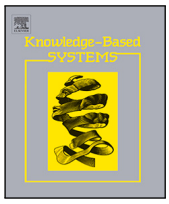
Knowledge-Based Systems 320 (2025) 113661

Contents lists available at ScienceDirect

Knowledge-Based Systems

journal homepage: www.elsevier.com/locate/knosys



Dynamic graph representation learning via edge temporal states modeling and structure-reinforced transformer

Shengxiang Hu [a], Guobing Zou [a,*], Song Yang [a], Shiyi Lin [a], Yanglan Gan [b], Bofeng Zhang [c]

[a] School of Computer Engineering and Science, Shanghai University, Shanghai, 200444, China
[b] School of Computer Science and Technology, Donghua University, Shanghai, 201620, China
[c] School of Computer and Information Engineering, Shanghai Polytechnic University, Shanghai, 201209, China

ARTICLE INFO

Keywords:
Dynamic graph
Graph representation learning
Edge temporal states
Structure-reinforced graph transformer

ABSTRACT

The rapid proliferation of time-evolving networks has rendered dynamic graph representation learning increasingly crucial for real-world applications, as existing approaches that combine recurrent neural networks (RNNs) with graph neural networks (GNNs) face two critical limitations: insufficient modeling of edge temporal states and their impact on node feature evolution, along with the inherent over-smoothing problem of GNNs that impedes effective extraction of global structural features. To address these challenges, we introduce the **R**ecurrent **S**tructure-reinforced **G**raph **T**ransformer (RSGT), a novel framework that advances dynamic graph representation learning through two key innovations. First, it introduces a principled approach to explicitly model edge temporal states using differentiated edge types and weights derived from sequential snapshot analysis, effectively integrating temporal dynamics into the graph's topological structure. Second, it designs a structure-reinforced graph transformer that leverages a recurrent learning paradigm to capture comprehensive node representations, simultaneously encoding both local connectivity patterns and global structural features while preserving temporal evolution characteristics. Comprehensive experiments on four real-world datasets demonstrate RSGT's superior performance in discrete dynamic graph representation learning, consistently outperforming existing methods in dynamic link prediction tasks.

1. Introduction

Graphs serve as powerful abstractions for modeling complex systems across diverse domains including traffic flow forecasting [1,2], recommender systems [3–5] and stock prediction [6], wherein entities and their interactions naturally evolve over time through emergence or dissolution of connections. These temporal variations necessitate the conceptualization of dynamic graphs, which can be categorized as either discrete [7,8] or continuous [9,10] based on their temporal modeling approach. Learning the representation of these dynamic graphs is crucial as it encapsulates both interaction mechanisms and evolutionary patterns, thereby facilitating system behavior understanding and prediction.

While static graph representation learning [11,12] primarily focuses on fixed topological features, dynamic graph representation learning confronts the dual challenge of preserving evolving structures while capturing temporal dynamics, significantly increasing model design complexity. This necessitates architectures capable of simultaneously modeling spatial dependencies and temporal evolution patterns while maintaining structural integrity at each time step. Our study concentrates on discrete dynamic graph representation learning due to its widespread applicability [7,8] and efficacy in capturing significant state changes in real-world systems where graph structures typically evolve through distinct, observable transitions rather than continuous changes.

A discrete dynamic graph is modeled as a sequence of ordered static snapshots across discrete time intervals. The predominant approach combines Graph Neural Networks (GNNs) with Recurrent Neural Networks (RNNs) [13,14] to capture both structural and temporal patterns. However, this approach faces fundamental challenges in maintaining the balance between structural and temporal feature learning [7], often leading to suboptimal representation quality. Recent research has explored more sophisticated architectures to address these limitations [15,16], particularly focusing on better handling of long-range dependencies and improved inductive learning capabilities. Emerging research trends focus on developing methods that can effectively

* Corresponding author.
E-mail address: gbzou@shu.edu.cn (G. Zou).

https://doi.org/10.1016/j.knosys.2025.113661
Received 1 October 2024; Received in revised form 9 April 2025; Accepted 25 April 2025
Available online 10 May 2025

balance the modeling of both short-term and long-term temporal dependencies [7], while also addressing the challenges of scalability and computational efficiency in large-scale dynamic graphs [17]. Contemporary advancements aim to yield more comprehensive and nuanced dynamic graph representations that simultaneously capture localized interactions and global evolutionary patterns.

However, existing approaches exhibit limitations that constrain their effectiveness in dynamic graph representation learning. Firstly, current methods often overlook the dynamic influence of edge temporal states on node features. The temporal evolution of edges — their persistence, emergence, or disappearance — significantly impacts node characteristics, yet many approaches fail to explicitly model these dynamics [18,19]. Secondly, while recent research [20,21] has begun to explore transformer architectures to address the over-smoothing problem in GNNs [22,23], challenges remain in effectively integrating graph structural information with complex temporal dependencies. Moreover, existing RNN-based approaches typically process temporal and structural information in a segregated manner, first learning node representations within static snapshots and then modeling their temporal evolution. This artificial separation leads to the loss of valuable temporal information during neighborhood aggregation, as the temporal evolution of neighboring nodes is not directly considered in the feature aggregation process. Consequently, there is a need for an approach that can maintain an integrated view of temporal–structural dependencies while effectively preserving node distinctiveness.

To address these limitations, we propose a novel framework for dynamic graph representation learning, termed **R**ecurrent **S**tructure-reinforced **G**raph **T**ransformer (RSGT). At the core of RSGT lies an innovative graph transformation technique that converts temporal snapshots into weighted multi-relation graphs, wherein edge temporal states are explicitly encoded through distinct edge types and weights, effectively capturing inter-snapshot variations. Building upon this transformed representation, we introduce a structure-reinforced graph Transformer with a novel structure-aware attention mechanism that intrinsically integrates topological information into the attention computation process. This architectural enhancement enables simultaneous modeling of both intricate graph structures and temporal dynamics, while selectively emphasizing significant historical patterns and structural relationships encoded in edge attributes. Through a recurrent learning process, our framework iteratively refines node representations by integrating both temporal and structural information, effectively capturing the complex evolution patterns in dynamic graphs.

Our extensive empirical evaluations on dynamic link prediction tasks across four real-world datasets demonstrate RSGT's superior performance in dynamic graph representation learning, highlighting its potential for dynamic graph analysis. The key contributions of this research can be summarized as follows:

- We propose RSGT, a novel recurrent representation learning framework that transforms temporal graph snapshots into weighted multi-relational difference graphs, effectively capturing edge temporal states and their integration with topological structures to comprehensively represent dynamic graph evolution.
- We design a structure-reinforced graph transformer that concurrently extracts local–global topological features and temporal dynamics, mitigating the over-smoothing problem while enhancing the model's capacity to capture complex temporal–structural dependencies.
- Extensive experiments on real-world datasets demonstrate RSGT's superior performance in dynamic link prediction tasks, validating its effectiveness in capturing intricate dynamic patterns and its potential for advancing dynamic graph analysis applications.

The remainder of this paper is organized as follows: Section 2 provides preliminaries and problem formulations. Section 3 details the proposed RSGT framework. Experimental results are presented in Section 4. Related work is reviewed in Section 5, followed by conclusions in Section 6.

2. Problem formulation

Here, we establish a formal mathematical framework for dynamic graph representation learning through a series of definitions that form the theoretical foundation of our approach.

Definition 1 (*Dynamic Graph*)**.** A discrete dynamic graph, denoted as $\mathcal{G}$, is an ordered sequence of static graphs represented as $\{\mathcal{G}_1, \ldots, \mathcal{G}_t\}$. Here, $\mathcal{G}_t = \langle V_t, E_t \rangle$ corresponds to a snapshot of the dynamic graph $\mathcal{G}$ at time slice t, where V_t is the set of nodes, E_t is the set of edges. An edge $e_{ij}^t \in E_t$ denotes a link between nodes v_i, v_j at discrete time point t.

The temporal evolution of a dynamic graph is manifested primarily through changes in its edge structure. While the dynamic graph captures the overall structure at each time step, it is crucial to understand how individual edges evolve over time. This leads us to the concept of edge temporal states:

Definition 2 (*Edge Temporal States*)**.** The edge temporal states, denoted as S_t, characterize the evolutionary patterns of edges across temporal transitions in the graph. For each edge $e_{ij}^t \in E_t$, its temporal state $s_{ij}^t \in S_t$ is defined as a tuple $s_{ij}^t = \langle tp_{ij}^t, \omega_{ij}^t \rangle$. Here, tp_{ij}^t represents the current state of the edge, which can be categorized as persistent (the edge exists at both $t-1$ and t), emerging (the edge appears at t but did not exist at $t-1$), or disappearing (the edge existed at $t-1$ but no longer exists at t). The second component, ω_{ij}^t, quantifies the temporal persistence of the current state, measured by the number of consecutive time slices during which the edge has maintained its present state.

While dynamic graphs and edge temporal states provide a framework for describing evolving systems, effectively analyzing these systems requires learning compact representations that capture both structural and temporal information. This leads us to the task of dynamic graph representation learning:

Definition 3 (*Dynamic Graph Representation Learning*)**.** Dynamic graph representation learning involves constructing a mapping function that projects the high-dimensional dynamic graph data into a low-dimensional representation $\mathbf{H}^t \in \Re^{|V_t| \times d}$ ($d \ll |V_t|$) by utilizing a succession of snapshots $\{\mathcal{G}_1, \ldots, \mathcal{G}_t\}$ and their corresponding edge temporal states $\{S_1, \ldots, S_t\}$ up to time slice t. The ith row of $\mathbf{H}^t$, denoted as $\mathbf{h}_i^t$, represents the derived representation of node v_i at time slice t, simultaneously preserving both the network topology and the evolutionary trajectory of the node. A generalized learning framework can be formulated as:

$$\mathbf{H}^t = g(f(\mathcal{G}_t, S_t, \mathbf{X}|\Theta_f), \mathbf{I}_{t-1}|\Theta_g) \tag{1}$$

where $\mathbf{X}$ denotes the initial node features or embedding matrix, $\mathbf{I}_{t-1}$ represents the most recent latent temporal state, f functions as a feature extraction mechanism that captures the intrinsic structural and temporal characteristics, while g serves as a sequence model for discerning evolving patterns. Θ_f and Θ_g are the learnable parameters of f and g respectively.

