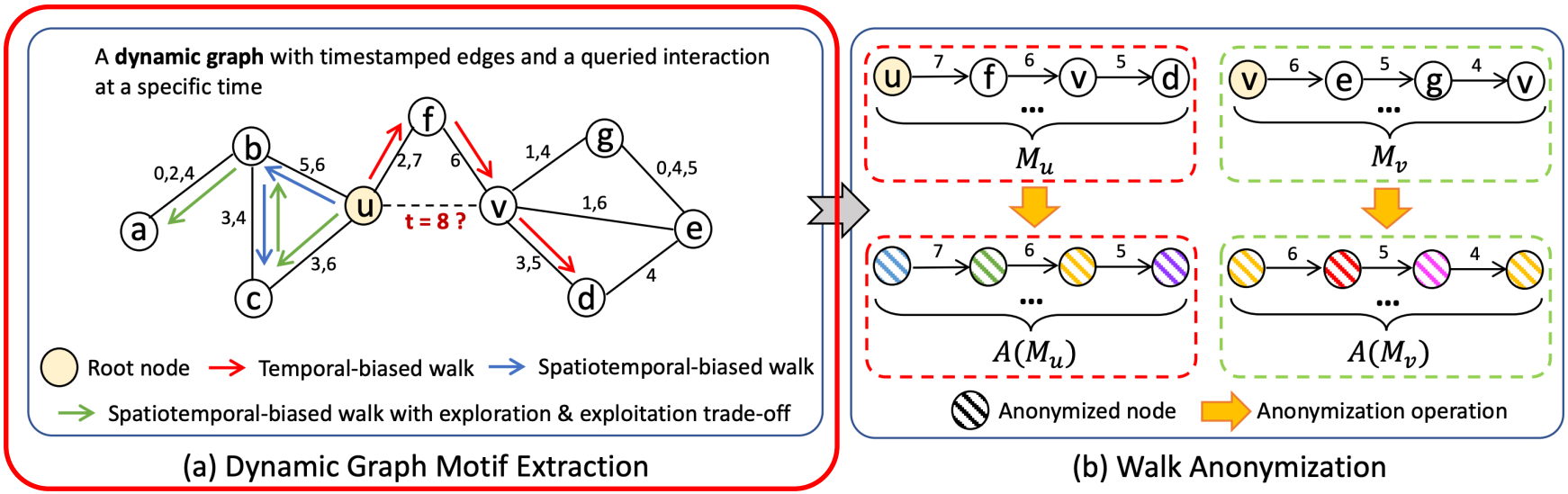
$$A(w; W) = |\{v_0, \cdots, v_{i*} \mid v_i \in W\}|, \text{where } i^* \text{ is the smallest index s.t. } v_{i*} = w.$$

Remark 2

A valid temporal walk can be generalized to a specific dynamic graph motifs by removing temporal node identities

Methodology: Walk Sampling

- **Motif extraction:** Firstly, we consider not only *temporal* but also *spatial* constrains when sampling walks



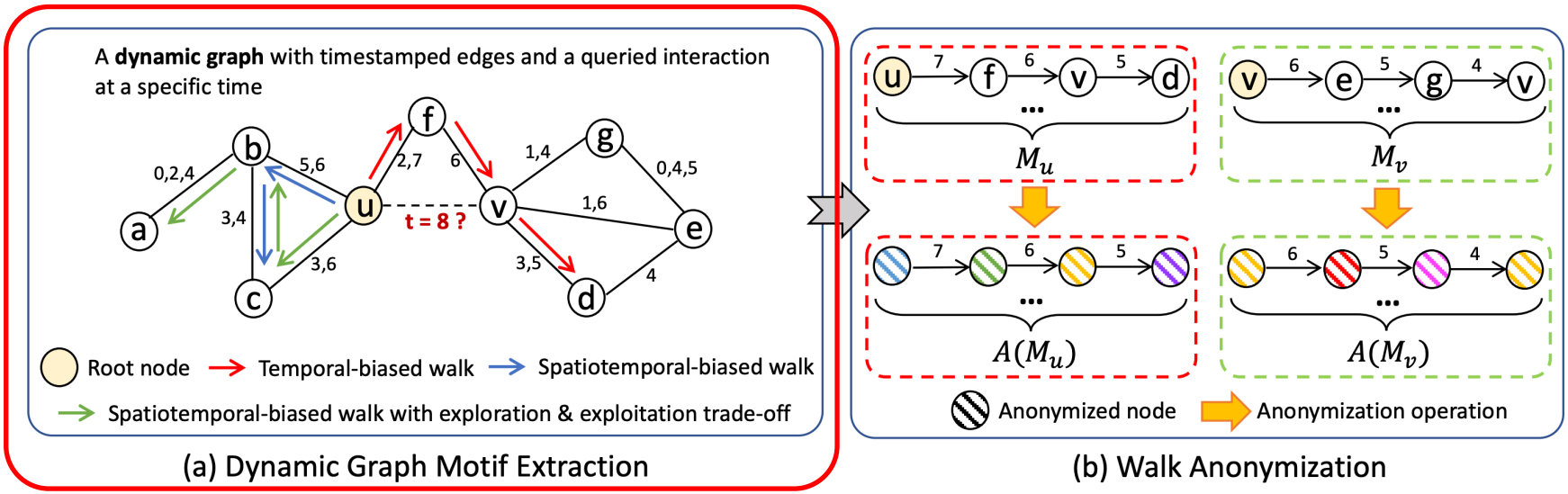
Temporal-biased Walk Sampling

Most-recent neighbors should be allocated a larger sampling probability since they are typically more informative w.r.t. a node at time t:

$$Pr_t(a) = \frac{exp(\alpha(t_a - t))}{\sum_{a' \in \mathcal{G}_{u,t}} exp(\alpha(t_{a'} - t))}$$

