Capturing Temporal Node Evolution via Self-supervised Learning: A New Perspective on Dynamic Graph Learning

Lingwen Liu*
2201839@stu.neu.edu.cn
School of Computer Science and Engineering, Northeastern University
Shenyang, Liaoning, China

Guangqi Wen*
2110658@stu.neu.edu.cn
School of Computer Science and Engineering, Northeastern University
Shenyang, Liaoning, China

Peng Cao†
caopeng@cse.neu.edu.cn
School of Computer Science and Engineering, Northeastern University
Shenyang, Liaoning, China

Jinzhu Yang
yangjinzhu@cse.neu.edu.cn
School of Computer Science and Engineering, Northeastern University
Shenyang, Liaoning, China

Weiping Li
wpli@ss.pku.edu.cn
School of Software and Microelectronics, Peking University
Beijing, China

Osmar R. Zaiane
zaiane@cs.ualberta.ca
Alberta Machine Intelligence Institute, University of Alberta
Edmonton, Alberta, Canada

ABSTRACT

Dynamic graphs play an important role in many fields like social relationship analysis, recommender systems and medical science, as graphs evolve over time. It is fundamental to capture the evolution patterns for dynamic graphs. Existing works mostly focus on constraining the temporal smoothness between neighbor snapshots, however, fail to capture sharp shifts, which can be beneficial for graph dynamics embedding. To solve it, we assume the evolution of dynamic graph nodes can be split into temporal shift embedding and temporal consistency embedding. Thus, we propose the Self-supervised Temporal-aware Dynamic Graph representation Learning framework (STDGL) for disentangling the temporal shift embedding from temporal consistency embedding via a well-designed auxiliary task from the perspectives of both node local and global connectivity modeling in a self-supervised manner, further enhancing the learning of interpretable graph representations and improving the performance of various downstream tasks. Extensive experiments on link prediction, edge classification and node classification tasks demonstrate STDGL successfully learns the disentangled temporal shift and consistency representations. Furthermore, the results indicate significant improvements in our STDGL over the state-of-the-art methods, and appealing interpretability and transferability owing to the disentangled node representations.

CCS CONCEPTS

• **Theory of computation → Dynamic graph algorithms**; **Machine learning theory**; • **Information systems → Data mining**.

*Both authors contributed equally to this research.
†Corresponding author.

WSDM '24, March 4–8, 2024, Merida, Mexico

ACM ISBN 979-8-4007-0371-3/24/03
https://doi.org/10.1145/3616855.3635765

KEYWORDS

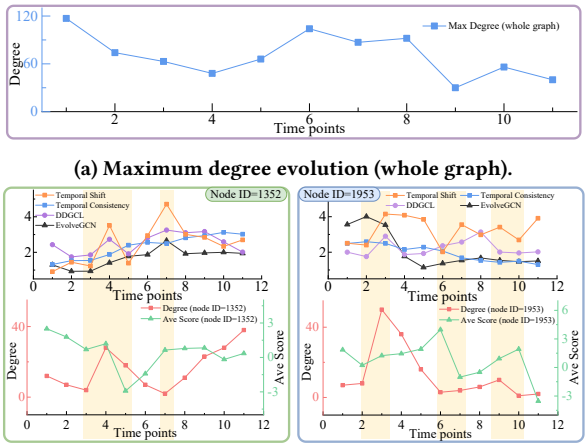
Dynamic Graph; Self-supervised Learning; Link Prediction; Node Classification; Edge Classification

ACM Reference Format:
Lingwen Liu, Guangqi Wen, Peng Cao, Jinzhu Yang, Weiping Li, and Osmar R. Zaiane. 2024. Capturing Temporal Node Evolution via Self-supervised Learning: A New Perspective on Dynamic Graph Learning. In *Proceedings of the 17th ACM International Conference on Web Search and Data Mining (WSDM '24), March 4–8, 2024, Merida, Mexico.* ACM, New York, NY, USA, 9 pages. https://doi.org/10.1145/3616855.3635765

1 INTRODUCTION

Graph representation learning has received much attention due to its wide range of application domains, *e.g.*, social network analysis, protein analysis and recommendation systems. Most of the graph representation learning algorithms focus on the static graphs containing a fixed set of nodes and edges by learning low-dimensional node embeddings for various graph analysis, *e.g.*, link prediction [2], node classification [16], and edge classification [1]. However, many real-world graphs, *e.g.*, recommendation graphs [5, 10], paper citation graphs [19, 23], or dynamic brain graphs [18, 20], are dynamic where graph structures constantly evolve over time, and the simple application of representation learning for the static graph fails to capture the temporal information.

Although various designs are proposed to improve the representation performance of the dynamic graph [8, 17], these methods generally assume that temporal evolution between adjacent time steps is smooth, and attempt to enforce the temporal smoothness of node representations from adjacent snapshots [9, 21]. However, such methods are effective only in high-sparsity settings, failing when nodes exhibit complex temporal characteristics in dynamic graphs [26], as shown in Fig. 1(a). Learning the temporal representations reflecting the inherent evolution patterns in dynamic graphs is a crucial task. It can be observed that temporal evolution is not always smooth and exhibits a more complex evolution, *e.g.*, the maximum node degree varies across time points in the bitcoin dataset. The reason is that economic fluctuations in the Bitcoin market are prone to affect Bitcoin transactions, resulting in sudden changes or irregular patterns in the user transactions or the trust



(a) Maximum degree evolution (whole graph).

(b) Node ID=1352 **(c) Node ID=1953**

Figure 1: The temporal variation on the Bitcoin-OTC dataset. (a): The variation *wrt.* maximum degree for the whole graph. The top of (b) and (c) show the temporal graph evolutions with the disentangled embeddings (PCA) of STDGL and the comparable embeddings (PCA) learned by EvolveGCN [25] and DDGCL [28]. The bottom of (b) and (c) show the temporal graph evolutions *wrt.* degree and the average who-trusts-whom scores.

Table 1: The comparison of EvolveGCN, DDGCL, and our STDGL for the various tasks on the Bitcoin-OTC dataset.

Model	Link prediction		Edge classification
	MRR	AUC	F1
EvolveGCN [25]	14.2±1.4	65.1±0.8	86.2±0.9
DDGCL [28]	18.8±0.9	71.3±0.8	88.1±1.2
STDGL-*c* (Temporal Consistency)	24.5±0.8	81.1±1.6	88.9±0.4
STDGL-*s* (Temporal Shift)	28.7±0.6	85.4±0.8	89.2±0.7

scores [14]. However, numerous works [8, 9] ignore the consideration of the complicated temporal characteristics in dynamic graphs. For example, Jiang [8] proposed a temporal-aware node embedding approach with a joint temporal-aware inference model using temporal consistency information as constraints. Liao [17] and Mahdavi [21] assume that a dynamic network should vary smoothly and continuously between two neighbor snapshots over time, and learn the latent representations by minimizing the difference between two embedding distributions of the neighbor graphs.

To address the limitations of existing works, we consider designing a model that explicitly explores the characteristics of the temporal shift to better model the complex and nonlinearly evolving processes in the dynamic graphs. To this end, we propose a Self-supervised Temporal-aware Dynamic Graph representation Learning framework (STDGL). The framework aims to explicitly disentangle two kinds of temporal information for dynamic node embedding learning by exploring graph temporal and structural properties, *i.e.*, temporal shift information (the sharp evolution mode introduced by sudden events) and temporal consistency information (the stable mode across the time points), thus enhancing interpretable node temporal representation and improving the downstream task performance. To the best of our knowledge, this is the first work to explicitly capture the temporal shift embedding for the dynamic graph analysis.

