Neural Networks 174 (2024) 106240

Contents lists available at ScienceDirect

Neural Networks

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

Full Length Article

Contrastive representation learning on dynamic networks

Pengfei Jiao [a,c], Hongjiang Chen [a], Huijun Tang [a], Qing Bao [a], Long Zhang [b], Zhidong Zhao [a,c,*], Huaming Wu [d,*]

[a] School of Cyberspace, Hangzhou Dianzi University, Hangzhou, 310018, China
[b] College of Intelligence and Computing, Tianjin University, Tianjin, 300350, China
[c] Data Security Governance Zhejiang Engineering Research Center, Hangzhou, 310018, China
[d] Center for Applied Mathematics, Tianjin University, Tianjin, 300072, China

ARTICLE INFO

Keywords:
Dynamic network
Contrastive learning
Mutual information
Representation learning

ABSTRACT

Representation learning for dynamic networks is designed to learn the low-dimensional embeddings of nodes that can well preserve the snapshot structure, properties and temporal evolution of dynamic networks. However, current dynamic network representation learning methods tend to focus on estimating or generating observed snapshot structures, paying excessive attention to network details, and disregarding distinctions between snapshots with larger time intervals, resulting in less robustness for sparse or noisy networks. To alleviate these challenges, this paper proposes a contrastive mechanism for temporal representation learning on dynamic networks, inspired by the success of contrastive learning in visual and static network representation learning. This paper proposes a novel Dynamic Network Contrastive representation Learning (DNCL) model. Specifically, contrast objective functions are constructed using intra-snapshot and inter-snapshot contrasts to capture the network topology, node feature information, and network evolution information, respectively. Rather than estimating or generating ground-truth network features, the proposed approach maximizes mutual information between nodes from different time steps and views generated. The experimental results of link prediction, node classification, and clustering on several real-world and synthetic networks demonstrate the superiority of DNCL over state-of-the-art methods, indicating the effectiveness of the proposed approach for dynamic network representation learning.

1. Introduction

Real-world complex systems such as social networks (Huang, Shang, Lin, Fu, & Wang, 2018), co-authorship networks (Gehrke, Ginsparg, & Kleinberg, 2003), and protein-protein interaction networks (Theocharidis, Van Dongen, Enright, & Freeman, 2009) can be rendered into networks where the representation of individual entities occurs as nodes and interactions between nodes signify links. These networks display dynamic characteristics — both their structure (topology) and attributes of nodes demonstrate mutations over time (Jiao et al., 2021). For example, in social networks, events of communication like emails and texts transpire recurrently, whilst friendships transition. Co-authorship networks mirror a similar dynamism, as changes happen in the patterns of collaboration periodically among authors. Thus, the investigation of dynamic networks becomes critical to comprehend the evolution process of complex systems.

There is a pertinent need to efficiently represent dynamic networks in a manner that captures the structural and evolutionary information at hand. Network representation learning, which uses its topology and node attributes to map a network onto a low-dimensioned Euclidean space, is a prevalent technique. This mapping allows for the usage of machine learning methods in network analysis. It has proven to be successful across various applications in the real world, such as in node/edge classification and predictive modeling for links. While static networks have seen significant advancements in representation learning (Grover & Leskovec, 2016; Kipf & Welling, 2016, 2017a; Perozzi, Al-Rfou, & Skiena, 2014; Tang et al., 2015), dynamic networks present a unique set of challenges due to their complexity and continuously shifting nature. These challenges encompass nodes that join or leave, links that appear or disintegrate, and communities that merge or divide. All of these demand the creation of effective algorithms aimed at the representation learning of dynamic networks (see Fig. 1).

In order to address the aforementioned challenges, an array of dynamic network representation learning methodologies have been proposed (Chen, Jiao, Tang, & Wu, 2023; Goyal, Chhetri, & Canedo, 2020; Goyal, Kamra, He, & Liu, 2018; Han et al., 2021; Jiao, Li, et al., 2022; Zhou, Yang, Ren, Wu, & Zhuang, 2018) in recent times. These

* Corresponding authors.
E-mail addresses: zhaozd@hdu.edu.cn (Z. Zhao), whming@tju.edu.cn (H. Wu).

https://doi.org/10.1016/j.neunet.2024.106240
Received 4 October 2023; Received in revised form 11 March 2024; Accepted 13 March 2024
Available online 19 March 2024
0893-6080/© 2024 Elsevier Ltd. All rights reserved.

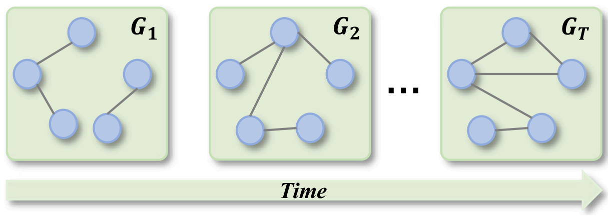

Fig. 1. An illustration of a dynamic network.

approaches intend to utilize the temporal information in processes that vary with time. As illustrated in Fig. 1, a dynamic network can be depicted as a sequence of network snapshots $\{G_t\}_{t=1}^{T}$, where each snapshot G_t represents the state of the network at a particular point in time. The objective of dynamic network representation learning is the estimation or generation of the present network snapshot G_T, based on the snapshots available $\{G_t\}_{t=1}^{T-1}$. Existing dynamic network representation methods which capture the evolving characteristics of dynamic networks can be broadly divided into two primary categories: random walk-based methods (Du, Wang, Song, Lu, & Wang, 2018; Qiu et al., 2020; Wang, Chang, Liu, Leskovec, & Li, 2021) and Graph Neural Network (GNN)-based methods (Manessi, Rozza, & Manzo, 2020; Pareja et al., 2020; Sankar, Wu, Gou, Zhang, & Yang, 2020). Random walk-based methodologies generate node embeddings by sampling random walks in a sequential manner, and then use the normalized inner products of corresponding node embeddings to model transition probabilities. On the other hand, GNN-based methods employ GNN to encode topological or attribute information and Recurrent Neural Network (RNN) or Transformer networks to capture temporal patterns.