Methodology: Walk Sampling

- **Motif extraction:** Firstly, we consider not only *temporal* but also *spatial* constrains when sampling walks



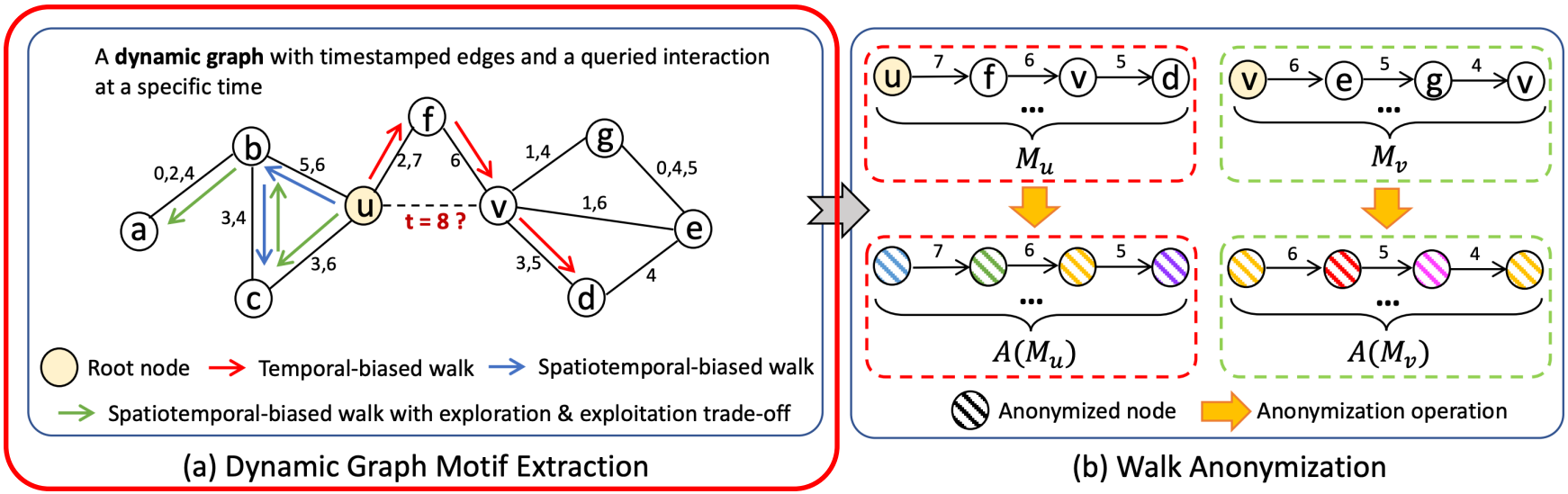
Spatial-biased Walk Sampling

Neighbors with higher connectivity (e.g., node degree $d_a = |\mathcal{G}_{a,t'}|$) need to be emphasized to allow exploring more diverse and potentially expressive motifs:

$$Pr_s(a) = \frac{exp(-\beta/d_a)}{\sum_{a' \in \mathcal{G}_{u,t}} exp(-\beta/d_{a'})}$$

Methodology: Walk Sampling

- **Motif extraction:** We also consider *tree traversal properties* to avoid sampling too much homogeneous motifs