More specifically, our model consists of two individual encoders to disentangle temporal shift embedding and temporal consistency embedding, and a cross decoder to reconstruct the graph structure. The two individual encoders consist of a local connectivity encoder which leverages the local connectivity to obtain node embedding, and an edge-level encoder which aggregates global connectivity to obtain topology embedding. To better encourage disentangling, we design three losses, *i.e.*, the temporal disentangling loss forcing two neighbor snapshots' temporal shift embeddings dissimilar and temporal consistency embeddings similar, the reconstruction loss and discrepancy loss encourage the consistency between the two graphs reconstructed by combining temporal shift embedding at current snapshot with temporal consistency embeddings at current/neighbor snapshots. Finally, the disentangled temporal embeddings are utilized for the downstream tasks. To investigate the relationship between the disentangled temporal shift embedding and the temporal evolution patterns, we conduct a pilot experiment to investigate it in Fig. 1(b)(c). It can be observed that the disentangled temporal shift embedding is consistent with the significant evolution patterns highlighted by the yellow bands, while other methods fail to capture these trends sufficiently. Contrary to temporal shift embedding, the disentangled temporal consistency embedding exhibits stability across consecutive time-steps. In Table 1, the significant improvement achieved by STDGL-*s* also gives evidence that leveraging temporal shift embedding provides more meaningful patterns for graph-related tasks.

Comprehensive experiments on six real-world datasets across three different tasks, including link prediction, edge classification and node classification, demonstrate the significant improvement of our framework over the existing alternatives. Specifically, STDGL-*s* outperforms DDGCL [28], which is also a self-supervised dynamic graph learning method, with an average improvement of 5.2% and 3.4% on MRR and AUC in the link prediction task on five datasets, outperforms EvolveGCN [25] with an average improvement of 5.4%/5.5% on F1 in the edge/node classification task. The various experimental results prove our assumption that the temporal nodes can be split into temporal shift information and temporal consistent information by a well-designed pretext task from the data itself. Moreover, the comprehensive evaluations demonstrate that disentangled temporal shift information is critical to capture the evolution nature of dynamic graphs.

Our contributions are as follows:

1. This work formalizes the task of learning latent node representations for various temporal graph analysis from the perspective of disentangling the underlying factors (temporal shift and temporal consistency embeddings) hidden in the observable temporal graphs through a self-supervised learning paradigm. Owing to the disentangled temporal node representations, the proposed model has appealing interpretability and transferability. Such a model helps reveal more insights into the underlying dynamic graph evolution.

2. Unlike other self-supervised dynamic graph learning methods that only focus on node embedding learning, we propose a series of encoders and multiple regularizations to capture the structural

representation of the dynamic graphs from the aspects of local connectivity information and global connectivity information.

3. Extensive experiment results on link prediction, edge classification and node classification tasks demonstrate that the proposed disentangled representation learning scheme via pre-training significantly facilitates the learning of the node embedding suited to different downstream tasks.

2 RELATED WORK

2.1 Dynamic Graph Embedding Learning

The dynamic graph representation learning is a very important research topic in the graph domain. TGCN [33] incorporates a Graph Neural Network (GNN) into the Gated Recurrent Unit (GRU) cell by replacing linear transformations in GRU with graph convolution operators to capture the spatio-temporal correlation representations. EvolveGCN [25] utilizes an RNN to dynamically update weights of internal GNNs, which allows the GNN model to change during the test time. DySAT [26] captures node embedding through joint self-attention along the two dimensions of the structural neighborhood and temporal dynamics. However, they cannot explicitly model the temporally-aware information, leads to poor performance.

Recently, many works [8, 9, 22] focus on the temporal-aware graph embedding, which tries to learn the evolving patterns of a graph and incorporate time information into embedding learning. For instance, t-TransE [9] provides a link prediction method by using temporal order constraints to model transformation between time-sensitive relations. In the embedding process, t-TransE enforces the embeddings to be temporally consistent. To incorporate the valid time of facts, [8] proposes a temporal-aware graph embedding approach with a joint temporal-aware inference model using temporal consistency information as constraints. However, a purely supervised learning scheme for the dynamic graph data usually leads to poor generalization due to insufficient supervision.

2.2 Self-supervised Dynamic Graph Learning

Recently, Tian et al. [28] first introduced self-supervised learning into the dynamic graph and proposed a dynamic graph contrastive learning method. They adopted a continuous-time formulation and introduced a time-dependent generalization of the representation similarity metric that measures the agreement between two temporal views of the same node. Besides, Chen et al. [3] proposed a pre-training framework on dynamic graph neural networks based on the graph generation tasks, which can capture structural features and node attributes by fusing temporal information. However, these self-supervised dynamic graph approaches extend the self-supervised paradigm of static graphs and fail to sufficiently consider the rich temporal properties in the dynamic graphs. Moreover, these works cannot explain how the self-supervised paradigm assists dynamic graph learning.

3 METHODOLOGY

3.1 Notation and Problem Statement

A dynamic graph $\mathcal{G} = \{G^1, G^2, \cdots, G^T\}$ can be modeled as a set of multiple graphs G^t at different time points t, where $G^t = (\mathcal{V}^t, \mathcal{E}^t)$. $\mathcal{V}^t = \{v_1, v_2, \cdots, v_{\mathcal{N}^t}\}$ represents the node set of G^t and $\mathcal{N}^t$ denotes the node number at time point t. Let $\mathcal{E}^t = \{e_{ij} | v_i, v_j \in \mathcal{V}^t\}$ denotes the edge set of G^t, where e_{ij} connects two nodes v_i and v_j. Let $A^t \in \mathbb{R}^{\mathcal{N}^t \times \mathcal{N}^t}$ and $X^t \in \mathbb{R}^{\mathcal{N}^t \times D}$ denote the adjacency matrix and the node features at t, $\mathcal{A} = \{A^1, A^2, \cdots, A^T\}$ and $\mathcal{X} = \{X^1, X^2, \cdots, X^T\}$ denote the adjacency matrices and node features of $\mathcal{G}$ across times. In dynamic graphs, the evolution nature is reflected by the changes from the aspects of $\mathcal{V}$, $\mathcal{E}$, $\mathcal{A}$ and $\mathcal{X}$ across time.

3.2 Method

3.2.1 Overview. The overview of our proposed method is illustrated in Fig. 2. It consists of two main components, *i.e.*, a couple of temporal-aware encoders for yielding temporal shift embedding and temporal consistency embedding, and a cross decoder for the reconstruction of the graph structure. Specifically, the temporal-aware encoder contains the local connectivity encoder E_l composed of a stack of dynamic graph convolution layers for node embedding learning and a global connectivity encoder E_g for learning the global connectivity of nodes. The cross decoder takes the combination of temporal shift embedding and temporal consistency embedding as inputs and reconstructs the graph structures. To encourage the disentangling of temporal shift embeddings and temporal consistency embeddings, we design the overall losses including the temporal disentangling loss $\mathcal{L}_t$ forcing the temporal shift embeddings of neighboring snapshot dissimilar and temporal consistency embeddings similar, while the reconstruction loss $\mathcal{L}_{rec}$ and discrepancy loss $\mathcal{L}_d$ constraining the consistency between the two graphs reconstructed by combining temporal shift embedding at current snapshot with temporal consistency embeddings at current/neighbor snapshots. The disentangled embeddings are used for different downstream tasks.

3.2.2 Temporal-aware encoder. We design a temporal-aware encoder to extract the temporal shift information and temporal consistency information. It should be noted that the weights of encoders for temporal shift/consistency information are independent of each other. Due to the evolution of dynamic graphs not only in terms of node attributes and the addition/absence of nodes but also in terms of topological dynamics of graph structure, in order to achieve better graph disentanglement representation, it is essential to fully capture the node embeddings along two dimensions: local connectivity and global connectivity. To this end, we propose the local connectivity encoder E_l and global connectivity encoder E_g.

(1) Local connectivity encoder The traditional graph convolution layer on graph $G = (A, X)$ can be formulated as: $H = \sigma(\widetilde{A}XW)$, where $\widetilde{A}$ is the normalized adjacency matrix of the input graph, W is the learnable parameter, σ denotes the activation function *ReLU*, and $H \in \mathbb{R}^{N \times D}$ is the node embedding of graph G. However, such a simple graph convolution layer cannot describe the complex intra-node and inter-node evolution relationship between two graphs at consecutive time points. To incorporate the embeddings of the previous time point in the graph convolution layer for graphs G^t, we propose a dynamic graph convolution layer.