The effectiveness of node representations learned through this process is crucial for various graph-oriented tasks [10,24,25]. Among these applications, dynamic link prediction serves as a particularly rigorous benchmark for evaluating the quality of temporal–structural representations:

Definition 4 (*Dynamic Link Prediction*)**.** Given a series of snapshots $\{\mathcal{G}_1, \ldots, \mathcal{G}_t\}$ up to time slice t, the dynamic link prediction task entails estimating the probability of existence for a candidate edge e_{ij}^{t+1} in the upcoming snapshot $\mathcal{G}_{t+1}$ at time slice $t+1$. This can be formulated as:

$$p_{ij}^{t+1} = \tau(\mathcal{G}_{t+1}|\mathcal{G}_1, \ldots, \mathcal{G}_t, S_1, \ldots, S_t; \Theta_\tau) \tag{2}$$

where p_{ij}^{t+1} represents the predicted probability that $e_{ij} \in E_{t+1}$, and τ denotes the predictive model that transforms the historical graph sequence into probabilistic edge forecasts, parameterized by Θ_τ.

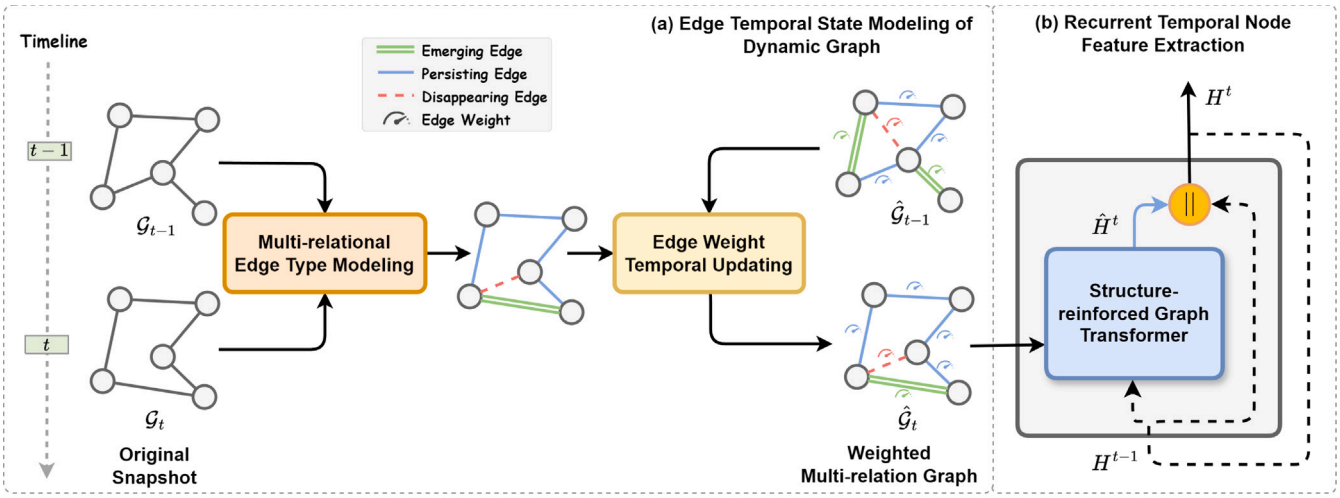


Fig. 1. Overall framework of the proposed Recurrent Structure-reinforced Graph Transformer (RSGT). (a) Edge Temporal State Modeling of Dynamic Graph: transforming each graph snapshot into a weighted multi-relation graph to model the edge temporal states. (b) Recurrent Temporal Node Feature Extraction: capturing both graph topology and evolving dynamics through a recurrent learning paradigm with the structure-reinforced graph transformer.

3. Approach

Fig. 1 illustrates the architecture of our proposed Recurrent Structure-reinforced Graph Transformer (RSGT), which comprises two complementary modules designed to address the challenges of dynamic graph representation learning. The first module, Edge Temporal State Modeling of Dynamic Graph, systematically transforms each original graph snapshot into a weighted multi-relation difference graph by assigning diverse edge types and weights based on their evolutionary patterns between consecutive time slices. It explicitly delineates short-term dynamics through multi-relational edge type modeling while capturing long-term dependencies via an edge weight temporal updating mechanism. The second module, Recurrent Temporal Node Feature Extraction, introduces a structure-reinforced graph transformer that simultaneously integrates topological dependencies and temporal edge states across the weighted multi-relation difference graph. Through a recurrent learning paradigm, this transformer iteratively refines node representations, effectively encoding both fine-grained local structures and comprehensive global dependencies. These two modules work in tandem to learn comprehensive and time-aware node representations, enabling RSGT to effectively model the complex dynamics of evolving graphs. In the subsequent sections, we delve deeper into the intricacies of each module.

3.1. Edge temporal state modeling of dynamic graph

In a dynamic graph $\mathcal{G}$, the feature of a focal node v_i can be conceptualized as a reflection of its interactions with other nodes $v_j \in (N_i^h)^t$ that it directly or indirectly engages with [26]. Here, $(N_i^h)^t$ signifies the h-hop neighborhood of v_i at time slice t, representing all nodes that can be reached from v_i within h steps at time t.

Social network theory, as observed in [18,19], suggests that the strength of relationships between nodes predominantly depends on historical interaction patterns, particularly their frequency and duration. This relationship strength subsequently influences future interactions, playing a critical role in the formation and evolution of the graph. However, capturing these dynamic relationships presents significant challenges for node representation learning. Node interactions demonstrate various temporal behaviors — persistence, emergence, or disappearance — across consecutive snapshots, resulting in continuous transformations of edge temporal states. They are characterized by dual aspects: short-term dynamics (immediate state transitions) and long-term dynamics (cumulative influence on inter-node correlation strength). Understanding the evolution of $\mathcal{G}$ thus hinges on accurately modeling how distinct edge temporal states impact node features across different time slices.

To address the challenges and capture such multi-faceted dynamics, we propose explicitly modeling edge temporal states, which provides essential heuristic information to downstream feature extraction processes. Specifically, we develop a heuristic edge temporal state modeling method that transforms each $\mathcal{G}_t \in \mathcal{G}$ into a weighted multi-relation difference graph $\hat{\mathcal{G}}_t$ based on the differences between adjacent graph snapshots. It uses different edge types and edge weights to explicitly model the temporal states of edges, capturing both short-term and long-term dynamics. Formally, we define this graph as follows:

Definition 5 (*Weighted Multi-relation Difference Graph*)**.** A weighted multi-relation difference graph $\hat{\mathcal{G}}_t = \langle V_t, \hat{E}_t, S_t \rangle$ at time slice t is defined as a graph comprising a set of nodes V_t, a set of edges $\hat{E}_t$ encompassing connections from both current and previous time slices, and a set of edge temporal states $S_t = \{\langle tp_{ij}^t, \omega_{ij}^t \rangle\}$, where tp_{ij}^t represents the edge type capturing short-term dynamics, and ω_{ij}^t denotes the edge weight reflecting long-term dynamics. Both tp_{ij}^t and ω_{ij}^t are modeled based on the differences between adjacent time slice snapshots $\mathcal{G}_{t-1}$ and $\mathcal{G}_t$, thereby capturing the temporal dynamics of the graph structure.

For implementation, we employ two key mechanisms. First, as depicted in the left section of Fig. 1, multi-relational edge type modeling utilizes a Markov process to assign a unique type tp_{ij}^t to each edge $e_{ij}^t \in \hat{E}_t$. The value of tp is drawn from the set $\{e, p, d\}$, where e denotes an emerging edge, p symbolizes a persisting edge, and d signifies an edge that has disappeared from the previous snapshot. This process is mathematically expressed as follows:

$$\hat{E}_t = E_{t-1} \cup E_t \tag{3}$$

$$tp_{ij}^t = \begin{cases} e & \text{if } e_{ij}^t \in E_t - E_{t-1} \\ p & \text{if } e_{ij}^t \in E_t \cap E_{t-1} \\ d & \text{if } e_{ij}^t \in E_{t-1} - E_t \end{cases} \tag{4}$$

where Eq. (3) combines edges from consecutive time slices, while Eq. (4) categorizes these edges based on their temporal behavior. This categorization allows us to explicitly model the dynamic nature of edges, including those that have disappeared, which is crucial for understanding the evolving graph structure. In $\hat{\mathcal{G}}_t$, we provisionally reconstruct vanished edges between consecutive snapshots, recognizing that edge disappearance does not necessarily indicate immediate

cessation of inter-node influence, but rather may trigger diverse effects on the connected nodes.

Second, edge weight temporal updating captures the long-term dynamics by assigning and updating weights to edges based on their historical persistence and strength. To quantify the duration of edges and their impact on node relationships, we introduce a temporal counter k that tracks how long an edge has persisted in the graph. For each edge e_{ij}, we maintain and update its duration counter k_{ij}^t at each time slice t according to the edge's temporal state:

$$k_{ij}^t = \begin{cases} 1 & \text{if } tp_{ij}^t = e \\ k_{ij}^{t-1} + 1 & \text{if } tp_{ij}^t = p \\ k_{ij}^{t-1} & \text{if } tp_{ij}^t = d \end{cases} \tag{5}$$

For emerging edges ($tp_{ij}^t = e$), we initialize k_{ij}^t to 1, representing the start of a new interaction. For persisting edges ($tp_{ij}^t = p$), we increment the previous counter value by 1, reflecting the continued existence of the relationship. For disappeared edges ($tp_{ij}^t = d$), we maintain the last known duration value, preserving the historical information of the relationship strength.

Based on the edge duration counter k, we assign and update the strength coefficient ω for each edge:

$$\omega_{ij}^t = \begin{cases} \alpha k^\beta & \text{if } tp_{ij}^t = e \text{ or } p \\ \omega_{ij}^{t-1} & \text{if } tp_{ij}^t = d \end{cases} \tag{6}$$

where the parameter α (typically set between 0 and 1) controls the base weight scale, while β (typically greater than 0) determines how quickly the edge weight grows with duration. For emerging and persisting edges, the weight is computed as a function of their duration, reflecting the intuition that longer-lasting relationships tend to be stronger. For disappeared edges, we maintain their previous weight to preserve the historical relationship strength information, which may be relevant for future interactions.

Consequently, the dynamic graph $\hat{\mathcal{G}}_t$ conserves the original topology while concurrently modeling the temporal state of the edge through complementary edge types and weights, thus providing a comprehensive representation that captures both immediate transitions and cumulative effects of node interactions across time slices.