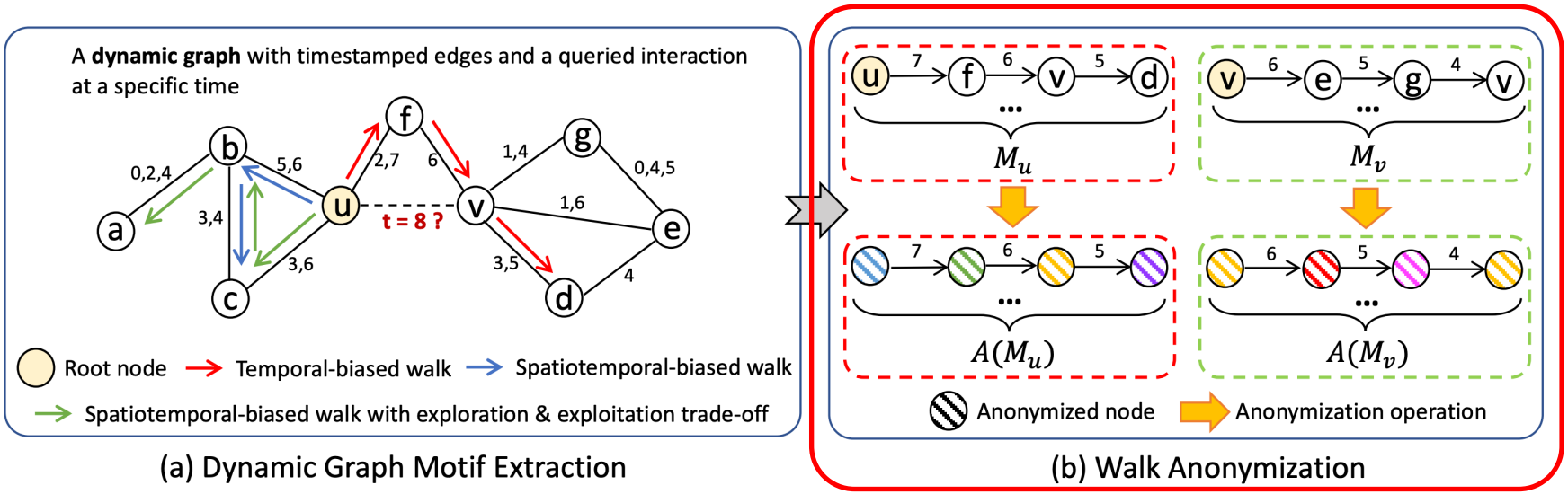
Exploitation & Exploration Trade-Off

If a temporal neighbor has been sampled s_a times, its sampling probability in the next turn is inversely proportional to s_a:

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

Methodology: Walk Anonymization

- **Motif generation:** Walk anonymization replaces node identities with position encodings (aka relative identities), which injects structural information while maintaining the inductiveness of our method



(a) Dynamic Graph Motif Extraction

(b) Walk Anonymization

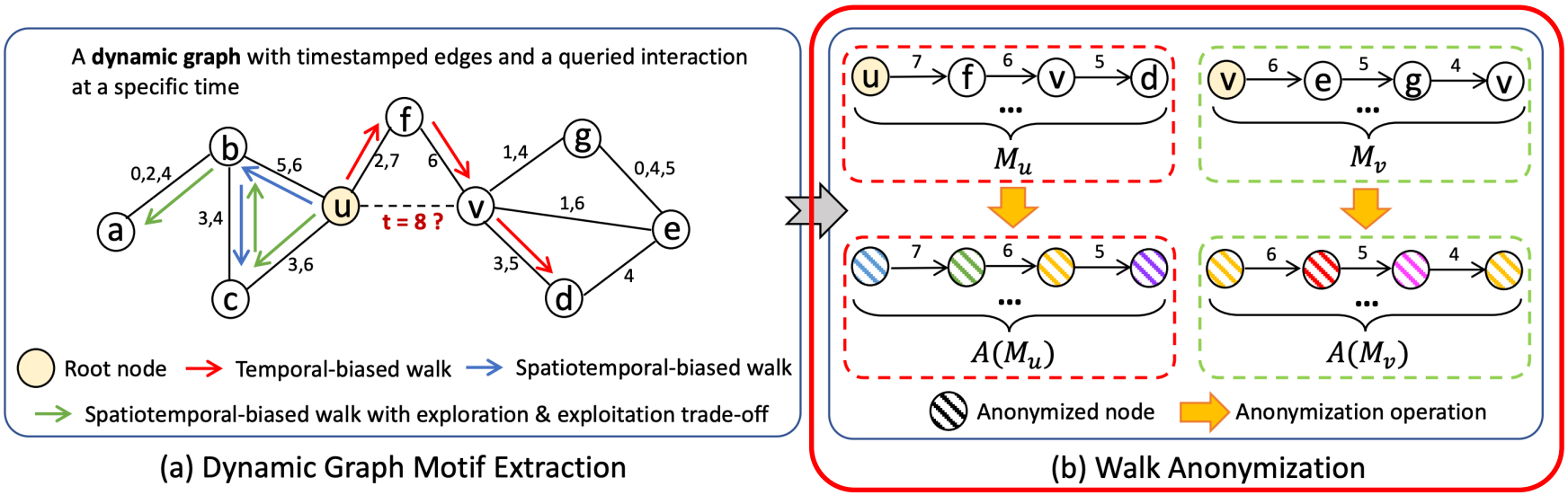
Unitary Anonymization

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:

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

Methodology: Walk Anonymization

- **Motif generation:** Walk anonymization replaces node identities with position encodings (aka relative identities), which injects structural information while maintaining the inductiveness of our method