Let X^t be the node embedding matrix for graphs G^t. We assume that the node embeddings H_l^{t-1} at $t-1$ have been updated with a previous graph convolution layer. The extended graph convolution formulation can be defined as follows: $H_l^t = E_l(G^t, G^{t-1}) =$

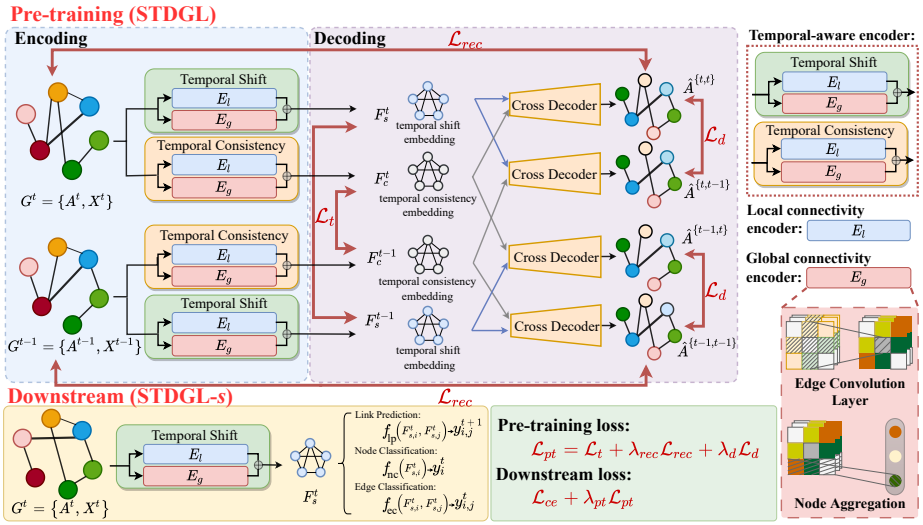


Figure 2: Illustration of the proposed STDGL. In the pre-training stage, our model includes an encoder-decoder architecture to exploit the dynamic graphs with the self-supervised learning scheme. We aim to disentangle the representations into temporal shift embedding $F_s^{\cdot}$ and temporal consistency embedding $F_c^{\cdot}$ by local connectivity and global connectivity encoders. Then, we design a cross decoder for reconstructing the graph. Moreover, the temporal disentangling loss $\mathcal{L}_t$, the topology reconstruction loss $\mathcal{L}_{rec}$ and the discrepancy loss $\mathcal{L}_d$ are introduced to better encourage the discrimination of temporal shift and temporal consistency embeddings. In the downstream stage, only the temporal shift embedding (named STDGL-*s*) is used for graph-related tasks which are supervised by the cross-entropy loss. We preserve the three losses used during the pre-training stage by a ratio of 0.5.

$\sigma((\sigma(\widetilde{A}^t X^t W_1); H_l^{t-1})W_2)$ where W_1 and W_2 are trainable parameters. The output node embeddings H_l^t capture information from both the current and previous time points, allowing the model to learn temporal dependencies in dynamic graphs.

(2) Global connectivity encoder Most recent works on dynamic graph representation learning only aggregate information via a node's local connections to generate the node embedding [32]. However, these works cannot capture the global connectivity patterns of the node. To fully consider the node topology information in dynamic graphs, we propose a global connectivity encoder to explore the global connectivity information and then aggregate the information into node embeddings. Specifically, we introduce a specialized edge convolution operation consisting of vertical and horizontal convolution kernels, which is designed to explore the potential graph structure information by considering the spatial topological association in the graph structure. To overcome the expensive computation cost on large-scale graph adjacency matrix A^t of time point t, we propose to divide the graph into s sub-graphs $\{A^t_{\{1\}}, A^t_{\{2\}}, ..., A^t_{\{s\}} | A^t_{\{l\}} \in \mathbb{R}^{\mathcal{M}\times\mathcal{M}}\}$, each with $\mathcal{M}$ nodes [4]. At first, we introduce a learnable semantic augmentation matrix $S \in [0,1]^{\mathcal{M}\times\mathcal{M}}$, to indicate the critical structure of the graph. The enhanced graph structure can capture and highlight the explicit interaction relation and the implicit semantic relation. Specifically, we obtain the augmented graph structure $\widetilde{A}^t_{\{k\}}$ via $\widetilde{A}^t_{\{k\}} = A^t_{\{k\}} \odot S$.

The global connectivity encoder E_g consists of an edge convolution layer and a node aggregation operation. To capture the essential topological representation, we introduce an edge convolution layer to globally leverage the connectivity by aggregating the features of the all edges associated with the nodes at the two ends of a connection. Our edge convolution layer involves multiple cross-shaped kernels for edge embedding learning, which can be formulated as:

$$F^t = \text{Concat}\Big(\sum_{i=0}^{\mathcal{M}} w_i^v \cdot \widetilde{A}^t_{\{k\}i\cdot} + \sum_{j=0}^{\mathcal{M}} w_j^h \cdot \widetilde{A}^t_{\{k\}\cdot j}\Big)_{k=1}^{s} \tag{1}$$

where $w_i^v \in \mathbb{R}^{1\times\mathcal{M}}$ and $w_j^h \in \mathbb{R}^{\mathcal{M}\times 1}$ are learnable parameters of vertical and horizontal convolution kernels, $F^t \in \mathbb{R}^{\mathcal{N}_t\times\mathcal{N}_t}$ is the topology embedding of $\widetilde{A}^t$. The edge convolution is capable of enhancing the potentially important connections and generating enhanced edge embedding. Then the node embedding H_g^t is obtained through modeling global connectivity patterns by an aggregation operation as follows: $H_g^t = \sum_{i=0}^{\mathcal{N}_t} w_i^a \cdot F_{i\cdot}^t$, where $w_i^a \in \mathbb{R}^{1\times\mathcal{N}_t}$ is the aggregation operator.

Obviously, our edge convolution operation is different from the convolution operations on images. The spatial locality in our edge convolution refers to the local connectivity topology associated with a certain edge, while the locality of the image is the neighborhood of pixels, as shown in Fig. 3. Moreover, global connectivity is different from local connectivity, which is used in graph convolution. As Fig. 4 shown, the difference is that the local connectivity pattern refers to the independent connection between a node and its each neighbor, *e.g.*, e_{12} or e_{13}, while the global connectivity pattern refers to the edge set of the associated connections, *e.g.*, $\{e_{12}, e_{13}, e_{14}, e_{15}, e_{56}, e_{57}, e_{58}\}$, in which the associated connections

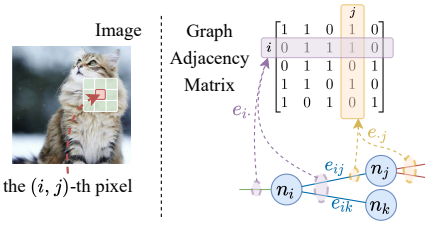


Figure 3: Illustration of the comparison on spatial locality between images and graphs. The difference is that the locality in images denotes the elements that are close together (for the (i, j)-th element, the locality is the 3×3 neighborhood), while the locality in the graph refers to the connectivity structure associated with each edge (for the (i, j)-th edge, the locality is the edge set of $e_{i\cdot}$ and $e_{\cdot j}$ associated the i-th node and j-th node), while the locality in the graph refers to the global connectivity structure associated with each node.

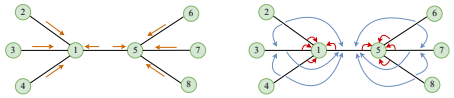


Figure 4: Comparison of local connectivity and global connectivity. The orange arrows represent the message passing from local neighbor nodes. The blue arrows represent the interrelated connections for generating the edge embeddings by considering the explicit topology patterns, and then the edge embeddings are aggregated into the node embedding globally (red arrows).

are interrelated and learned from each other. The edge sets associated with two nodes of n_1 and n_5 contain potential topology. To sufficiently leverage the edge set, the edge convolution operation globally exploits the topological association in the connectivities and enhances the useful edge embeddings.