3.2. Recurrent temporal node feature extraction

The Recurrent Temporal Node Feature Extraction process is the key component of our RSGT model, designed to address the challenge of learning comprehensive node representations in dynamic graphs. Our framework adopts an integrated perspective that fundamentally differs from traditional approaches by directly incorporating the temporal trajectories of neighboring nodes during feature aggregation. This design enables node features to inherently encode both structural context and temporal evolution patterns, leading to more effective capture of dynamic graph characteristics through the natural consideration of how neighboring nodes' temporal evolution influences the central node's representation. This process proposes a structure-reinforced graph transformer (SGT) to progressively refine node representations along the temporal dimension, integrating global semantic correlations, topological dependencies, and edge temporal states across consecutive snapshots.

Fig. 1 illustrates the process of recurrent temporal node feature extraction over the constructed weighted multi-relation difference graph $\hat{\mathcal{G}}_t$. For a sequence of graph snapshots $\{\hat{\mathcal{G}}_1, \hat{\mathcal{G}}_2, \ldots, \hat{\mathcal{G}}_T\}$, the SGT updates node representations through the following process:

$$\hat{\mathbf{H}}^t = \mathcal{F}_{g_l}(\hat{\mathcal{G}}_t, \mathbf{H}^{t-1}), \quad t = 1, 2, \ldots, T \tag{7}$$

$$\mathbf{H}^t = \hat{\mathbf{H}}^t + \mathbf{H}^{t-1} \tag{8}$$

where $\mathbf{H}^t \in \Re^{|V| \times d}$ is the node feature matrix at time slice t, $\mathcal{F}_{g_l}(\cdot)$ denotes an l-layer SGT that extracts temporal–structural features, $|V|$ denotes the maximum node count across all time slices, and d represents the dimensionality of the temporal node representation. Eq. (8) implements a residual connection, enabling the model to preserve historical information while incorporating new temporal–structural patterns.

For the initial time slice ($t = 1$), we initialize node representations using $\mathbf{H}^0 = \mathbf{X} \in \Re^{|V| \times d}$, where each row represents the initial feature vector of a particular node. It can be derived from various sources, including node identifiers embedded in a d-dimensional space, node attribute embeddings encoding domain-specific features, and pre-trained embeddings that leverage existing knowledge.

Next we will detail the core operations of our SGT, as shown in Fig. 2, including global semantic encoding, structure encoding, and temporal feature extraction.

3.2.1. Global semantic encoding

The global semantic encoding module, illustrated in the upper left portion of Fig. 2, captures semantic relationships between nodes independent of their topological proximity. This global perspective enables the model to identify potential interactions between distant nodes that may become relevant as the graph evolves.

We employ self-attention to compute pairwise semantic correlations, following the standard mechanism [27]. This approach creates a complete semantic graph that complements the existing topological structure. The module takes $\mathbf{H}^{t-1}$ as input and processes it as follows:

$$\mathbf{A}^t = \frac{\mathbf{Q}^t (\mathbf{K}^t)^T}{\sqrt{d_K}} = \frac{\mathbf{H}^{t-1}\mathbf{W}_Q^t (\mathbf{H}^{t-1}\mathbf{W}_K^t)^T}{\sqrt{d_K}} \tag{9}$$

$$\mathbf{V}^t = \mathbf{H}^{t-1}\mathbf{W}_V^t \tag{10}$$

where $\mathbf{W}_Q^t, \mathbf{W}_K^t \in \Re^{d \times d'}$, and $\mathbf{W}_V^t \in \Re^{d \times d}$ are trainable projection matrices. $\mathbf{Q}^t$, $\mathbf{K}^t$, and $\mathbf{V}^t$ represent the query, key, and value matrices, respectively. The attention matrix $\mathbf{A}^t \in \Re^{|V| \times |V|}$ quantifies semantic correlations between node pairs, where $a_{ij}^t \in \mathbf{A}^t$ represents the correlation between nodes v_i and v_j. For simplicity, we assume $d' = d$ and consider single-head self-attention, though extension to multi-head attention is straightforward.

3.2.2. Structure encoding

While global semantic encoding captures overall node relationships, graph topology provides essential structural context for node representation learning [20,21]. Our structure encoding module, depicted in the bottom right of Fig. 2, enriches the transformer architecture with both local and global structural features, enhancing its ability to model complex graph dynamics.

The selection of appropriate topological features is crucial for effectively characterizing nodes within the graph structure. Drawing from fundamental network analysis principles [28,29], we incorporate node degree centrality and path-based measures, which effectively capture node connectivity patterns and structural roles. Specifically, node degree centrality reflects a node's immediate influence in the network, while the shortest path length between nodes indicates their structural proximity and potential for information exchange. Thus, for each node pair $\langle v_i, v_j \rangle$, we construct a topological attribute vector $\mathbf{attr}_s$:

$$\mathbf{attr}_{(ij)s}^t = [sd_{out}, td_{in}, spd] \tag{11}$$

where sd_{out}, td_{in}, and spd denote the out-degree of the source node, the in-degree of the target node, and the shortest path length between the nodes, respectively. Then we employ a transformation mechanism that projects this attribute vector into a higher-dimensional space, followed by a normalization step to ensure stable training dynamics:

$$\mathbf{r}_{(ij)s}^t = \text{LayerNorm}(\mathbf{W}_s \mathbf{attr}_{(ij)s}^t + \mathbf{b}_s) \tag{12}$$

where $\mathbf{W}_s \in \Re^{d_e \times 3}$ and $\mathbf{b}_s \in \Re^{d_e}$ are learnable parameters, and LayerNorm denotes layer normalization. The resulting representation $\mathbf{r}_{(ij)s}^t \in \Re^{d_e}$ encapsulates the complex topological relationships between

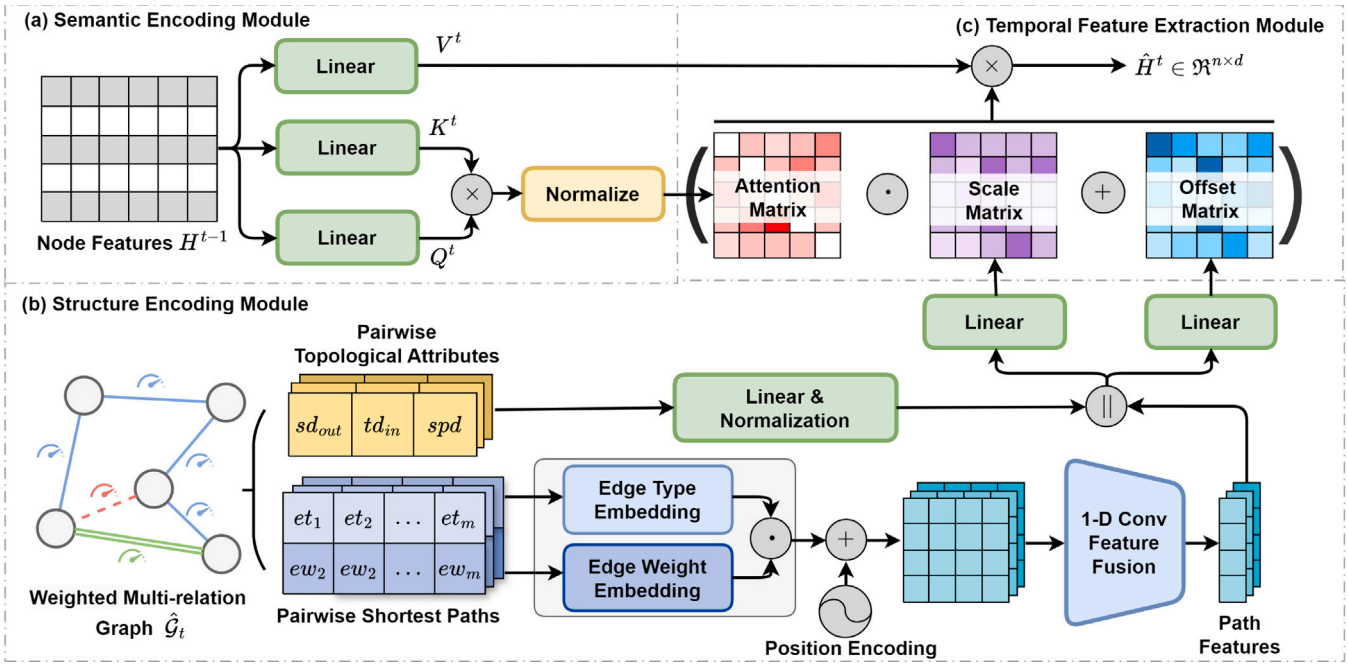


Fig. 2. Structure-reinforced Graph Transformer for temporal node feature extraction. (a) Semantic Encoding Module: capturing global semantic correlations between nodes. (b) Structure Encoding Module: extracting topological dependencies from the weighted multi-relation difference graph, incorporating both original graph structure and temporal information. (c) Temporal Feature Extraction Module: fusing semantic and enhanced topological correlations to generate time-aware node representations.

nodes, facilitating the model's ability to discern intricate structural patterns within the dynamic graph.

To additionally represent edge temporal states, we construct a path feature matrix $\mathbf{ATTR}^t_{(ij)p} \in \Re^{2\times spd}$ for each node pair, encoding both topological relationships and temporal evolution patterns:

$$\mathbf{ATTR}^t_{(ij)p} = [\mathbf{attr}^t_e; \mathbf{attr}^t_w] \tag{13}$$

where $\mathbf{attr}^t_e$ represents the edge types along the path $p^t_{v_i\to v_j}$ at time t, and $\mathbf{attr}^t_w$ represents the corresponding edge weights.

We transform these discrete attributes into continuous representations through embedding layers:

$$\mathbf{attr}^t_e \mapsto \mathbf{R}^t_e \in \Re^{spd\times d_e} \tag{14}$$

$$\mathbf{attr}^t_w \mapsto \mathbf{R}^t_w \in \Re^{spd\times d_e} \tag{15}$$

where d_e is the embedding dimension. These embeddings enable rich representations of temporal–structural patterns.

To capture the varying impact of different edge types in temporal evolution, we integrate edge type features with their corresponding weights through the Hadamard product:

$$\mathbf{R}^t_p = \mathbf{R}^t_e \odot \mathbf{R}^t_w \tag{16}$$

where $\odot$ denotes the Hadamard product.

To preserve positional information of edges along paths, we incorporate positional encoding following the transformer design [27]:

$$\mathbf{PE}_{pos,2i} = sin(\frac{pos}{10000^{2i/d_e}}) \tag{17}$$

$$\mathbf{PE}_{pos,2i+1} = cos(\frac{pos}{10000^{2i/d_e}}) \tag{18}$$

$$\hat{\mathbf{R}}^t_p = \mathbf{R}^t_p + \mathbf{PE} \tag{19}$$

where pos is the position of an edge in the path and i is the column index.