(a) Dynamic Graph Motif Extraction

(b) Walk Anonymization

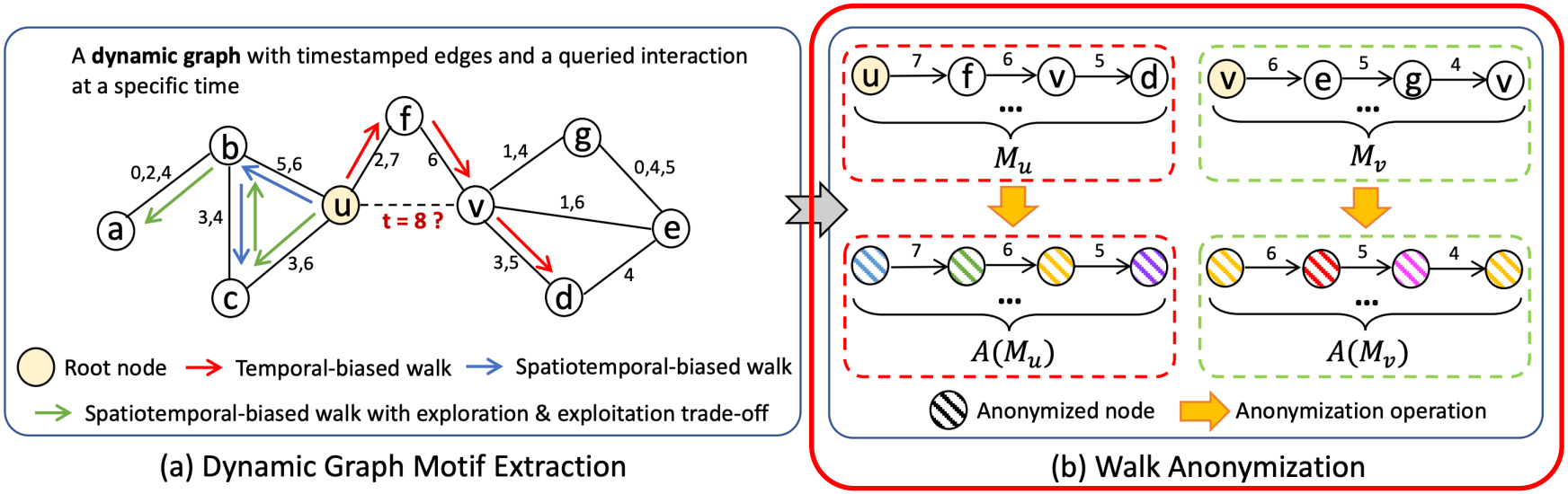
Binary Anonymization

Establishing the connections between $W \in M_u \cup M_v$ (i.e., unifying the name spaces between $A(w; M_u)$ and $A(w; M_v)$) may be beneficial for edge-level tasks:

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

Methodology: Walk Anonymization

- **Motif generation:** Walk anonymization replaces node identities with position encodings (aka relative identities), which injects structural information while maintaining the inductiveness of our method



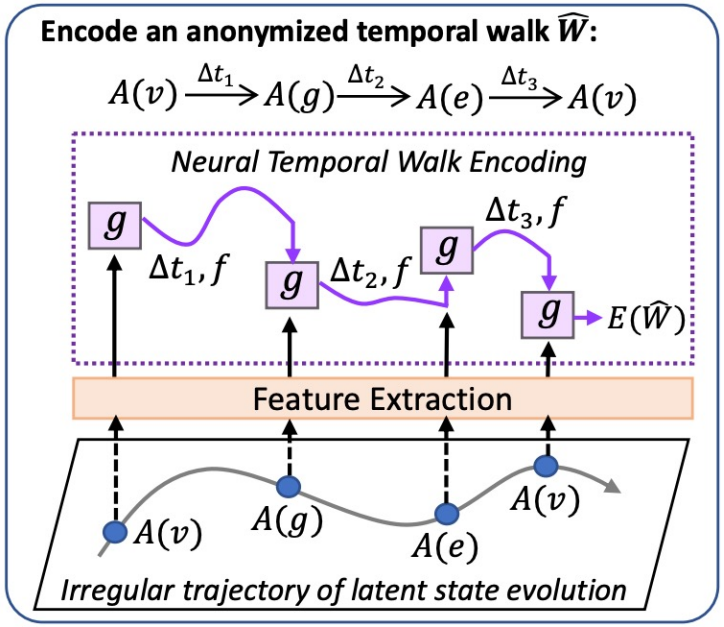
(a) Dynamic Graph Motif Extraction

(b) Walk Anonymization

- We transform a temporal walk $W = \{(w_i, t_i) \mid (w_i, t_i) \in W \text{ for } i = 0, \cdots, l\}$ to a dynamic graph motif $\widehat{W} = \{(A(w_i), t_i) \mid (w_i, t_i) \in W \text{ for } i = 0, \cdots, l\}$
- $A(w_i)$ can be either unitary or binary anonymization

Methodology: Neural Motif Encoding

- 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



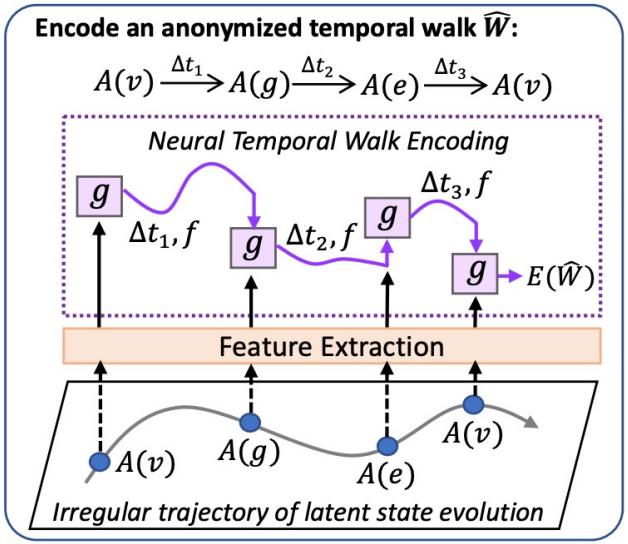
Algorithm 2 Neural Temporal Walk Encoding