Therefore, with the temporal-aware encoder, we obtain the temporal shift embedding $F_s^t = H_{l,s}^t + H_{g,s}^t$ and temporal consistency embedding $F_c^t = H_{l,c}^t + H_{g,c}^t$.

3.2.3 Cross decoder. The purpose of the cross decoder is to reconstruct the graph structure at t with the combination of the temporal shift and temporal consistency embeddings. We assume that the disentangled representation satisfies the following conditions: 1) An original graph can be reconstructed by its disentangled temporal shift and temporal consistency embeddings at the current time point. 2) An original graph can be reconstructed by its disentangled temporal shift embedding at the current point and the temporal consistency embedding disentangled from the original graph at the previous time point, as the temporal consistency embedding captures the evolution mode that changes slowly or remains consistent. Therefore, the cross decoder receives two inputs, *e.g.*, the temporal shift embedding and the temporal consistency embedding at t, or the temporal shift embedding at t and the temporal consistency embedding at $t-1$, and produces the hybrid embeddings $H^{\{t,t\}}$ or $H^{\{t,t-1\}}$, which is defined as: $H^{\{t,t\}} = \mathbf{MLP}(F_s^t + F_c^t)$; $H^{\{t,t-1\}} = \mathbf{MLP}(F_s^t + F_c^{t-1})$. Finally, we obtain the reconstructed graph adjacency matrices $\hat{A}^{\{t,t\}} = H^{\{t,t\}}(H^{\{t,t\}})^T$ and $\hat{A}^{\{t,t-1\}} = H^{\{t,t-1\}}(H^{\{t,t-1\}})^T$.

3.2.4 Loss. **(1) Temporal disentangling loss.** We assume that the temporal consistency embedding of graphs at two consecutive time points are similar, while the temporal shift embedding are dissimilar. Hence, we design a temporal disentangling loss:

$$\mathcal{L}_t = \sum_{t=2}^{T} \left[dis(F_c^t, F_c^{t-1}) - dis(F_s^t, F_s^{t-1}) + \beta \right] \tag{2}$$

where $dis(x, y) = \|x - y\|^2$ is the distance function, and β is the toleration value.

(2) Topology reconstruction loss. In order to learn meaningful embedding of graphs, we propose a reconstruction loss to constrain the topology consistency of the original and reconstructed graphs. The topology reconstruction loss is defined as:

$$\mathcal{L}_{rec} = \sum_{t=1}^{T} dis(\hat{A}^{\{t,t\}}, A^t) + \sum_{t=2}^{T} (dis(\hat{A}^{\{t,t-1\}}, A^t) + dis(\hat{A}^{\{t-1,t\}}, A^{t-1})) \tag{3}$$

(3) Discrepancy loss. From the graph structural similarity aspect, we assume that the reconstructed graph $\hat{A}^{\{t,t-1\}}$ from the combination embedding $H^{\{t,t-1\}}$ should be consistent with $\hat{A}^{\{t,t\}}$ from $H^{\{t,t\}}$. Hence, we propose a discrepancy loss to regularize the disentanglement process as follow:

$$\begin{aligned} \mathcal{L}_d = & \frac{1}{\mathcal{N}^t(\mathcal{N}^t - 1)} \sum_u \sum_{k \neq u} \mathbf{k}\left(\hat{A}_u^{\{t,t\}}, \hat{A}_k^{\{t,t\}}\right) \\ & - 2\frac{1}{\mathcal{N}^t \cdot \mathcal{N}^t} \sum_u \sum_k \mathbf{k}\left(\hat{A}_u^{\{t,t\}}, \hat{A}_k^{\{t,t-1\}}\right) \\ & + \frac{1}{\mathcal{N}^t(\mathcal{N}^t - 1)} \sum_u \sum_{k \neq u} \mathbf{k}\left(\hat{A}_u^{\{t,t-1\}}, \hat{A}_k^{\{t,t-1\}}\right) \end{aligned} \tag{4}$$

where $\mathbf{k}(\cdot)$ is the Gaussian kernel function.

Overall loss. With the above defined losses, the overall loss of the pre-training stage can be defined as: $\mathcal{L}_{pt} = \mathcal{L}_t + \lambda_{rec}\mathcal{L}_{rec} + \lambda_d\mathcal{L}_d$ where λ_{rec} denotes the weight of $\mathcal{L}_{rec}$, and λ_d denotes the weight of $\mathcal{L}_d$. In this paper, we set $\lambda_{rec} = 0.01$, $\lambda_d = 0.05$.

After the pre-training stage, we leverage the cross-entropy loss to supervise the link prediction, node classification and edge classification tasks. Besides, We found that incorporating the pre-training loss of representation disentanglement as an auxiliary objective to the fine-tuning downstream tasks facilitates improving the generalization of the supervised model and accelerates convergence. Hence, we preserve the loss $\mathcal{L}_{pt}$ used during the pre-training stage by a ratio of λ_{pt}, we set $\lambda = 0.5$ in this paper. This allows the model to further learn the temporal shift information under the guidance of the graph-related tasks.

4 EXPERIMENTS

In this section, we aim to answer the following three questions to analyze dynamic node embedding quality:

- **Q1:** How does only the disentangled temporal shift embedding behave on link prediction, edge classification and node classification tasks?

Table 2: Summary of datasets and task split. lp: link prediction. ec: edge classification. nc: node classification

Name	#Edges	#Nodes	#Snapshots	train(%)/val(%)/test(%)
Reddit-B	286,561	35,776	178	lp: 80/10/10
Reddit-T	571,927	54,075	178	lp: 80/10/10
UCI-M	59,835	1,899	29	lp: 80/10/10
Bitcoin-OTC	35,592	5,881	279	lp: 80/10/10; ec: 70/10/20
Bitcoin-Alpha	24,186	3,783	274	lp: 80/10/10; ec: 70/10/20
Elliptic	234,355	203,769	49	nc: 65/10/25

- **Q2:** Can STDGL learn the potential and transferable temporal node embedding via pre-training, which is beneficial for various graph analysis tasks?
- **Q3:** How does our self-supervised learning paradigm assist in understanding dynamic graphs via disentanglement?

4.1 Datasets and Evaluation Metrics

We evaluate our model on six different datasets. Concretely, Reddit-T and Reddit-B are networks of hyperlinks in titles and bodies of Reddit posts, respectively [12]. UCI-M dataset consists of private messages sent on an online social network system among students [24]. Bitcoin-OTC and Bitcoin-Alpha contain who-trusts-whom networks of people who trade on the OTC and Alpha platforms [13, 14]. Elliptic is a bitcoin transaction network, where each node represents one transaction and the edges indicates the payment flows. The aim is to categorize the unlabeled transactions. Table 2 provides summary statistics of the above-mentioned datasets.

We choose mean reciprocal rank (MRR) and AUC as evaluation metrics for link prediction. For each node v with a positive (real) edge (v, u) at $t + 1$, we randomly sample 100 negative (nonexistent) edges emitting from v and predict the rank of edge (v, u)'s score among all other negative edges. MRR score is the mean of reciprocal ranks over all nodes u. We run each experiment with 3 random seeds following the experimental setting in [31]. We choose the F1 score for both edge classification and node classification tasks to evaluate the model performance.

4.2 Methods for Comparison

To comprehensively evaluate the performance of multiple tasks, we compare our STDGL with state-of-the-art models. Specifically, the comparable methods can be grouped into three categories: static graph embedding learning (*i.e.*, Graph Attention Networks (GAT) [29] and GraphSAGE [6]), supervised dynamic graph embedding learning (*i.e.*, Graph Convolutional Recurrent Networks (GCRN) [27], Evolving Graph Convolutional Networks (EvolveGCN) [25], Temporal Graph Convolutional network (TGCN) [33], DGCRN [15], CAW-N [30] and DySAT [26]) and self-supervised dynamic graph embedding leaning (*i.e.*, Debiased Dynamic Graph Contrastive Learning (DDGCL) [28]). We compare our STDGL with GCN, GCRN, EvolveGCN and DDGCL on edge or node classification tasks.