We then apply a one-dimensional convolutional layer to reduce the dimensionality of $\mathbf{R}^t_p$, resulting in the final path feature $\hat{\mathbf{r}}^t_{(ij)p} \in \Re^{d_e}$. The complete structural representation $\mathbf{r}^t_{ij} \in \Re^{2d_e}$ is formed by concatenating the topology and path features:

$$\mathbf{r}^t_{ij} = \mathbf{r}^t_{(ij)s} \oplus \hat{\mathbf{r}}^t_{(ij)p} \tag{20}$$

where $\oplus$ denotes vector concatenation.

3.2.3. Temporal feature extraction

To effectively integrate structural information into the transformer framework, we propose a structure-aware attention mechanism, as shown in the upper right section of Fig. 2. This mechanism adapts the semantic attention scores $a^t_{ij} \in \mathbf{A}^t$ based on structural features $\mathbf{r}^t_{ij}$:

$$\lambda^t_{ij} = \mathbf{W}_s(\mathbf{r}^t_{ij})^T \tag{21}$$

$$\sigma^t_{ij} = \mathbf{W}_\sigma(\mathbf{r}^t_{ij})^T \tag{22}$$

$$\hat{a}^t_{ij} = \lambda^t_{ij} a^t_{ij} + \sigma^t_{ij} \tag{23}$$

$$\hat{\mathbf{H}}^t = Normalize(\hat{\mathbf{A}}^t)\mathbf{V} \tag{24}$$

where λ^t_{ij} and σ^t_{ij} are scale and offset coefficients, respectively. $\hat{a}_{ij} \in \hat{\mathbf{A}}^t$ represents the structure-enhanced attention score. The final output $\hat{\mathbf{H}}^t \in \Re^{|V|\times d}$ thus integrates semantic correlations with temporal–structural patterns.

From a representation learning perspective, our model follows the GNN paradigm of neighborhood feature aggregation. However, RSGT addresses the over-smoothing challenge that typically arises when nodes share overlapping neighborhoods. Our structure-aware attention mechanism generates distinct aggregation weights by considering unique structural relationships (path lengths, edge weights, and types) and semantic correlations between each central node and its neighbors. Combined with residual connections (Eq. (8)), this approach preserves node distinctiveness while capturing comprehensive temporal–structural dependencies.

3.3. Model training

We train our model using dynamic link prediction as the optimization objective. For a candidate edge e^{t+1}_{ij}, we compute its feature $\mathbf{h}^{t+1}_{e_{ij}} \in \Re^d$ as the absolute difference of node features: $|\mathbf{h}^t_i - \mathbf{h}^t_j|$, following

Table 1
Statistics of the four datasets.

Dataset	# of Nodes	# of Edges	Train/Test splits
twi-Tennis	1000	40,839	100/20
CollegeMsg	1899	59,835	25/63
cit-HepTh	7577	51,315	77/1
sx-MathOF	24,818	506,550	64/15

the widely-adopted practice in graph representation learning [10,30], which provides an efficient and effective measure of node similarity. The edge features are then classified through a neural network:

$$\hat{p}_{ij}^{t+1} = \sigma(\mathbf{W}_o \mathbf{h}_{e_{ij}}^{t+1} + \mathbf{b}_o) \tag{25}$$

where $\mathbf{W}_o$ and $\mathbf{b}_o$ are trainable weights, and $\hat{p}_{ij}^{t+1}$ denotes the predicted probability of edge existence.

The model is then optimized using cross-entropy loss with L2 regularization:

$$J_{ij} = -p_{ij}^{t+1} log \hat{p}_{ij}^{t+1} - (1 - p_{ij}^{t+1}) log(1 - \hat{p}_{ij}^{t+1}) \tag{26}$$

$$J = \mathbb{E}(\sum_{i,j \in V} J_{ij}) + \lambda \|\Theta\|_2^2 \tag{27}$$

where Θ encompasses all trainable parameters, and λ controls the regularization strength.

We employ the AdamW optimizer [31] with mini-batch training, which effectively handles the structural complexity and temporal dynamics of graph data while maintaining computational efficiency.

3.4. Complexity analysis

The computational complexity of RSGT stems from two main components. The Edge Temporal State Modeling has a complexity of $O(|E|)$ for processing $|E|$ edges, including edge type assignment and temporal weight updating. The Recurrent Temporal Node Feature Extraction operates on h-hop neighborhoods N_i^h for each node v_i, involving semantic encoding $O(|N_i^h|d)$, structure encoding $O(|N_i^h| \log |N_i^h| + |N_i^h|d)$, and attention computation $O(|N_i^h|^2 d)$ for d-dimensional features. Considering all nodes and l layers, the complexity for a single time step becomes $O(l \sum_{i=1}^{|V|}(|N_i^h| \log |N_i^h| + |N_i^h|^2 d))$.

In dense graphs or applications requiring global structural awareness, the h-hop neighborhood size $|N_i^h|$ can approach $|V|$, significantly impacting computational efficiency. This challenge can be addressed through neighborhood sampling, where a fixed number $s \ll |V|$ of nodes is selected. Such optimization would reduce the overall complexity to $O(Tl|V|s(s + \log s + d) + T|E|d)$ across T time steps. The practical efficiency can be further enhanced by the sparsity of real-world graphs ($|E| \ll |V|^2$) and the ability to configure the sampling size s based on specific computational constraints while maintaining model effectiveness.

4. Experiments

In order to rigorously evaluate the effectiveness of the proposed RSGT model, we conduct comprehensive experiments across diverse dynamic graph scenarios. These experiments are performed using two NVIDIA GTX 1080Ti GPUs, an Intel(R) Xeon(R) Gold 6130 processor (2.60 GHz), and 192 GB of RAM. The RSGT model's components are implemented using Python 3.7.1 and PyTorch 1.4.0.

4.1. Datasets

To evaluate RSGT's performance across diverse network environments, we selected four representative real-world dynamic graph datasets: twi-Tennis, CollegeMsg, cit-HepTh, and sx-MathOF. These datasets encompass different application domains and exhibit distinct characteristics in terms of temporal dynamics, node and edge densities, and interaction types.

The twi-Tennis dataset [32] captures temporal interactions in social media during tennis tournaments, with nodes representing Twitter accounts and edges denoting mentions. This dataset exemplifies rapidly evolving interaction patterns. CollegeMsg [33] is derived from a university's online social network, where edges represent private messages between users. The cit-HepTh dataset [34] models scientific citation networks, with nodes as papers and edges as citations, capturing the cumulative nature of knowledge diffusion. The sx-MathOF dataset [35] represents interactions on the Math Overflow forum, demonstrating heterogeneous temporal patterns with varying degrees of persistence.

Each dataset is divided into multiple snapshots, capturing the temporal evolution of the networks. Table 1 summarizes the key statistics and train/test split ratios.

4.2. Competing methods

To assess the efficacy of our proposed RSGT, we compare it with thirteen state-of-the-art methods spanning three categories of graph representation learning approaches. In the category of static graph embedding, DeepWalk [11] pioneered the random walk-based approach by treating walk sequences as sentences for skip-gram [36] training, while GraphSAGE [30] introduced an inductive framework for neighborhood feature aggregation. For discrete-time dynamic graphs, EvolveGCN [7] employs RNN to evolve GCN parameters, and ROLAND [8] provides a framework for adapting static GNNs to dynamic scenarios through hierarchical state updates. The continuous-time dynamic graph methods represent more recent advances: CTDNE [9] extends random walks to continuous dynamics, TGAT [37] introduces temporal attention for interaction modeling, CAW [38] leverages causal anonymous walks for temporal pattern extraction, TREND [10] incorporates Hawkes processes [39] for evolutionary modeling, DyGFormer [40] employs transformer architecture with neighbor co-occurrence encoding, R-GSAGE [41] captures embedding trajectories through differential equations, ConTIG [17] integrates recurrent structures with GraphSAGE, and TimeSGN [42] presents a decoupled temporal–structural processing framework for scalable embedding generation.

4.3. Experiment results and analyses

4.3.1. Prediction task and parameter settings

Following the experimental framework outlined in [10], we use dynamic link prediction [43] to evaluate the efficacy of the proposed RSGT model. For rigorous evaluation, we divide the snapshots from various time slices of dynamic graph $\mathcal{G}$ into training and test datasets. The training dataset $\mathcal{D}^{tr}$ encompasses all snapshots prior to time tr, serving as the historical context for model learning. The test dataset $\mathcal{D}^{te}$ consists of snapshots after time tr, used to assess the model's predictive capability. To comprehensively evaluate our model's capability in learning both short-term and long-term historical information, as well as its performance in predicting links over varying time horizons, we employ different train–test split ratios for each dataset, as shown in Table 1. This diverse set of split ratios allows us to simulate various real-world scenarios, from rapidly evolving social networks to more stable citation networks, thereby providing a thorough assessment of RSGT's adaptability and predictive power across different temporal scales.