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: $\quad h_i^{'} = \text{ODESolve}(h_{i-1}, f_\theta, t_{i-1}, t_i)$
5: $\quad A^{'}(w_i) = \text{MLP}_\psi(A(w_i))$
6: $\quad h_i = g_\phi(h_i^{'}, A^{'}(w_i))$
7: **end for**
8: **return** The walk embedding h_l

- Specifically, our method consists of two interleaving steps: Continuous evolution and instantaneous activation

Methodology: Neural Motif Encoding



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: $\quad h_i^{'} = \text{ODESolve}(h_{i-1}, f_\theta, t_{i-1}, t_i)$
5: $\quad A^{'}(w_i) = \text{MLP}_\psi(A(w_i))$
6: $\quad h_i = g_\phi(h_i^{'}, A^{'}(w_i))$
7: **end for**
8: **return** The walk embedding h_l

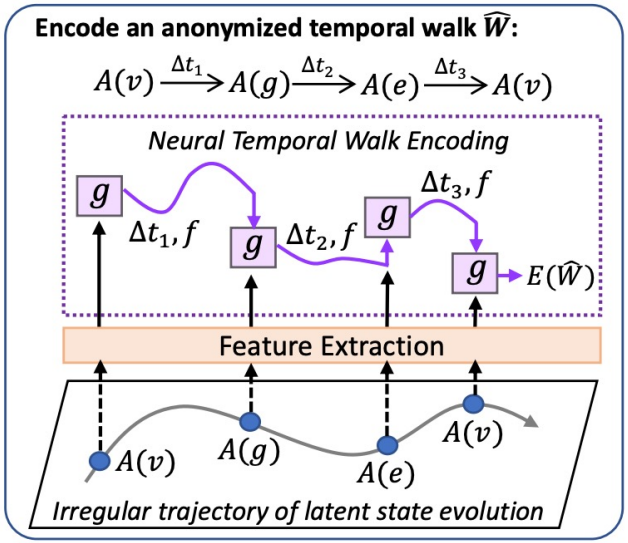
Continuous Evolution

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:

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

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.

Methodology: Neural Motif Encoding



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: $\quad h_i^{'} = \text{ODESolve}(h_{i-1}, f_\theta, t_{i-1}, t_i)$
5: $\quad A^{'}(w_i) = \text{MLP}_\psi(A(w_i))$
6: $\quad h_i = g_\phi(h_i^{'}, A^{'}(w_i))$
7: **end for**
8: **return** The walk embedding h_l

Instantaneous Activation

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

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

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 $\widehat{W}$.

Methodology: CL-based Optimization

- Here, we introduce a harder contrastive pretext task than other works. Our task aims to maximize the mutual information between interacting temporal nodes while pushing other irrelevant nodes away

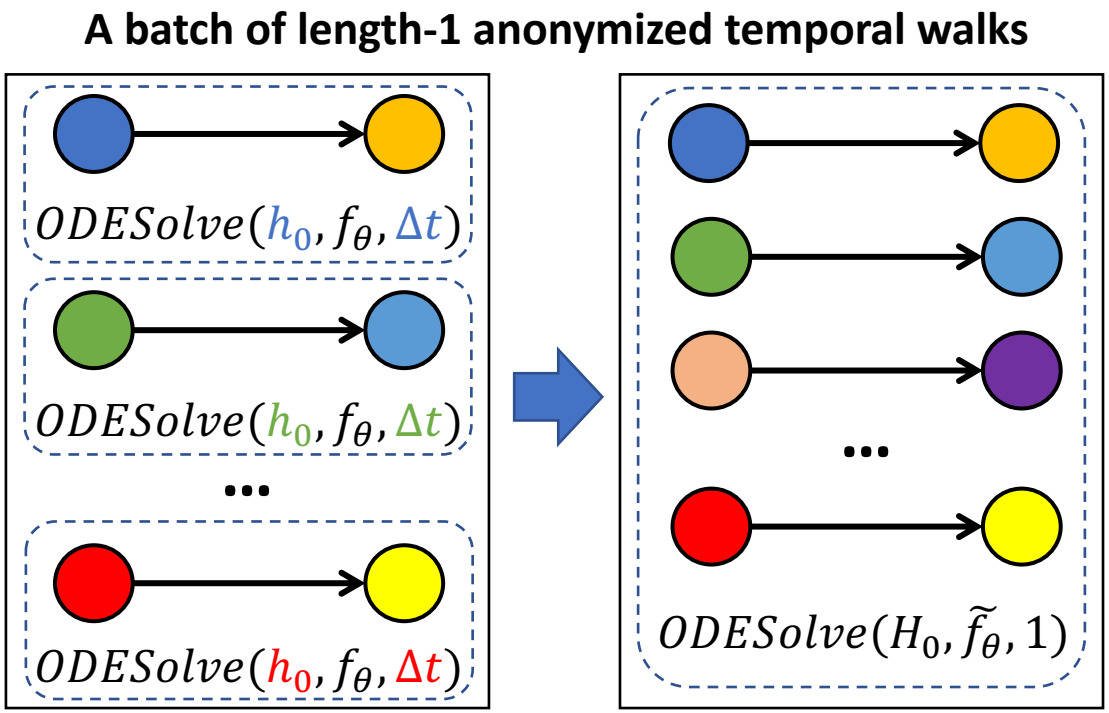
Learning Objective

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

$\text{sim}(\cdot)$ is a similarity function defined as $\text{sim}(\overline{h}_u, \overline{h}_v) = \sigma\big(\text{MLP}(\overline{h}_u, \overline{h}_a, \xi)\big)$, where $\sigma(\cdot)$ and ξ are sigmoid activation and trainable parameters.