4.3 Results analysis (Answer for Q1)

We evaluate four models: STDGL *-w/o* Pre-training , STDGL-*s* (utilizing temporal shift embedding), STDGL-*c* (utilizing temporal consistency embedding) and STDGL-*both* (utilizing both embeddings). The results in Table 3 and 4 show that STDGL-*s* performs better than the comparable methods, achieving a new state-of-the-art in most cases, which demonstrates that the proposed self-supervised learning paradigm can provide a promising solution to model complex and nonlinearly evolving processes of the dynamic graph, leading to improved performance among graph-related tasks. In addition, we highlight the following observations:

Link prediction. We can observe that STDGL-*s* and STDGL-*both* perform generally better than other supervised and self-supervised graph learning models across all datasets except the UCI-M dataset. Especially, STDGL-*s* outperforms STDGL-*c* and STDGL-*both* across all datasets, which suggests the importance of temporal shift information in understanding dynamic graphs for link prediction. However, STDGL-*c* does not yield better performance, and STDGL-*both*, which utilizes both embeddings, also performs inferior to STDGL-*s*, indicating that the learned temporal consistency embedding does not contribute to the link prediction. The reason is that the temporal consistency embedding fails when nodes exhibit significant and complicated evolutionary behaviors in the dynamic graphs. Moreover, STDGL-*s* significantly outperforms the self-supervised method, DDGCL, across all the datasets, notably on the Bitcoin-OTC dataset, achieving improvements of 9.9% and 14.1% in terms of MRR and AUC, respectively. This demonstrates the benefits and potential of a well-designed pretext task that explores temporal information components for dynamic graph learning. Since the MRR score evaluates the dynamic graph model across all time points, it fails to reflect the model performance in modeling short-term evolution patterns [31]. Thus, to evaluate the model performance of modeling embeddings with short-term data, we report MRR@10 and MRR@20, which calculate the MRR score at only the earliest 10% and 20% time points, respectively. It is a more challenging task due to the limited available samples and insufficient temporal correlation. The results show that STDGL-*s* outperforms other methods in terms of MRR@10 and MRR@20, indicating that temporal shift embedding is robust for the short-term link prediction.

Node classification. We perform the task of node classification on the Elliptic dataset. Following the setting in [25], we only report the minority F1 value here. We can see that all the three version methods outperform the compared models, and STDGL-*Both* achieves the best result, indicating the effectiveness of STDGL on the node classification task. Besides, we can see that with the only temporal shift embedding, our model can achieve better performance than the temporal consistency embedding on the node classification task, which further validates our assumptions.

Edge classification. From the results shown in Table 4, we can observe that STDGL-*c* obtains the best F1 score (87.6%) on the Bitcoin-Alpha dataset, and slightly underperform STDGL-*s* on the Bitcoin-OTC dataset, indicating that STDGL-*c* is effective in edge classification. Interestingly, on the same Bitcoin-Alpha dataset, STDGL-*c* performs well in edge classification but behaviors poorly in link prediction, while STDGL-*s* performs better for link prediction than edge classification. It demonstrates the distinct contributions of temporal shift and consistency embeddings to different tasks on the same dataset. Moreover, STDGL-*s* yields the best performance on the Bitcoin-OTC dataset but performs inferior to the other two versions on the Bitcoin-Alpha dataset, demonstrating that various dynamic graph datasets exhibit different graph evolution modes, which have distinct impacts on temporal graph learning.

Table 3: Comparison with the state-of-the-art link prediction methods on the five datasets in terms of MRR and AUC. The best results are bold in red and the second best results are bold in blue. Pre: Pre-training.

Model	Reddit-T				Reddit-B				UCI-M				Bitcoin-OTC				Bitcoin-Alpha			
	MRR@10	MRR@20	MRR	AUC	MRR@10	MRR@20	MRR	AUC	MRR@10	MRR@20	MRR	AUC	MRR@10	MRR@20	MRR	AUC	MRR@10	MRR@20	MRR	AUC
GAT [29]	25.1±1.4	24.4±1.5	27.1±1.4	65.5±1.7	17.8±1.0	17.0±1.4	19.8±1.7	61.4±0.9	6.8±1.0	8.9±1.1	8.8±0.9	65.6±1.5	3.8±0.9	4.5±1.6	7.6±1.7	54.7±0.9	13.2±1.6	13.5±1.2	14.5±1.4	57.3±1.4
GraphSAGE [6]	30.9±1.4	27.8±1.0	33.1±1.8	68.1±1.7	19.2±1.6	22.9±1.8	24.3±1.5	63.3±0.9	8.2±1.7	9.4±1.5	10.3±1.4	68.8±1.9	7.1±1.8	7.5±1.8	8.8±1.1	56.6±1.6	15.9±1.7	16.3±1.5	15.1±0.7	60.3±2.1
GCRN [27]	31.6±1.7	32.9±0.8	33.8±1.4	72.9±1.5	19.0±1.6	19.3±1.6	21.7±0.9	67.9±0.8	10.4±0.9	8.8±1.5	8.9±0.7	71.2±1.6	19.1±0.8	17.1±1.9	17.3±1.5	64.1±1.6	16.5±0.9	18.8±1.5	21.0±1.4	64.8±0.7
EvolveGCN [25]	29.5±0.8	34.0±1.3	35.1±1.9	69.7±1.7	27.7±0.8	35.6±1.3	43.1±1.5	87.9±0.8	9.2±1.5	13.3±0.9	14.6±1.6	76.9±1.4	15.3±0.9	14.9±1.5	14.2±1.4	65.1±0.8	20.4±1.4	27.1±1.6	23.4±0.8	79.3±1.0
TGCN [33]	43.1±1.1	46.2±1.2	49.1±1.4	65.3±0.8	51.0±1.5	53.7±0.9	55.1±1.6	82.1±1.8	10.3±1.1	13.9±1.5	13.0±1.6	74.0±0.9	14.4±0.7	14.9±1.4	18.3±1.6	63.0±0.9	20.4±1.7	18.8±0.8	16.9±1.5	75.0±0.7
DGCRN [15]	49.4±0.6	56.6±0.5	66.1±0.9	95.8±0.8	48.5±1.8	54.3±1.4	57.3±1.3	95.1±0.8	19.1±0.7	22.6±1.4	23.9±0.9	86.5±1.5	24.1±0.8	19.9±1.6	19.4±0.9	77.9±1.4	27.7±0.7	**30.4±1.6**	35.1±1.0	85.9±1.1
CAW-N [30]	46.4±1.8	46.7±1.9	46.8±2.3	94.1±1.6	45.9±1.5	45.4±1.7	46.0±0.8	93.1±1.3	**28.4±0.9**	**28.8±0.7**	28.4±1.7	**92.7±0.8**	23.8±1.6	23.3±0.9	24.9±1.8	80.1±1.5	26.2±1.5	27.3±0.8	29.2±1.8	75.0±0.9
DySAT [26]	48.1±0.9	51.8±1.7	60.2±1.5	92.8±1.1	54.6±2.1	57.4±0.7	60.2±1.9	90.7±0.7	19.1±1.6	21.3±1.8	**29.3±0.9**	83.9±1.3	**27.1±1.4**	**27.6±0.4**	**26.2±1.4**	82.7±1.7	25.3±0.8	23.9±1.5	26.8±0.8	87.3±0.7
DDGCL [28]	51.2±1.6	55.7±0.8	64.3±1.7	96.3±1.9	**57.1±0.6**	**59.9±1.8**	**64.1±1.8**	95.5±1.4	23.1±1.5	24.8±1.9	27.1±1.6	85.8±1.8	18.4±1.8	19.3±1.5	18.8±0.9	71.3±0.8	27.0±1.3	**28.1±1.9**	**35.4±1.6**	**88.2±0.9**
STDGL -*w/o* Pre	50.1±1.3	56.5±0.7	66.3±0.6	95.8±0.5	50.4±1.3	56.6±1.5	61.2±1.7	96.6±1.4	20.1±0.8	23.9±0.8	23.9±0.7	86.8±0.6	24.4±1.7	21.5±1.8	24.1±1.4	80.9±0.8	27.6±1.4	25.8±1.1	31.0±0.7	83.3±1.5
STDGL-*c*	51.0±1.4	56.7±1.3	66.2±1.3	95.9±0.6	50.5±1.1	56.6±0.9	61.7±0.9	96.7±1.0	21.3±1.2	24.1±1.1	23.9±0.4	87.4±0.7	25.1±0.9	21.6±0.5	24.5±0.8	81.1±1.6	27.9±1.2	25.7±1.1	31.5±0.7	84.5±0.8
STDGL-*s*	**58.0±0.7**	**61.9±0.8**	**69.0±0.9**	**98.0±0.6**	**58.8±0.7**	**63.1±0.8**	**67.8±0.6**	**97.2±0.7**	**24.3±1.4**	**26.5±0.8**	**29.5±0.3**	**90.3±0.5**	**29.7±1.5**	**28.0±0.7**	**28.7±0.6**	**85.4±0.8**	**29.4±0.9**	25.7±1.7	**36.0±1.6**	**87.4±0.9**
STDGL-*both*	**57.1±0.8**	**60.3±1.5**	**68.1±0.8**	**97.2±0.7**	54.4±0.8	57.8±0.6	63.8±0.8	**97.1±0.9**	22.0±0.9	24.8±0.7	25.3±0.6	89.1±0.8	26.9±0.7	23.3±0.7	25.5±0.9	**84.1±1.5**	**28.9±0.8**	24.5±1.4	32.8±0.9	84.5±1.1