Based on empirical analysis of information density and computational efficiency, we set the node representation dimension d to 32 for the twi-Tennis, CollegeMsg, and sx-MathOF datasets, and 16 for the cit-HepTh dataset. During the testing phase, we generate temporal node representations using the trained model, which are then input into a logistic regression classifier to predict edge existence probabilities for each snapshot. For each test snapshot in $\mathcal{D}^{te}$, we maintain a balanced positive–negative sample ratio of 1:1, with 80% of edges used for

Table 2
Performance comparisons of dynamic link prediction among competing baselines on twi-Tennis and ColledgeMsg.

	twi-Tennis				CollegeMsg			
	Accuracy ↑	Recall ↑	Precision ↑	F1 ↑	Accuracy ↑	Recall ↑	Precision ↑	F1 ↑
DeepWalk	61.96 ± 1.27	61.05 ± 2.83	61.51 ± 2.62	61.27 ± 2.81	66.54 ± 5.36	67.57 ± 5.81	68.22 ± 5.95	67.86 ± 5.86
GraphSAGE	62.76 ± 1.76	62.02 ± 1.63	62.50 ± 1.78	62.26 ± 1.66	58.91 ± 3.67	60.23 ± 4.15	60.57 ± 4.30	60.45 ± 4.22
EvolveGCN	64.73 ± 0.64	63.51 ± 0.92	64.12 ± 1.01	63.80 ± 0.98	63.27 ± 4.42	65.62 ± 4.64	65.37 ± 4.81	65.44 ± 4.72
ROLAND	68.45 ± 2.08	68.61 ± 1.74	69.03 ± 1.81	68.80 ± 1.74	69.44 ± 2.86	70.19 ± 2.58	70.53 ± 2.63	70.32 ± 2.64
CTDNE	58.14 ± 2.67	58.33 ± 2.04	58.91 ± 2.19	58.61 ± 2.12	62.55 ± 3.67	65.34 ± 2.20	65.82 ± 2.36	65.56 ± 2.34
TGAT	69.01 ± 1.54	69.03 ± 1.41	69.56 ± 1.56	69.24 ± 1.47	70.35 ± 2.54	71.01 ± 2.35	71.43 ± 2.46	71.20 ± 2.37
CAW	71.35 ± 1.68	71.15 ± 1.41	72.03 ± 1.46	71.57 ± 1.42	73.04 ± 2.11	72.71 ± 1.95	73.23 ± 2.01	72.95 ± 1.98
TREND	74.02 ± 1.78	73.83 ± 1.61	75.46 ± 1.63	74.63 ± 1.66	74.55 ± 1.95	74.33 ± 1.91	75.91 ± 2.04	75.64 ± 2.09
DyGFormer	75.89 ± 1.52	75.67 ± 1.43	77.21 ± 1.58	76.43 ± 1.50	76.32 ± 1.78	76.11 ± 1.69	77.54 ± 1.86	76.82 ± 1.77
R-GSAGE	74.85 ± 1.63	74.62 ± 1.55	76.18 ± 1.59	75.39 ± 1.57	75.21 ± 1.83	74.98 ± 1.75	76.43 ± 1.92	75.70 ± 1.83
ConTIG	75.13 ± 1.59	74.91 ± 1.48	76.49 ± 1.55	75.69 ± 1.51	75.78 ± 1.81	75.54 ± 1.72	77.02 ± 1.89	76.27 ± 1.80
TimeSGN	80.45 ± 1.62	80.12 ± 1.58	81.35 ± 1.65	80.73 ± 1.61	81.85 ± 1.42	81.52 ± 1.38	82.45 ± 1.45	81.98 ± 1.41
RSGT	**87.59 ± 0.55**	**87.13 ± 0.52**	**88.04 ± 0.67**	**87.55 ± 0.55**	**86.81 ± 0.14**	**86.36 ± 0.19**	**87.21 ± 0.21**	**86.70 ± 0.13**
Gains	8.87%	8.76%	8.25%	8.44%	6.06%	5.94%	5.77%	5.76%

Table 3
Performance comparisons of dynamic link prediction among competing baselines on cit-HepTh and sx-MathOF.

	cit-HepTh				sx-MathOF			
	Accuracy ↑	Recall ↑	Precision ↑	F1 ↑	Accuracy ↑	Recall ↑	Precision ↑	F1 ↑
DeepWalk	51.55 ± 0.90	49.89 ± 0.92	50.89 ± 0.89	50.39 ± 0.98	66.23 ± 0.89	66.47 ± 0.94	67.81 ± 0.98	67.14 ± 1.12
GraphSAGE	70.72 ± 1.96	70.56 ± 2.12	71.98 ± 2.04	71.27 ± 2.41	65.32 ± 1.55	65.52 ± 1.35	66.84 ± 1.69	66.18 ± 1.21
EvolveGCN	61.57 ± 1.53	61.80 ± 1.52	63.04 ± 1.44	62.42 ± 1.54	68.35 ± 0.68	68.33 ± 0.58	69.71 ± 0.39	69.02 ± 0.35
ROLAND	70.57 ± 1.51	70.61 ± 1.44	72.03 ± 1.28	71.32 ± 1.52	73.35 ± 0.98	72.31 ± 0.82	73.77 ± 0.69	73.04 ± 0.85
CTDNE	49.42 ± 1.86	43.79 ± 1.98	44.67 ± 2.67	44.23 ± 2.92	60.72 ± 0.28	60.30 ± 0.39	61.52 ± 0.19	60.91 ± 0.51
TGAT	71.42 ± 1.28	70.39 ± 1.30	71.81 ± 1.36	71.10 ± 1.34	74.15 ± 1.01	74.00 ± 0.81	75.50 ± 0.83	74.75 ± 0.79
CAW	72.75 ± 1.19	71.78 ± 1.21	73.22 ± 1.28	72.50 ± 1.23	77.95 ± 0.77	76.89 ± 0.76	78.45 ± 0.65	77.67 ± 0.72
TREND	80.37 ± 2.08	80.32 ± 1.97	81.94 ± 1.82	81.13 ± 1.92	79.82 ± 1.56	79.21 ± 1.27	80.81 ± 1.31	80.01 ± 1.34
DyGFormer	81.25 ± 1.89	81.18 ± 1.82	82.73 ± 1.75	81.95 ± 1.78	80.94 ± 1.43	80.36 ± 1.21	81.89 ± 1.28	81.12 ± 1.24
R-GSAGE	81.58 ± 1.85	81.52 ± 1.79	83.08 ± 1.71	82.29 ± 1.75	81.63 ± 1.38	81.02 ± 1.17	82.57 ± 1.23	81.79 ± 1.20
ConTIG	82.19 ± 1.92	82.13 ± 1.85	83.72 ± 1.78	82.92 ± 1.81	81.21 ± 1.41	80.62 ± 1.19	82.16 ± 1.25	81.38 ± 1.22
TimeSGN	83.15 ± 1.65	82.85 ± 1.58	83.95 ± 1.62	83.39 ± 1.60	83.35 ± 1.25	83.05 ± 1.18	84.15 ± 1.32	83.59 ± 1.25
RSGT	**87.20 ± 0.49**	**86.85 ± 0.46**	**87.49 ± 0.32**	**87.17 ± 0.40**	**87.91 ± 0.32**	**87.45 ± 0.38**	**88.35 ± 0.46**	**87.90 ± 0.42**
Gains	4.87%	4.83%	4.22%	4.53%	5.47%	5.30%	4.99%	5.16%

classifier training and 20% for testing. The scale factor α and the power factor β are kept constant at 1 across all datasets. We conduct five independent runs and report the mean and standard deviation of classification accuracy and F1 scores.

4.3.2. Comparison with competing baselines

The experimental results for dynamic link prediction, as presented in Tables 2 and 3, demonstrate that RSGT achieves state-of-the-art performance across all evaluated datasets, with significant improvements in both accuracy (4.22%–8.87%) and F1 scores (4.53%–8.44%). This comprehensive evaluation validates the effectiveness of our proposed temporal–structural modeling framework.

The experimental results reveal three critical aspects in dynamic graph representation learning: temporal pattern characterization, structural information preservation, and temporal–structural feature integration. The substantial performance gap between static methods (DeepWalk, GraphSAGE) and dynamic approaches empirically validates the necessity of explicit temporal modeling, with static methods exhibiting 15%–20% performance degradation across datasets. This limitation is particularly evident in networks with high temporal volatility, where static approaches fail to capture evolving neighborhood structures and dynamic edge formation patterns. Recent dynamic graph learning methods have shown progressive improvements through increasingly sophisticated modeling strategies. Notably, TimeSGN achieves strong baseline performance through its temporal state modeling and message passing mechanism, yet its decoupled temporal–structural processing paradigm inherently constrains the potential for integrated feature learning. In contrast, RSGT's unified modeling framework demonstrates consistent performance advantages (4.22%–8.87% accuracy improvement), empirically validating the effectiveness of joint temporal–structural feature learning.

The performance analysis across diverse graph scenarios provides empirical validation for RSGT's key technical innovations. In highly dynamic social networks (twi-Tennis, CollegeMsg), RSGT's edge temporal state modeling mechanism demonstrates superior effectiveness, achieving 8.87% and 6.06% accuracy improvements respectively over TimeSGN. This advantage empirically validates the effectiveness of our weighted multi-relation framework in capturing both rapid state transitions and evolving structural patterns. For citation networks (cit-HepTh) characterized by stable edges and strong global dependencies, the structure-reinforced attention mechanism proves particularly effective, maintaining a 4.87% accuracy advantage through comprehensive modeling of persistent relationships and global structural features. The heterogeneous temporal patterns in sx-MathOF further validate RSGT's adaptability, with the integrated temporal–structural modeling achieving a 5.47% accuracy improvement through effective handling of diverse temporal dependencies.

These consistent performance improvements can be attributed to RSGT's architectural innovations in addressing fundamental challenges. The weighted multi-relation framework enables fine-grained temporal pattern modeling through explicit state tracking and evolution modeling. The structure-reinforced transformer architecture effectively preserves both local and global structural information while capturing temporal dependencies. This unified approach to temporal–structural feature learning, combined with the adaptive feature aggregation mechanism, enables RSGT to effectively model complex graph dynamics across diverse scenarios.

4.3.3. Computational efficiency analysis

To analyze the computational efficiency of RSGT and align with real-world scenarios where direct learning on large-scale complete

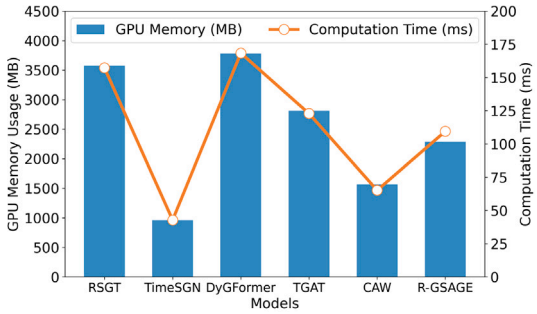


Fig. 3. Computational efficiency comparison across models. The bars (left y-axis) indicate GPU memory consumption in MB, while the line (right y-axis) shows average computation time in milliseconds per batch.

graphs is typically infeasible, we conduct comparative analysis using minibatch training. Fig. 3 illustrates the efficiency comparison between RSGT and five representative baselines, evaluated with batch size 32 and neighborhood hop count 2, averaged across datasets.

Comparative Efficiency Analysis The empirical analysis reveals distinct efficiency patterns corresponding to model architectures. Transformer-based models (RSGT and DyGFormer) demonstrate higher resource requirements in both memory utilization and computation time, primarily due to the quadratic complexity of self-attention mechanisms and the overhead of maintaining structural information in attention matrices. Conversely, TimeSGN exhibits superior efficiency through its decoupled temporal–structural processing paradigm, which circumvents the computational intensity of full self-attention via specialized state updates. TGAT, CAW, and R-GSAGE show intermediate efficiency profiles, reflecting their balanced approaches to temporal–structural modeling.