Technical Challenge 1: Batching for Scalability

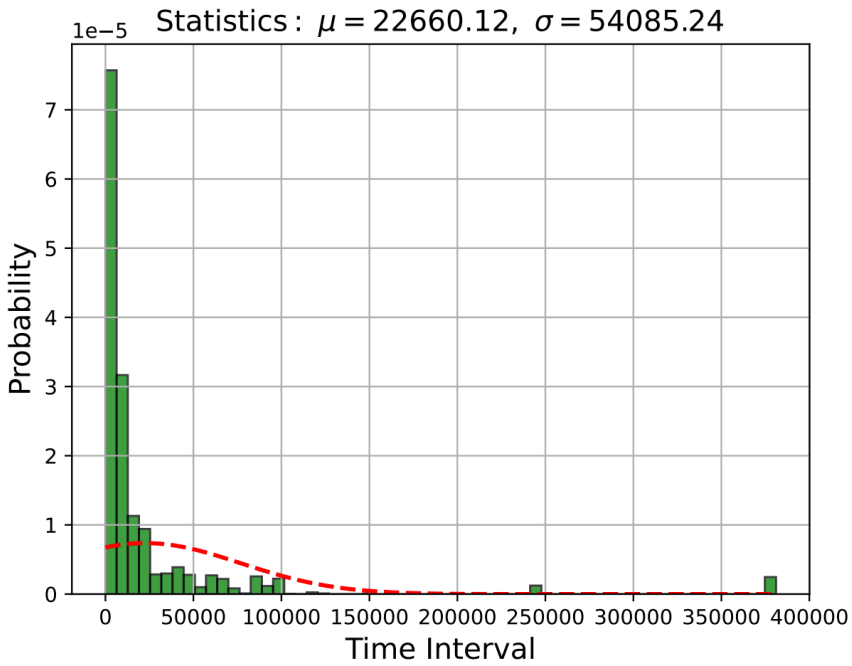
- We employ a "substitute variable" trick to solve a batch of neural ODEs instead of solving them one by one (see Appendix B.3 for details)



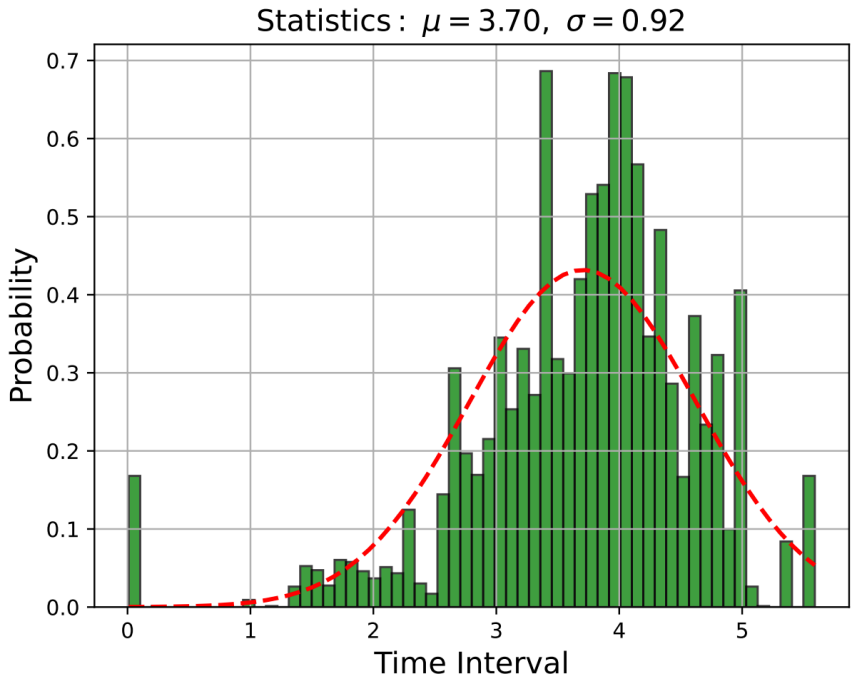
- Unifying the integral time among all ODEs to the same range, resulting in a lower time complexity $\mathcal{O}(1)$ instead of $\mathcal{O}(B)$ in the above example

Technical Challenge 2: Time Interval Normalization

- Another challenge is how to make the solving of continuous evolution processes tractable when facing very large time intervals. (See Appendix B.3)



(a) Distribution of raw time intervals in seconds



(b) Distribution of logarithmically scaled time intervals with the base 10

- Retaining the relative differences between small and large time intervals is the key to preserving critical temporal patterns in dynamic graph modeling