Table 6: Comprehensive ablation experiments to verify the effectiveness of each proposed component.

Model	Reddit-T		Reddit-B	
	MRR	AUC	MRR	AUC
STDGL -*w/o* E_l	65.8	94.7	65.1	97.0
STDGL -*w/o* E_g	67.1	96.2	67.2	97.1
STDGL -*w/o* Pre-training	66.3	95.8	61.2	96.6
STDGL-*s* - Pre-training on Reddit-T	-	-	64.2	97.1
STDGL-*s* - Pre-training on Reddit-B	67.1	97.5	-	-
DDGCL (linear probing)	45.3	62.1	43.9	61.8
DDGCL (fine-tuning)	66.3	95.8	61.2	96.6
STDGL-*s* (linear probing)	50.4	63.3	45.9	62.6
STDGL-*s* (fine-tuning)	**69.0**	**98.0**	**67.8**	**97.2**

Table 4: Comparison of node/edge classification performance on three datasets in terms of F1. The two best results are bold in red and blue, respectively.

Model	Node classification	Edge classification	
	Elliptic	Bitcoin-OTC	Bitcoin-Alpha
GCN [11]	42.9±1.9	80.1±1.3	77.3±1.6
GCRN [27]	60.9±1.4	82.2±1.7	68.1±1.2
EvolveGCN [25]	57.3±1.2	86.2±0.9	81.0±1.4
DDGCL [28]	60.3±1.8	88.1±1.2	82.3±1.1
STDGL-*c*	61.5±0.8	**88.9±0.4**	**87.6±0.6**
STDGL-*s*	**61.8±1.1**	**89.2±0.7**	86.5±0.5
STDGL-*both*	**62.5±0.9**	88.6±0.8	**86.9±0.3**

Table 5: Effectiveness of the proposed components in STDGL. The best results are bold. Pre: Pre-training. Bit-O: Bitcoin-OTC. Bit-A: Bitcoin-Alpha.

Model	Losses			S	Reddit-T		Reddit-B		UCI-M		Bit-O		Bit-A	
	$\mathcal{L}_t$	$\mathcal{L}_{rec}$	$\mathcal{L}_d$		MRR	AUC	MRR	AUC	MRR	AUC	MRR	AUC	MRR	AUC
STDGL -*w/o* Pre	-	-	-	-	66.3	95.8	61.2	96.6	23.9	86.8	24.1	80.9	31.0	83.3
STDGL-*s*	✓				66.5	96.0	61.6	96.7	24.4	86.1	25.5	81.8	33.4	84.7
		✓	✓		67.4	96.9	63.8	96.9	26.1	87.7	27.1	82.2	35.0	84.8
	✓	✓	✓		68.2	97.4	65.4	97.0	27.9	89.4	27.6	83.1	35.1	85.1
	✓	✓	✓	✓	**69.0**	**98.0**	**67.8**	**97.2**	**28.0**	**89.5**	**28.7**	**85.4**	**36.0**	**87.1**

4.4 Ablation Study

We show an ablation analysis of different losses and the semantic augmentation matrix S in STDGL-*s* in Table 5. Compared with STDGL -*w/o* pretraining, STDGL-*s* with $\mathcal{L}_t$ achieves improved performance, validating its advantages for the various downstream tasks. The improvement achieved by STDGL-*s* using $\mathcal{L}_{rec}$ and $\mathcal{L}_d$ indicates the capability of encoder-decoder framework in capturing the hidden embeddings. STDGL-*s* with all three losses performs a higher result than using only $\mathcal{L}_t$ or both $\mathcal{L}_{rec}$ and $\mathcal{L}_d$, indicates these multiple losses can collaborate boosts the disentangled representation learning. Besides, we observed that semantic augmentation matrix S yields a significant improvement to our model, especially improvements of 0.8% MRR and 0.6% AUC obtained by incorporating the semantic matrix on Reddit-T dataset, respectively.

In addition, to justify the effectiveness of the local connectivity and global connectivity encoders, a careful ablation study is conducted on the Reddit-T and Reddit-B datasets (the first part of Table 6). It corroborates that the local connectivity and global connectivity encoders are both effective and necessary for node embedding learning. This result also demonstrates that modeling the node connectivity from the local and global perspectives is complementary. In other words, modeling global connectivity provides a powerful complement to the common graph convolution operation.

4.5 Discussion

4.5.1 Partial fine-tuning. We evaluate the pre-training performance of STDGL-*s* and DDGCL [28] with two strategies including fine-tuning and linear probing. The results in Table 6 demonstrate that our STDGL-*s* surpasses DDGCL on Reddit-T and Reddit-B datasets in both fine-tuning and linear probing, which indicates that STDGL-*s* is capable of capturing the critical embedding of the data itself in the pre-training and yield better generalized representations regardless of linear probing and fine-tuning.

4.5.2 Transfer learning (Answer for Q2). To further investigate the generalization ability of STDGL in representation learning, we compare the pre-trained models from pre-trained individually on the same and different datasets with the model from scratch, the results are shown in the second part of Table 6. We can find that the model pre-trained from the Reddit-T/Reddit-B for link prediction on Reddit-B/Reddit-T achieves better performance than the model from scratch, demonstrating that STDGL-*s* can learn more transferable representations to achieve better generalization and robustness via pre-training. The results again demonstrate that self-supervised learning enables the model to learn more generalized informative knowledge through well-designed pretext tasks from the data itself to achieve better performance.