Effectiveness-Efficiency Trade-off The results reveal an inherent trade-off in model design. RSGT's sophisticated architecture, while computationally intensive, delivers consistently superior prediction accuracy across all datasets. This exemplifies a classic effectiveness-efficiency trade-off, where the enhanced modeling capability comes at the cost of increased computational overhead. For applications where prediction quality is paramount, this computational cost may be justified by the significant performance gains.

4.4. Ablation study

To systematically evaluate the effectiveness and necessity of each architectural component within RSGT, we performed a comprehensive ablation study on dynamic link prediction tasks across four datasets. The ablation experiments were designed to quantitatively assess two fundamental aspects of our model: those related to edge temporal state modeling and those related to graph topology learning.

4.4.1. Edge temporal state modeling

To rigorously examine the impact of our temporal modeling framework, we evaluate the following variants:

- RSGT-T: Retains only edge weights without considering edge types. This variant enables us to quantify the contribution of short-term dynamics captured through edge type transitions.
- RSGT-W: Retains only edge types without explicit edge weights modeling. This configuration allows us to isolate the impact of long-term temporal dependencies modeled through weight evolution.
- RSGT-TW: Uses the original dynamic graph directly without explicit modeling of edge temporal states or weights. This baseline variant helps evaluate the collective impact of our temporal modeling framework.

Figs. 4(a)–4(d) illustrate the performance of these variants across all datasets. The experimental results reveal that RSGT-T and RSGT-W perform similarly on the twi-Tennis, ColledgeMsg, and sx-MathOF datasets, with only minor differences in their performance metrics. This empirical observation suggests the complementary nature of edge types and weights in capturing temporal dynamics. The full RSGT model consistently outperforms these variants across all datasets, validating the synergistic effect of combining both temporal modeling components.

A particularly noteworthy finding emerges from the cit-HepTh dataset, where RSGT and RSGT-T demonstrate clear performance superiority over RSGT-W and RSGT-TW. This distinctive pattern can be attributed to the inherent characteristics of citation networks, where edges exhibit high temporal stability post-formation, emphasizing the crucial role of long-term structural evolution in prediction accuracy.

4.4.2. Graph topology learning

To comprehensively assess the impact of structural information integration, we evaluate the following variants:

- RSGT-S: This variant excludes node-pair topological attributes ($\mathbf{attr}_s^t$) such as degree information and path metrics.
- RSGT-P: This configuration omits path-based temporal information ($\mathbf{ATTR}_p^t$), focusing solely on local structural features.
- RSGT-SP: This variant relies exclusively on the base self-attention mechanism, excluding all explicit structural information integration.

The experimental results, as visualized in Figs. 4(e)–4(h), demonstrate significant performance degradation for both RSGT-S and RSGT-P across all datasets, with RSGT-P exhibiting more severe deterioration. This observation empirically validates that path-based temporal features provide more comprehensive structural dependency information compared to local topological attributes.

As theoretically anticipated, RSGT-SP performed the worst among all ablated models due to its reliance solely on semantic attention, highlighting the limitations of pure transformer architectures in capturing graph-specific dependencies. This further confirms the efficacy of our structure-reinforced attention mechanism that integrates graph topological information into the Transformer architecture.

In conclusion, RSGT outperforms all other ablation variants across all datasets, empirically demonstrating the effectiveness of our dual-aspect modeling approach: explicit temporal state representation through typed and weighted edges, combined with comprehensive structural information integration in the learning process.

4.5. Performance impact of parameters

To systematically optimize the performance of our proposed RSGT model, we conducted a comprehensive analysis of four key hyperparameters: shortest path distance (spd), window size, number of encoding layers (N_{layer}), and number of attention heads (N_{head}). This analysis aims to understand the impact of each parameter on model performance and determine their optimal values for our experiments.

Shortest Path Distance The shortest path distance (spd) parameter, which defines the maximum hop count of neighbors incorporated in node representation learning, was systematically evaluated by varying from 1 to 9 across our experiments. Our analysis reveals a consistent positive correlation between spd and model performance across all datasets, with a particularly noteworthy observation being the sustained performance improvement at higher spd values. This finding contrasts significantly with traditional GNN models, which typically suffer from performance degradation beyond 2–3 hops due to over-smoothing [22]. The improvement is most pronounced in the sx-MathOF dataset, demonstrating our model's capacity to effectively leverage higher-order neighborhood information while preserving node distinctiveness. We observe performance improvements plateau after

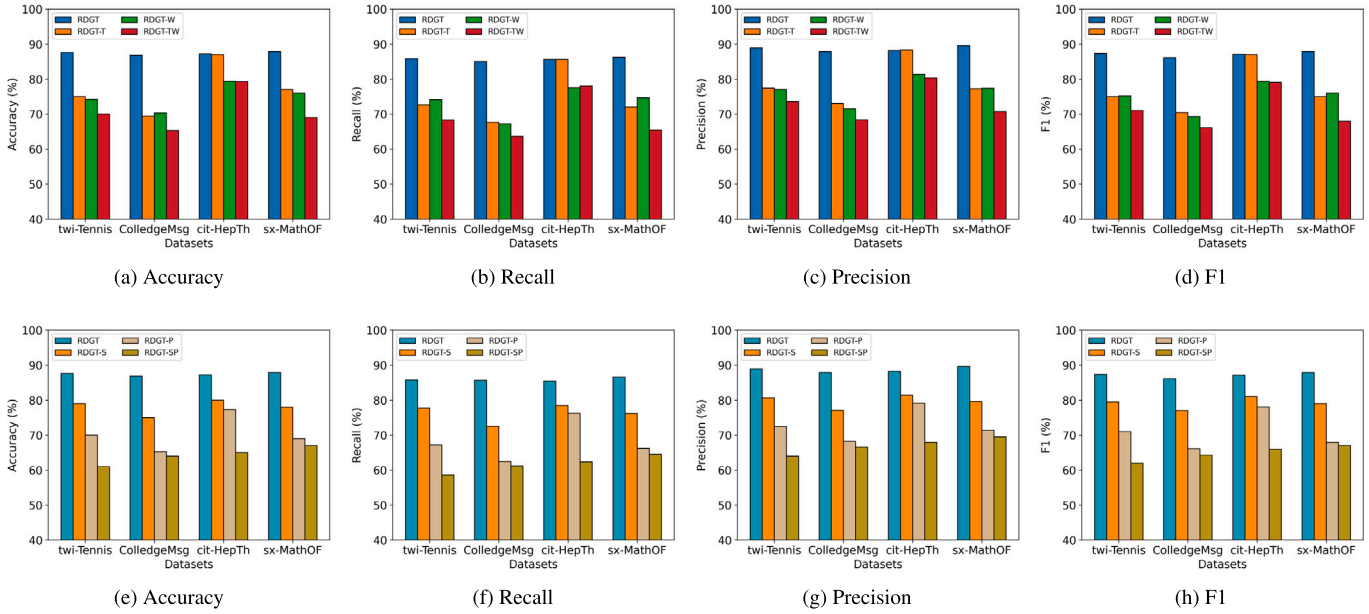


Fig. 4. Performance of different ablated variations of RSGT. (a)–(d) show the impact of edge temporal state modeling, while (e)–(h) illustrate the effect of graph topology learning across different performance metrics and datasets.

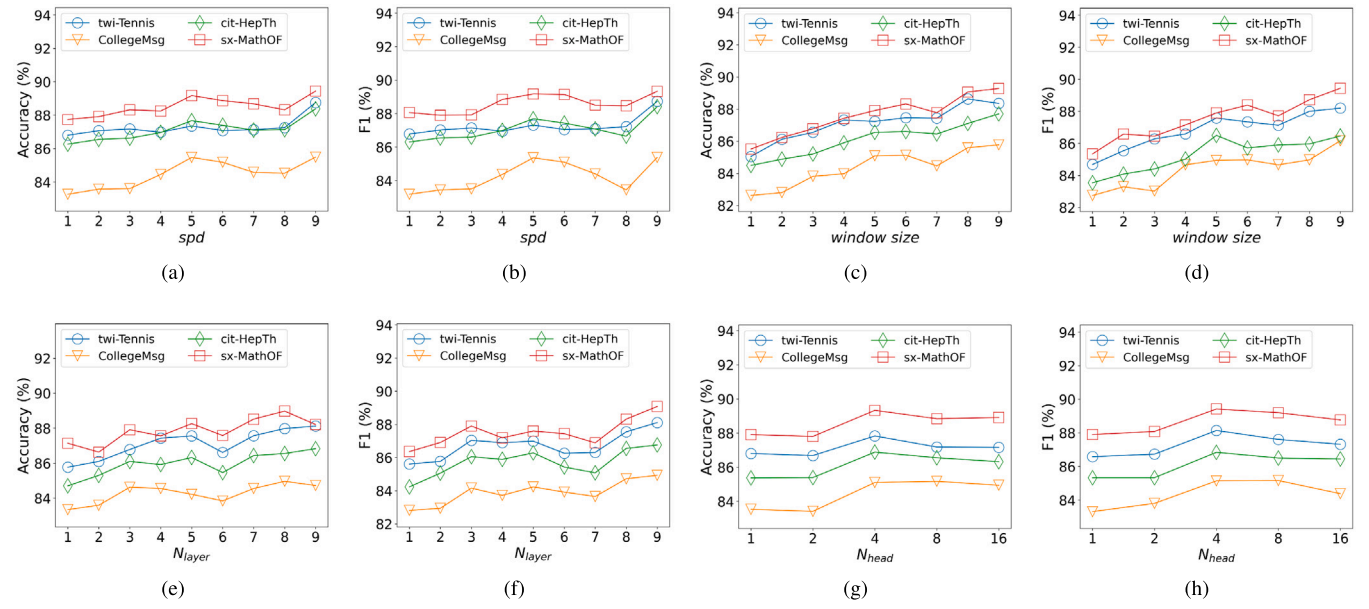


Fig. 5. Impact of key hyperparameters on RSGT performance across four datasets. The subplots show the effect of (a–b) shortest path distance (spd), (c–d) window size, (e–f) number of encoding layers (N_{layer}), and (g–h) number of attention heads (N_{head}) on model performance. The x-axis in each subplot represents the varying values of the respective parameter, and the y-axis shows the corresponding performance metric in percentage.