Experimental Results

Main Results: Temporal Link Prediction

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	0.7703 ± 0.032
		CAWs [36]	0.9002 ± 0.002	0.9520 ± 0.002	0.8719 ± 0.001	0.6948 ± 0.053
		NeurTWs	0.9526 ± 0.002	0.9564 ± 0.005	**0.9100 ± 0.014**	**0.7756 ± 0.031**
		NeurTWs†	**0.9750 ± 0.004**	**0.9704 ± 0.012**	0.8911 ± 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	**0.9612 ± 0.002**	0.8744 ± 0.004	0.7479 ± 0.023
		NeurTWs	0.9575 ± 0.011	0.9525 ± 0.002	**0.9316 ± 0.018**	**0.7822 ± 0.004**
		NeurTWs†	**0.9699 ± 0.010**	0.9566 ± 0.007	0.9037 ± 0.013	0.7772 ± 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	0.9777 ± 0.001	0.8762 ± 0.004	0.7558 ± 0.036
		NeurTWs	0.9649 ± 0.008	**0.9906 ± 0.007**	**0.9242 ± 0.005**	**0.8329 ± 0.010**
		NeurTWs†	**0.9768 ± 0.008**	0.9858 ± 0.015	0.9140 ± 0.013	0.8302 ± 0.007

Main Results: Temporal Node Classification

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]	0.8781 ± 0.003	**0.6706 ± 0.009**
NeurTWs	**0.8851 ± 0.003**	0.6652 ± 0.022

- Our method surpasses the strongest baseline by up to 8% in transductive or inductive temporal link prediction tasks
- In addition, our approach achieves the best or on-par performances on temporal node classification tasks

Ablation Study: Main Results

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

- Spatiotemporal-biased walk sampling is highly preferred, and incorporating traversal properties can provide significant benefits on certain datasets
- The proposed continuous evolution process is essential for embedding anonymized walks that include irregularly-sampled temporal nodes
- Our contrastive learning objective provides general improvements, although they may not be very substantial

Ablation Study: Modeling Temporal Dependencies

Table 10: Study on different strategies to model temporal dependencies in temporal walk encoding (Section 4.3). The performance in predicting *all inductive* interactions is reported.

Configuration	CollegeMsg		Taobao	
	AUC	AP	AUC	AP
Standard RNN	0.868 ± 0.02	0.898 ± 0.01	0.860 ± 0.04	0.901 ± 0.02
RNN with exponential decay	0.915 ± 0.03	0.925 ± 0.03	0.923 ± 0.01	0.920 ± 0.01
RNN with time encoding	0.910 ± 0.02	0.903 ± 0.01	0.889 ± 0.01	0.906 ± 0.02
Continuous evolution	**0.958 ± 0.01**	**0.966 ± 0.01**	**0.938 ± 0.02**	**0.933 ± 0.02**

- Standard RNNs perform poorly because they fail to consider the crucial time interval information
- Is using time encoding techniques the only solution for modeling temporal dependencies? The answer is no
- Our approach produces dominant results by significantly outperforming the best available techniques, i.e., time encoding and exponential time decay

Parametric Sensitivity

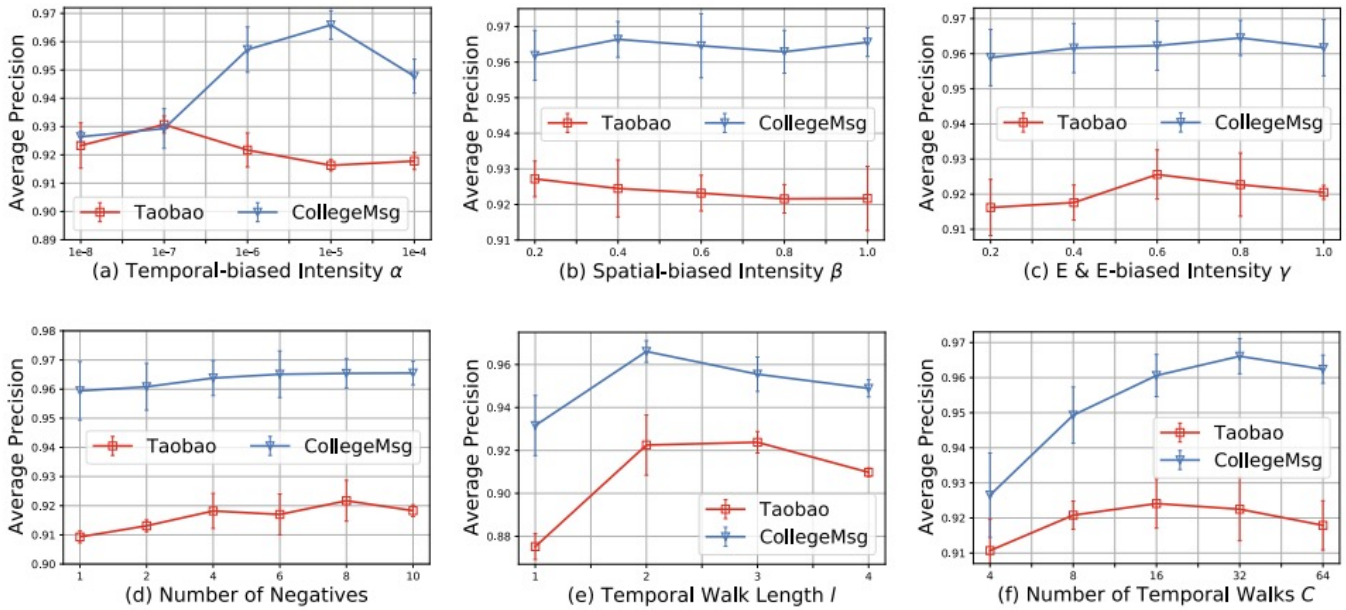
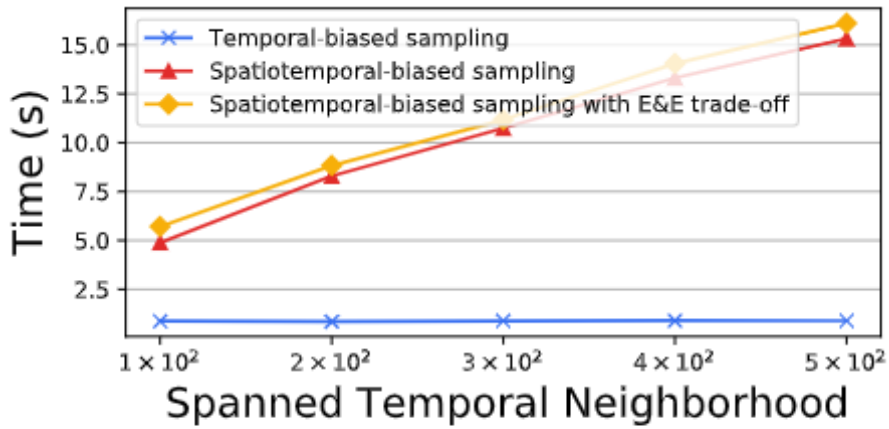