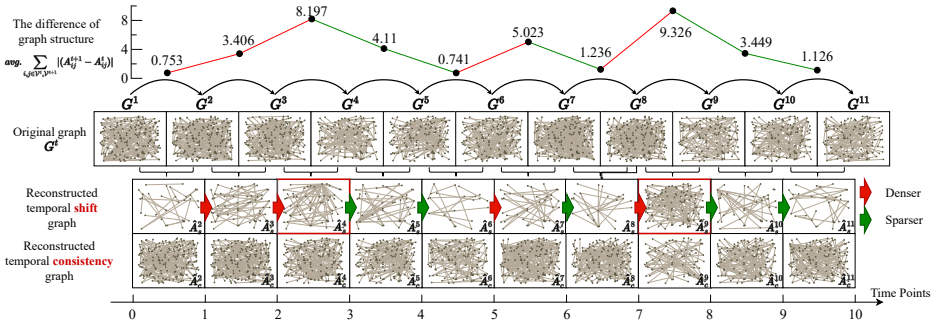


Figure 5: Visualization of the graph evolution on the Bitcoin-OTC dataset. The temporal graph evolution is shown *wrt.* the difference of graph structure. We also visualize the original graph G^t, the graph A_s^t reconstructed by temporal shift embedding F_s^t, and the graph A_c^t reconstructed by temporal consistency embedding F_c^t to better analyze the graph evolution trend. The A_s^t and A_c^t are obtained by the consecutive graph G^{t-1} and G^t with temporal shift and consistency embeddings, respectively.

Table 7: Results of various disentangling by number of consecutive graphs.

Model	Consecutive	Reddit-B		Bitcoin-OTC	Elliptic
		MRR	AUC	F1	F1
STDGL-*s*	2	**67.8**	**97.2**	**89.2**	**61.8**
	3	64.9	96.4	84.9	58.0
	4	62.2	96.1	83.1	58.1

4.5.3 The effect of the number of consecutive graphs. Our goal is to investigate whether more successive graphs engaged can better disentangle the embeddings and improve the performance by better modeling the temporal correlation. We conducted experiments on the three downstream tasks, as shown in Table. 7. We find that using three or four successive graphs becomes worse compared to only two consecutive graphs, which suggests that the most recent information is more influential than the earlier ones. This observation is also consistent with the Markov assumption, which also suggests that the current state is mainly influenced by the nearest state [7]. Moreover, involving more successive graphs leads to introducing irrelevant information and noise to the disentanglement process, limiting the model's ability to capture the evolution dependencies of graphs. Besides, if leveraging more than two successive graphs, the amount of available nodes which simultaneously exist across the multiple adjacent graphs becomes limited, resulting in insufficient data for training our model.

4.5.4 Visualization of temporal shift and temporal consistency graphs (Answer for Q3). In order to verify that temporal shift embedding can reflect the critical temporal evolution mode, we provide a qualitative analysis showing that the reconstructed temporal shift and consistency graphs through the disentangled latent embeddings are consistent with the evolution of the original temporal graphs. In this study, the original temporal graph evolution is intuitively reflected by the structural difference between graphs G^t and G^{t+1}. From Fig. 5, We observe a global trend where a larger structural difference corresponds to a higher density of temporal shift graphs, *e.g.*, $\hat{A}_s^4$ and $\hat{A}_s^9$, demonstrating that the temporal shift embedding can capture significant characteristics of graph evolution. Interestingly, the temporal shift graphs are more sparse than the original graphs and temporal consistency graphs, and there is a significant difference between the two consecutive temporal shift graphs. It is obvious that the reconstructed temporal shift graphs are consistent with the perceptual difference in graph structure. The temporal shift information provides insight into the inherent evolving patterns and allows us to understand the natural mechanisms of dynamic graph analysis task By disentangling the temporal shift information from temporal consistency information, we can make a deeper analysis of the dynamic graph's evolution nature.

5 CONCLUSION

Traditional dynamic graph embedding learning models have the limitations of insufficient consideration of complex temporal characteristics and their impacts on graph dynamics. In this study, we design a Self-supervised Temporal-aware Dynamic Graph representation Learning framework named STDGL for dynamic graph learning to address the mentioned limitations. Since dynamic graphs evolve in terms of node and edge attributes over time, STDGL learns dynamic node embeddings from both the node local as well as global connectivity perspectives and disentangles the graph properties into temporal shift and temporal consistency parts during pre-training. To constrain the disentangling, we introduce temporal disentangling loss, topology reconstruction loss and discrepancy loss to ensure the identification of temporal shift and temporal consistency embeddings. Then, the disentangled temporal embeddings are used for the dynamic graph tasks including link prediction, edge classification and node classification. Our experiment results on real-world datasets indicate significant performance over existing static and dynamic graph embedding learning methods, and suggest self-supervised learning based on well-designed representation disentangling pretext task is essential for dynamic graph learning.

6 ACKNOWLEDGMENTS

This research was supported by the National Key Research and Development Program of China (No.2020YFC0833302), the National Natural Science Foundation of China under Grant 62076059 and in part by the Science Project of Liaoning province under Grant 2021-MS-105.

REFERENCES

[1] Piotr Bielak, Tomasz Kajdanowicz, and Nitesh V Chawla. 2022. Attre2vec: Unsupervised attributed edge representation learning. *Information Sciences* 592 (2022), 82–96.

[2] Hongxu Chen, Hongzhi Yin, Weiqing Wang, Hao Wang, Quoc Viet Hung Nguyen, and Xue Li. 2018. PME: Projected Metric Embedding on Heterogeneous Networks for Link Prediction. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining* (London, United Kingdom) *(KDD '18)*. Association for Computing Machinery, New York, NY, USA, 1177–1186. https://doi.org/10.1145/3219819.3219986

[3] Ke-Jia Chen, Jiajun Zhang, Linpu Jiang, Yunyun Wang, and Yuxuan Dai. 2022. Pre-training on dynamic graph neural networks. *Neurocomputing* 500 (2022), 679–687.

[4] Wei-Lin Chiang, Xuanqing Liu, Si Si, Yang Li, Samy Bengio, and Cho-Jui Hsieh. 2019. Cluster-GCN: An Efficient Algorithm for Training Deep and Large Graph Convolutional Networks. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining* (Anchorage, AK, USA) *(KDD '19)*. Association for Computing Machinery, New York, NY, USA, 257–266. https://doi.org/10.1145/3292500.3330925

[5] Shuyun Gu, Xiao Wang, Chuan Shi, and Ding Xiao. 2022. Self-supervised Graph Neural Networks for Multi-behavior Recommendation. In *Proceedings of the Thirty-First International Joint Conference on Artificial Intelligence, IJCAI-22*, Lud De Raedt (Ed.). International Joint Conferences on Artificial Intelligence Organization, Vienna, Austria, 2052–2058. https://doi.org/10.24963/ijcai.2022/285 Main Track.

[6] Will Hamilton, Zhitao Ying, and Jure Leskovec. 2017. Inductive Representation Learning on Large Graphs. In *Advances in Neural Information Processing Systems*, I. Guyon, U. Von Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett (Eds.), Vol. 30. Curran Associates, Inc., Long Beach, CA, USA. https://proceedings.neurips.cc/paper_files/paper/2017/file/5dd9db5e033da9c6fb5ba83c7a7ebea9-Paper.pdf

[7] Ruining He, Wang-Cheng Kang, and Julian McAuley. 2017. Translation-Based Recommendation. In *Proceedings of the Eleventh ACM Conference on Recommender Systems* (Como, Italy) *(RecSys '17)*. Association for Computing Machinery, New York, NY, USA, 161–169. https://doi.org/10.1145/3109859.3109882

[8] Tingsong Jiang, Tianyu Liu, Tao Ge, Lei Sha, Baobao Chang, Sujian Li, and Zhifang Sui. 2016. Towards Time-Aware Knowledge Graph Completion. In *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers*, Yuji Matsumoto and Rashmi Prasad (Eds.). The COLING 2016 Organizing Committee, Osaka, Japan, 1715–1724. https://aclanthology.org/C16-1161

[9] Tingsong Jiang, Tianyu Liu, Tao Ge, Lei Sha, Sujian Li, Baobao Chang, and Zhifang Sui. 2016. Encoding Temporal Information for Time-Aware Link Prediction. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, Jian Su, Kevin Duh, and Xavier Carreras (Eds.). Association for Computational Linguistics, Austin, Texas, USA, 2350–2354. https://doi.org/10.18653/v1/D16-1260

[10] Bowen Jin, Chen Gao, Xiangnan He, Depeng Jin, and Yong Li. 2020. Multi-Behavior Recommendation with Graph Convolutional Networks. In *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval* (Virtual Event, China) *(SIGIR '20)*. Association for Computing Machinery, New York, NY, USA, 659–668. https://doi.org/10.1145/3397271.3401072

[11] Thomas N. Kipf and Max Welling. 2017. Semi-Supervised Classification with Graph Convolutional Networks. In *International Conference on Learning Representations*. arXiv, Palais des Congrès Neptune, Toulon, France, 1–14.