$spd = 5$ for most datasets, primarily due to the natural decay of information relevance with distance rather than over-smoothing. This successful utilization of higher-order neighborhood information validates the effectiveness of our structure-aware attention mechanism and edge temporal state modeling. Based on these empirical findings and computational considerations, we adopted $spd = 5$ for the twi-Tennis, CollegeMsg, and cit-HepTh datasets, and $spd = 2$ for the sx-MathOF dataset in our comparative experiments.

Window Size The window size parameter defines the number of consecutive snapshots preceding the target time slice used for model training. Through systematic experimentation with window sizes ranging from 1 to 9, we observed that model performance exhibits a positive correlation with window size. Figs. 5(c) and 5(d) demonstrate that model performance generally improves as the window size increases. This trend indicates that our recurrent learning framework effectively utilizes long-term temporal data and autonomously extracts valuable historical information. The performance gains are particularly pronounced for the cit-HepTh dataset, which aligns with the inherent characteristics of citation networks where long-term dependencies play a crucial role in link formation. Conversely, the CollegeMsg dataset exhibits reduced sensitivity to increased window size, reflecting the more ephemeral nature of messaging interactions. To balance performance gains with computational costs, we set the window size to 5 for all datasets in our experiments.

Number of Encoding Layer The N_{layer} parameter determines the depth of the structure-reinforced graph transformer and impacts its ability to learn deep latent features. Our experimental investigation varied N_{layer} from 1 to 9 to comprehensively assess its impact on

model performance. Figs. 5(e) and 5(f) reveal a general trend of improved performance with increased N_{layer}. This empirical observation suggests that deeper architectural configurations enable the capture of more sophisticated temporal–structural patterns in dynamic graphs. However, we observe diminishing returns and even slight performance degradation for some datasets (e.g., cit-HepTh) at higher N_{layer} values. This could be attributed to overfitting or the vanishing gradient problem in very deep networks. Considering the trade-off between model performance and computational complexity, we chose $N_{layer} = 4$ for our experiments, which provides a good balance across all datasets.

Number of Attention Head Multi-head attention allows the model to focus on different representation subspaces simultaneously. Our systematic evaluation encompassed N_{head} values of 1, 2, 4, 8, and 16. As shown in Figs. 5(g) and 5(h), increasing N_{head} generally improves model performance, particularly from 1 to 4 heads. This observation aligns with established findings in attention mechanism research [27], demonstrating the benefit of attending to multiple representation subspaces. A notable observation is that performance tends to plateau or even slightly decrease beyond 4 heads for most datasets. This could be due to increased model complexity leading to overfitting, especially on smaller datasets. Based on these results, we set $N_{head} = 4$ in our experiments, which provides optimal performance across datasets while maintaining computational efficiency.

5. Related work

Dynamic graph representation learning methods aim to learn node representations by incorporating both structural characteristics and temporal dynamics of evolving graphs. These methods can be broadly categorized into two types based on their modeling strategies: snapshot-based discrete dynamic graphs and timestamp-based continuous dynamic graphs. This section reviews related works in these two categories.

5.1. Discrete dynamic graph representation learning

Building on the advancements and state-of-the-art achievements of message-passing-based GNNs in static graph tasks [30], research has progressed toward integrating GNNs with sequence models for discrete dynamic graph representation learning. This integration paradigm leverages GNNs for extracting structural information while employing sequence models to capture evolutionary dynamics.

The established approach follows a two-stage process: GNNs first extract structural features from temporal snapshots, followed by sequence models that capture evolutionary patterns between snapshots. GCRN [13] exemplifies this methodology by combining GCN [30] for node embedding generation with LSTM for temporal pattern learning. VRGNN [14] extends this framework by incorporating Variational Graph Auto-Encoder, strengthening the model's expressive power and uncertainty modeling capabilities.

While these approaches provide effective solutions, their limited inductive learning capability hinders their practical applications, particularly for newly introduced nodes. Subsequent research has addressed this limitation through concurrent learning of structural and dynamic features. EvolveGCN [7] introduces innovative architectures that capture graph sequence dynamics by evolving GCN parameters through RNN, rather than directly updating node embeddings. DyTed [15] advances this direction through a disentangled representation learning framework, decomposing node representations into independent components for precise temporal–structural evolution characterization while maintaining robust inductive capabilities. Recent theoretical analyses [44] have sparked critical discussions on optimizing model complexity in temporal networks, emphasizing efficient architectures that capture essential dependencies without excessive complexity.

Recent innovations have systematically addressed the challenges of feature propagation and over-smoothing while enhancing temporal–structural integration. TR-SAGNN [16] presents a structure-adaptive framework that combines temporal representation learning with residual connections, utilizing temporal attention and adaptive graph structure learning to capture node dynamics while mitigating feature propagation issues. Similarly, TAL-TKGC [45] implements an importance-weighted mechanism that balances temporal dynamics with structural significance through temporal attention and weighted GCN architecture. ROLAND [8] introduces a comprehensive framework that adapts static GNNs to dynamic scenarios by treating multi-layer node embeddings as hierarchical states with recurrent temporal updates.

5.2. Continuous dynamic graph representation learning

The continuous dynamic graph represents edges at various temporal instances as timestamped events, which can be denoted as a set of triplets $\{\langle v_i, v_j, t\rangle\}$, where $\langle v_i, v_j, t\rangle$ signifies the presence of an edge between nodes v_i and v_j at the timestamp t. Various representation learning methods for continuous dynamic graphs [9,10,37,38] aim to understand and elucidate the evolving patterns of these graphs from such temporal event sequences.

Continuous dynamic graph representation learning methods can be broadly categorized into two main approaches: sequence-based modeling methods and advanced temporal modeling methods. Sequence-based modeling methods focus on capturing temporal-topological information by treating the graph evolution as a sequence of events. CTDNE [9] generalizes random walk-based embedding techniques to continuous dynamic graphs, effectively capturing local temporal patterns but potentially struggling with long-range dependencies. Attention-based methods like TGAT [37] and CAW [38] leverage self-attention mechanisms to capture temporal-topological interactions. TGAT introduces a temporal graph attention layer to aggregate features from temporal-topological neighborhoods, while CAW proposes Causal Anonymous Walks as an automatic retrieval mechanism for temporal graphs. These methods excel at capturing complex temporal dependencies but may face computational challenges with very large graphs. DyGFormer [40] employs a Transformer-based architecture with neighbor co-occurrence encoding and patching techniques to learn from nodes' historical first-hop interactions, effectively capturing both short-term and long-term dependencies.

Advanced temporal modeling methods utilize sophisticated techniques to model the continuous evolution of graphs. TREND [10] leverages Hawkes processes to incorporate the evolutionary characteristics of temporal edges into node representations. TimeSGN [42] advances this direction by introducing a divided temporal-message passing paradigm with an innovative state updater mechanism, achieving efficient modeling of node evolution while maintaining scalability. From a geometric perspective, recent work [46] leverages hyperbolic geometry to enhance temporal graph representation learning, offering unique advantages in preserving hierarchical structures while ensuring computational efficiency. R-GSAGE [41] employs neural ordinary differential equations to model the continuous dynamic evolution of node embedding trajectories, allowing for a more nuanced representation of temporal dynamics. GraSSP [47] proposes a novel stochastic process to model link durations and their absences in continuous-time graphs. ConTIG [17] integrates a recurrent structure into GraphSAGE to jointly explore structural and temporal patterns while maintaining a lightweight architecture, making it particularly efficient for large-scale dynamic graphs.

Despite these advancements, existing approaches still face several challenges. They learn evolutionary patterns implicitly, without explicitly considering the various temporal states of edges and their impact on node representations. Furthermore, the prevalent use of GNNs as the backbone for structural feature extraction often leads to over-smoothing issues, limiting model performance. Additionally, most current approaches struggle to effectively balance the modeling of both short-term and long-term temporal dependencies, especially in graphs

with diverse temporal scales. Our proposed RSGT addresses these limitations by introducing a novel edge temporal state modeling technique and a structure-reinforced graph transformer. It allows for explicit modeling of edge dynamics while effectively capturing both local and global structural information, thereby overcoming the challenges faced by existing methods.

6. Conclusion and future work

This paper introduces a novel recurrent framework for dynamic graph representation learning, termed the Recurrent Structure-reinforced Graph Transformer (RSGT). To better understand the temporal dynamics of graphs, we propose capturing the influence of dynamic interactions between nodes on the strength of their relationships. Thus, we explicitly model edge temporal states by converting each original snapshot into a weighted multi-relation graph based on differences with previous snapshots. Subsequently, we design a Structure-reinforced Graph Transformer (SGT) to recurrently learn and update node temporal representations across the dynamically modeled weighted multi-relation graphs. The SGT effectively encapsulates global node semantic correlations, graph topology dependencies, and edge temporal state features concurrently, stemming from a structure-aware attention-reinforced mechanism that modifies the original node-wise attention score with pairwise topological structure and the corresponding shortest path. This leads to the generation of high-quality node representations that encode both global semantic relevance and structural information. The effectiveness of RSGT is validated through extensive experiments, demonstrating superior performance in understanding the evolutionary characteristics of dynamic graphs compared to existing baseline methods.

In future work, although RSGT demonstrates a well-balanced effectiveness-efficiency trade-off, we aim to further enhance its scalability for large-scale graph applications. For extremely large graphs, two complementary optimization strategies warrant investigation. First, distributed parallel training across multiple computing nodes could leverage modern computational infrastructure to accelerate model convergence. Second, adaptive neighborhood sampling techniques could enable efficient subgraph extraction, significantly reducing computational requirements while preserving representation quality. These approaches would enable RSGT to maintain its superior predictive performance while efficiently processing large-scale dynamic graph data, making it suitable for real-world enterprise deployment scenarios. Beyond scalability, we aim to extend RSGT to heterogeneous dynamic graphs, which comprise multiple node and edge types with diverse attributes. This extension would enable more nuanced modeling of complex systems in domains such as social networks, e-commerce, and biomedical research. Such adaptations would demonstrate the versatility of our approach while potentially advancing dynamic graph analysis in novel directions.

CRediT authorship contribution statement

Shengxiang Hu: Writing – original draft, Resources, Investigation, Formal analysis, Conceptualization. **Guobing Zou:** Writing – review & editing, Supervision, Funding acquisition. **Song Yang:** Visualization, Validation, Data curation. **Shiyi Lin:** Methodology, Data curation. **Yanglan Gan:** Writing – review & editing, Methodology, Formal analysis. **Bofeng Zhang:** Writing – review & editing, Project administration.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgments

This work was supported by National Natural Science Foundation of China (No. 62272290, 62172088), and Shanghai Natural Science Foundation, China (No. 21ZR1400400).