Figure 5: Study on important settings of `NeurTWs`. The performance in predicting *all inductive* interactions is reported.

- For each dataset, there are optimal balances between the intensities of three sampling biases
- In most cases, sampling 16 or 32 walks with a length of 2 or 3 is sufficient to characterize a temporal node
- Increasing the number of negative samples can be beneficial, but it comes at the cost of increased model complexity

Limitations

- Calculating spatial-biased probabilities can be computationally intensive, though limiting the number of spanned temporal neighbors can help alleviate the burden on computation



(a) Average walk sampling runtime in a batch w.r.t. the number of spanned temporal neighborhood.

- A more sophisticated time interval normalization strategy is required. Although we propose a simple solution based on logarithmic transformations, there is no theoretical guarantee of stability when solving the continuous evolution process with this normalization trick

Summary

Summary

- 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
- We introduce a new perspective to encode dynamic graph motifs composed of irregularly-sampled temporal nodes, explicitly and better modeling the underlying spatiotemporal dynamics
- We integrate contrastive learning into dynamic graph modeling to enrich supervision signals, which lifts the learning ability of our model

References I

Chen, R. T., Rubanova, Y., Bettencourt, J., and Duvenaud, D. K. (2018). Neural ordinary differential equations. *Advances in neural information processing systems*, 31.

Gilmer, J., Schoenholz, S. S., Riley, P. F., Vinyals, O., and Dahl, G. E. (2017). Neural message passing for quantum chemistry. In *International conference on machine learning*, pages 1263–1272. PMLR.

Kumar, S., Zhang, X., and Leskovec, J. (2019). 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.

Nguyen, G. H., Lee, J. B., Rossi, R. A., Ahmed, N. K., Koh, E., and Kim, S. (2018). Continuous-time dynamic network embeddings. In *Companion proceedings of the the web conference 2018*, pages 969–976.

Rossi, E., Chamberlain, B., Frasca, F., Eynard, D., Monti, F., and Bronstein, M. (2020). Temporal graph networks for deep learning on dynamic graphs. In *ICML 2020 Workshop on Graph Representation Learning*.

References II

Sankar, A., Wu, Y., Gou, L., Zhang, W., and Yang, H. (2020). 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.

Trivedi, R., Farajtabar, M., Biswal, P., and Zha, H. (2019). Dyrep: Learning representations over dynamic graphs. In *International conference on learning representations*.

Wang, Y., Chang, Y.-Y., Liu, Y., Leskovec, J., and Li, P. (2021). Inductive representation learning in temporal networks via causal anonymous walks. In *International Conference on Learning Representations*.

Xu, D., Ruan, C., Korpeoglu, E., Kumar, S., and Achan, K. (2020). Inductive representation learning on temporal graphs. In *International Conference on Learning Representations (ICLR)*.

NEURAL INFORMATION PROCESSING SYSTEMS

MONASH University

Griffith UNIVERSITY

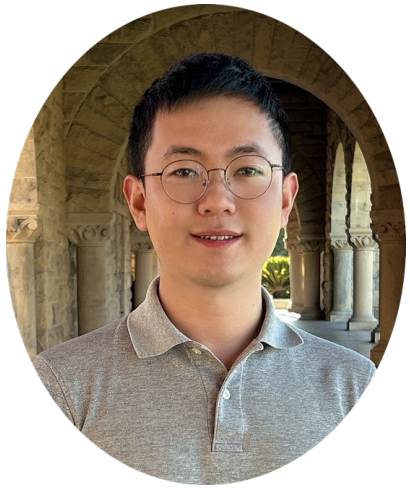
Neural Temporal Walks: Motif-Aware Representation Learning on Continuous-Time Dynamic Graphs

Ming Jin

Department of Data Science & AI

Faculty of IT, Monash University

- Email: ming.jin@monash.edu
- Page: https://mingjin.dev/



Thank You