[12] Srijan Kumar, William L. Hamilton, Jure Leskovec, and Dan Jurafsky. 2018. Community Interaction and Conflict on the Web. In *Proceedings of the 2018 World Wide Web Conference* (Lyon, France) *(WWW '18)*. International World Wide Web Conferences Steering Committee, Republic and Canton of Geneva, CHE, 933–943. https://doi.org/10.1145/3178876.3186141

[13] Srijan Kumar, Bryan Hooi, Disha Makhija, Mohit Kumar, Christos Faloutsos, and V.S. Subrahmanian. 2018. REV2: Fraudulent User Prediction in Rating Platforms. In *Proceedings of the Eleventh ACM International Conference on Web Search and Data Mining* (Marina Del Rey, CA, USA) *(WSDM '18)*. Association for Computing Machinery, New York, NY, USA, 333–341. https://doi.org/10.1145/3159652.3159729

[14] Srijan Kumar, Francesca Spezzano, VS Subrahmanian, and Christos Faloutsos. 2016. Edge weight prediction in weighted signed networks. In *2016 IEEE 16th International Conference on Data Mining (ICDM)*. IEEE, IEEE, Barcelona, Spain, 221–230.

[15] Fuxian Li, Jie Feng, Huan Yan, Guangyin Jin, Fan Yang, Funing Sun, Depeng Jin, and Yong Li. 2023. Dynamic graph convolutional recurrent network for traffic prediction: Benchmark and solution. *ACM Transactions on Knowledge Discovery from Data* 17, 1 (2023), 1–21.

[16] Yu Li, Yuan Tian, Jiawei Zhang, and Yi Chang. 2020. Learning Signed Network Embedding via Graph Attention. *Proceedings of the AAAI Conference on Artificial Intelligence* 34, 04 (2020), 4772–4779. https://doi.org/10.1609/aaai.v34i04.5911

[17] Siyuan Liao, Shangsong Liang, Zaiqiao Meng, and Qiang Zhang. 2021. Learning Dynamic Embeddings for Temporal Knowledge Graphs. In *Proceedings of the 14th ACM International Conference on Web Search and Data Mining* (Virtual Event, Israel) *(WSDM '21)*. Association for Computing Machinery, New York, NY, USA, 535–543. https://doi.org/10.1145/3437963.3441741

[18] Kai Lin, Biao Jie, Peng Dong, Xintao Ding, Weixin Bian, and Mingxia Liu. 2022. Convolutional recurrent neural network for dynamic functional mri analysis and brain disease identification. *Frontiers in Neuroscience* 16 (2022), 933660.

[19] Jiaying Liu, Feng Xia, Xu Feng, Jing Ren, and Huan Liu. 2022. Deep graph learning for anomalous citation detection. *IEEE Transactions on Neural Networks and Learning Systems* 33, 6 (2022), 2543–2557.

[20] Lingwen Liu, Guangqi Wen, Peng Cao, Tianshun Hong, Jinzhu Yang, Xizhe Zhang, and Osmar R. Zaiane. 2023. BrainTGL: A dynamic graph representation learning model for brain network analysis. *Computers in Biology and Medicine* 153 (2023), 106521. https://doi.org/10.1016/j.compbiomed.2022.106521

[21] Sedigheh Mahdavi, Shima Khoshraftar, and Aijun An. 2020. Dynamic Joint Variational Graph Autoencoders. In *Machine Learning and Knowledge Discovery in Databases*, Peggy Cellier and Kurt Driessens (Eds.). Springer International Publishing, Cham, 385–401.

[22] Maximilian Nickel, Lorenzo Rosasco, and Tomaso Poggio. 2016. Holographic Embeddings of Knowledge Graphs. *Proceedings of the AAAI Conference on Artificial Intelligence* 30, 1 (Mar. 2016). https://doi.org/10.1609/aaai.v30i1.10314

[23] Andreas Nugaard Holm, Barbara Plank, Dustin Wright, and Isabelle Augenstein. 2022. Longitudinal Citation Prediction using Temporal Graph Neural Networks. In *AAAI 2022 Workshop on Scientific Document Understanding (SDU 2022)*. AAAI Press, United States.

[24] Pietro Panzarasa, Tore Opsahl, and Kathleen M Carley. 2009. Patterns and dynamics of users' behavior and interaction: Network analysis of an online community. *Journal of the American Society for Information Science and Technology* 60, 5 (2009), 911–932.

[25] Aldo Pareja, Giacomo Domeniconi, Jie Chen, Tengfei Ma, Toyotaro Suzumura, Hiroki Kanezashi, Tim Kaler, Tao Schardl, and Charles Leiserson. 2020. EvolveGCN: Evolving Graph Convolutional Networks for Dynamic Graphs. *Proceedings of the AAAI Conference on Artificial Intelligence* 34, 04 (Apr. 2020), 5363–5370. https://doi.org/10.1609/aaai.v34i04.5984

[26] Aravind Sankar, Yanhong Wu, Liang Gou, Wei Zhang, and Hao Yang. 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* (Houston, TX, USA) *(WSDM '20)*. Association for Computing Machinery, New York, NY, USA, 519–527. https://doi.org/10.1145/3336191.3371845

[27] Youngjoo Seo, Michaël Defferrard, Pierre Vandergheynst, and Xavier Bresson. 2018. Structured Sequence Modeling with Graph Convolutional Recurrent Networks. In *Neural Information Processing*, Long Cheng, Andrew Chi Sing Leung, and Seiichi Ozawa (Eds.). Springer International Publishing, Cham, 362–373.

[28] Sheng Tian, Ruofan Wu, Leilei Shi, Liang Zhu, and Tao Xiong. 2021. Self-Supervised Representation Learning on Dynamic Graphs. In *Proceedings of the 30th ACM International Conference on Information & Knowledge Management* (Virtual Event, Queensland, Australia) *(CIKM '21)*. Association for Computing Machinery, New York, NY, USA, 1814–1823. https://doi.org/10.1145/3459637.3482389

[29] Petar Velickovic, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Lio, and Yoshua Bengio. 2017. Graph attention networks. *stat* 1050 (2017), 20.

[30] Yanbang Wang, Yen-Yu Chang, Yunyu Liu, Jure Leskovec, and Pan Li. 2021. Inductive Representation Learning in Temporal Networks via Causal Anonymous Walks. In *International Conference on Learning Representations*. Association for Computing Machinery, New York, NY, USA.

[31] Jiaxuan You, Tianyu Du, and Jure Leskovec. 2022. ROLAND: Graph Learning Framework for Dynamic Graphs. In *Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining* (Washington DC, USA) *(KDD '22)*. Association for Computing Machinery, New York, NY, USA, 2358–2366. https://doi.org/10.1145/3534678.3539300

[32] Jiasheng Zhang, Shuang Liang, Yongpan Sheng, and Jie Shao. 2022. Temporal knowledge graph representation learning with local and global evolutions. *Knowledge-Based Systems* 251 (2022), 109234.

[33] Ling Zhao, Yujiao Song, Chao Zhang, Yu Liu, Pu Wang, Tao Lin, Min Deng, and Haifeng Li. 2019. T-gcn: A temporal graph convolutional network for traffic prediction. *IEEE Transactions on Intelligent Transportation Systems* 21, 9 (2019), 3848–3858.

Received 20 February 2007; revised 12 March 2009; accepted 5 June 2009