Data availability

Data will be made available on request.

References

[1] Y. Zhao, X. Luo, W. Ju, C. Chen, X.-S. Hua, M. Zhang, Dynamic hypergraph structure learning for traffic flow forecasting, in: International Conference on Data Engineering, ICDE, 2023, pp. 2303–2316.
[2] W. Du, S. Chen, Z. Li, X. Cao, Y. Lv, A spatial-temporal approach for multi-airport traffic flow prediction through causality graphs, IEEE Trans. Intell. Transp. Syst. 25 (1) (2024) 532–544.
[3] Z. Yi, I. Ounis, C. MacDonald, Contrastive graph prompt-tuning for cross-domain recommendation, ACM Trans. Inf. Syst. 42 (2) (2024) 60:1–60:28.
[4] J. Yu, X. Xia, T. Chen, L. Cui, N.Q.V. Hung, H. Yin, XSimGCL: Towards extremely simple graph contrastive learning for recommendation, IEEE Trans. Knowl. Data Eng. 36 (2) (2024) 913–926.
[5] C. Liu, W. Wu, S. Wu, L. Yuan, R. Ding, F. Zhou, Q. Wu, Social-Enhanced explainable recommendation with knowledge graph, IEEE Trans. Knowl. Data Eng. 36 (2) (2024) 840–853.
[6] H. Tian, X. Zhang, X. Zheng, D.D. Zeng, Learning dynamic dependencies with graph evolution recurrent unit for stock predictions, IEEE Trans. Syst. Man Cybern.: Syst. 53 (11) (2023) 6705–6717.
[7] A. Pareja, G. Domeniconi, J. Chen, T. Ma, T. Suzumura, H. Kanezashi, T. Kaler, T. Schardl, C. Leiserson, EvolveGCN: Evolving graph convolutional networks for dynamic graphs, in: AAAI Conference on Artificial Intelligence, AAAI, 2020, pp. 5363–5370.
[8] J. You, T. Du, J. Leskovec, ROLAND: Graph learning framework for dynamic graphs, in: ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, SIGKDD, 2022, pp. 2358–2366.
[9] G.H. Nguyen, J.B. Lee, R.A. Rossi, N.K. Ahmed, E. Koh, S. Kim, Continuous-time dynamic network embeddings, in: The Web Conference, WWW, 2018, pp. 969–976.
[10] Z. Wen, Y. Fang, TREND: TempoRal event and node dynamics for graph representation learning, in: ACM Web Conference, WWW, 2022, pp. 1159–1169.
[11] B. Perozzi, R. Al-Rfou, S. Skiena, DeepWalk: Online learning of social representations, in: ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, SIGKDD, 2014, pp. 701–710.
[12] A. Grover, J. Leskovec, Node2Vec: Scalable feature learning for networks, in: ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, SIGKDD, 2016, pp. 855–864.
[13] Y. Seo, M. Defferrard, P. Vandergheynst, X. Bresson, Structured sequence modeling with graph convolutional recurrent networks, in: International Conference on Neural Information Processing, ICONIP, 2018, pp. 362–373.
[14] E. Hajiramezanali, A. Hasanzadeh, K. Narayanan, N. Duffield, M. Zhou, X. Qian, Variational graph recurrent neural networks, in: Advances in Neural Information Processing Systems (NeurIPS), 2019, pp. 10700–10710.
[15] K. Zhang, Q. Cao, G. Fang, B. Xu, H. Zou, H. Shen, X. Cheng, DyTed: Disentangled representation learning for discrete-time dynamic graph, in: ACM SIGKDD Conference on Knowledge Discovery and Data Mining, SIGKDD, 2023, pp. 3309–3320.
[16] X. Bi, Q. Jiang, Z. Liu, X. Yao, H. Nie, G.Y. Yuan, X. Zhao, Y. Sun, Structure-adaptive graph neural network with temporal representation and residual connections, World Wide Web 26 (5) (2023) 3389–3408.
[17] Z. Wang, P. Yang, X. Fan, X. Yan, Z. Wu, S. Pan, L. Chen, Y. Zang, C. Wang, R. Yu, Contig: Continuous representation learning on temporal interaction graphs, Neural Netw. 172 (2024) 106151.
[18] H. Zhu, X. Yang, J. Wei, Path prediction of information diffusion based on a Topic-Oriented relationship strength network, Inform. Sci. 631 (2023) 108–119.
[19] Y. Zhou, G. Yang, B. Yan, Y. Cai, Z. Zhu, Point-of-Interest recommendation model considering strength of user relationship for location-based social networks, Expert Syst. Appl. 199 (2022) 117147.
[20] C. Ying, T. Cai, S. Luo, S. Zheng, G. Ke, D. He, Y. Shen, T.-Y. Liu, Do transformers really perform badly for graph representation? in: Advances in Neural Information Processing Systems (NeurIPS), Vol. 34, 2021, pp. 28877–28888.
[21] D. Chen, L. O'Bray, K. Borgwardt, Structure-Aware transformer for graph representation learning, in: International Conference on Machine Learning, ICML, 2022, pp. 3469–3489.
[22] D. Chen, Y. Lin, W. Li, P. Li, J. Zhou, X. Sun, Measuring and relieving the over-smoothing problem for graph neural networks from the topological view, in: AAAI Conference on Artificial Intelligence, AAAI, 2020, pp. 3438–3445.

[23] T.K. Rusch, M.M. Bronstein, S. Mishra, A survey on oversmoothing in graph neural networks, 2023, arXiv preprint arXiv:2303.10993.
[24] K.G. Quach, P. Nguyen, H. Le, T.-D. Truong, C.N. Duong, M.-T. Tran, K. Luu, DyGLIP: A dynamic graph model with link prediction for accurate multi-camera multiple object tracking, in: IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR, 2021, pp. 13784–13793.
[25] D. Xu, W. Cheng, D. Luo, X. Liu, X. Zhang, Spatio-temporal attentive RNN for node classification in temporal attributed graphs, in: International Joint Conference on Artificial Intelligence, IJCAI, 2019, pp. 3947–3953.
[26] X. Wang, X. He, M. Wang, F. Feng, T.-S. Chua, Neural graph collaborative filtering, in: International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR, 2019, pp. 165–174.
[27] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, A.N. Gomez, Ł. Kaiser, I. Polosukhin, Attention is all you need, in: Advances in Neural Information Processing Systems (NeurIPS), 2017, pp. 5998–6008.
[28] S.P. Borgatti, Centrality and network flow, Soc. Netw. 27 (1) (2005) 55–71.
[29] M. Newman, Networks, 2018.
[30] W. Hamilton, Z. Ying, J. Leskovec, Inductive representation learning on large graphs, in: Advances in Neural Information Processing Systems (NeurIPS), 2017, pp. 1024–1034.
[31] I. Loshchilov, F. Hutter, Decoupled weight decay regularization, in: International Conference on Learning Representations, ICLR, 2019.
[32] F. Béres, R. Pálovics, A. Oláh, A.A. Benczúr, Temporal walk based centrality metric for graph streams, Appl. Netw. Sci. 3 (1) (2018) 1–26.
[33] P. Panzarasa, T. Opsahl, K.M. Carley, Patterns and dynamics of users' behavior and interaction: Network analysis of an online community, J. Am. Soc. Inf. Sci. Technol. 60 (5) (2009) 911–932.
[34] J. Leskovec, J. Kleinberg, C. Faloutsos, Graphs over time: Densification laws, shrinking diameters and possible explanations, in: ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, SIGKDD, 2005, pp. 177–187.
[35] A. Paranjape, A.R. Benson, J. Leskovec, Motifs in temporal networks, in: International Conference on Web Search and Data Mining, WSDM, 2017, pp. 601–610.
[36] T. Mikolov, I. Sutskever, K. Chen, G.S. Corrado, J. Dean, Distributed representations of words and phrases and their compositionality, in: Advances in Neural Information Processing Systems (NeurIPS), 2013, pp. 3111–3119.
[37] D. Xu, C. Ruan, E. Körpeoglu, S. Kumar, K. Achan, Inductive representation learning on temporal graphs, in: International Conference on Learning Representations, ICLR, 2020.
[38] Y. Wang, Y.-Y. Chang, Y. Liu, J. Leskovec, P. Li, Inductive representation learning in temporal networks via causal anonymous walks, in: International Conference on Learning Representations, ICLR, 2021.
[39] A.G. Hawkes, Spectra of some self-exciting and mutually exciting point processes, Biometrika 58 (1) (1971) 83–90.
[40] L. Yu, L. Sun, B. Du, W. Lv, Towards better dynamic graph learning: New architecture and unified library, in: Advances in Neural Information Processing Systems (NeurIPS), 2023, pp. 67686–67700.
[41] H.-Y. Yao, C.-Y. Zhang, Z.-L. Yao, C.P. Chen, J. Hu, A recurrent graph neural network for inductive representation learning on dynamic graphs, Pattern Recognit. 154 (2024) 110577.
[42] Y. Xu, W. Zhang, Y. Zhang, M. Orlowska, X. Lin, TimeSGN: Scalable and effective temporal graph neural network, in: IEEE International Conference on Data Engineering, ICDE, 2024, pp. 3297–3310.
[43] P. Sarkar, D. Chakrabarti, M.I. Jordan, Nonparametric link prediction in dynamic networks, in: International Conference on Machine Learning, ICML, 2012.
[44] W. Cong, S. Zhang, J. Kang, B. Yuan, H. Wu, X. Zhou, H. Tong, M. Mahdavi, Do we really need complicated model architectures for temporal networks? in: International Conference on Learning Representations, ICLR, 2023.
[45] H. Nie, X. Zhao, X. Yao, Q. Jiang, X. Bi, Y. Ma, Y. Sun, Temporal-structural importance weighted graph convolutional network for temporal knowledge graph completion, Future Gener. Comput. Syst. 143 (2023) 30–39.
[46] Y. Xu, W. Zhang, X. Xu, B. Li, Y. Zhang, Scalable and effective temporal graph representation learning with hyperbolic geometry, IEEE Trans. Neural Netw. Learn. Syst. (TNNLS) (2024) 1–15.
[47] A. Celikkanat, N. Nakis, M. Mørup, Continuous-time graph representation with sequential survival process, in: AAAI Conference on Artificial Intelligence, AAAI, 2024, pp. 11177–11185.