Despite these methods achieving success with dynamic networks, a majority concentrate on the local topology or attribute characteristics and learn network embedding following the time sequence. This scenario poses two significant challenges to dynamic network representation learning. (i) The local topology, attribute, and temporal characteristics become less significant as the scale of the network grows, and the estimated or generative objective function of existing techniques tends to rely overly on these local characteristics. Overreliance on local characteristics could lead to representation models that are **non-robust**. (ii) Techniques that consider $\{G_t\}_{t=1}^{T}$ as a sequence and process $\{G_t\}_{t=1}^{T}$ sequentially could result in the **accumulation of errors**. Furthermore, most methods only take into account the difference between adjacent snapshots, ignoring the distinction between snapshots with larger time intervals, and therefore leading to information loss in modeling the network evolutionary process. To overcome these challenges, a proposed solution is the employment of contrastive learning to obtain network representations in dynamic networks. Contrastive Learning (CL) (Chen, Kornblith, Norouzi, & Hinton, 2020; He, Fan, Wu, Xie, & Girshick, 2020; van den Oord, Li, & Vinyals, 2018) has risen as an effective approach for representation learning, particularly in computer vision (Chen et al., 2020). Previous work has successfully applied contrastive learning to static network representation learning, where the purpose is to maximize Mutual Information (MI) between nodes and the network (Hassani & Ahmadi, 2020; Velickovic et al., 2019) or between node representations (Peng et al., 2020). This strategy aids in retaining macro-level network information, lessening the dependency on local topology or attributes and reducing non-robustness as the network scale expands. In addition, developing effective contrast strategies between nonadjacent snapshots can help mitigate error accumulation over time. Nonetheless, extending contrastive learning to dynamic networks remains a challenge due to the necessity for an appropriate contrastive strategy that takes into account network structure, attributes, and temporal evolution. The static network contrastive representation learning methods have achieved state-of-the-art results in node and network classification tasks, making the extension to dynamic networks a compelling subject. The contrastive strategy needs an effective capturing of the network's temporal evolution along with its structure and attributes.

To capture both the network characteristics and network evolution characteristics, we propose a novel **D**ynamic **N**etwork **C**ontrastive representation **L**earning (DNCL) model that comprises an inter-snapshots view and an intra-snapshots contrastive view. The intra-snapshots contrastive view allows DNCL-learned representations to retain the topological and attribute characteristics of each snapshot within the dynamic network, while the inter-snapshots view facilitates DNCL-learned representations to maintain the temporal characteristics spanning across various snapshots during network evolution, in addition to capturing non-sequential characteristics. More specifically, the DNCL model initiates by producing two correlated views of the dynamic network through the application of stochastic corruption onto the input snapshot sequence. The model is then trained using a joint contrastive loss, with the aim of maximizing agreement among node embeddings within the same views across different snapshots (inter-snapshot), as well as between the two views of the same snapshot (intra-snapshot). Consequently, the DNCL model learns representations that proficiently capture both the local characteristics within snapshots and the temporal patterns across snapshots. Extensive experiments conducted on real-world datasets evaluate the efficacy of the proposed DNCL model. The outcomes demonstrate superiority of our model over existing state-of-the-art models in three commonly executed tasks, which include link prediction, node classification, and node clustering. These results underscore the effectiveness of the DNCL model in learning comprehensive representations for dynamic networks, amplifying its potential to augment various network analysis tasks.

The main contributions of this paper are as follows:

- We propose a novel approach to dynamic network representation learning by applying contrastive learning. Our proposed Dynamic Network Contrastive representation Learning (DNCL) model effectively contains the temporal and topological attributes of dynamic networks, providing a promising trajectory for the accuracy enhancement in network analysis tasks.
- We introduce a dynamic contrastive aim that encapsulates the topology, attributes, and temporal characteristics of dynamic networks. This objective comprises both inter-snapshot contrast and intra-snapshot contrast, serving as vital components in our DNCL model. Our proposed method empowers effective representation learning in dynamic networks, augmenting their analysis prowess.
- We conduct comprehensive experiments on tasks such as link prediction, node classification, and clustering to evaluate the performance of our DNCL model. The outcomes validate our approach's effectiveness in dynamic network representation learning by outperforming state-of-the-art methods in these tasks.

The remainder of the paper is delineated as follows. Section 2 offers an extensive review of pertinent literature and historic work in the field. Section 4 explicates our proposed DNCL model for dynamic network representation learning in a detailed manner. The experimental setup, evaluation metrics, utilized datasets, and the performance comparison of our model with baselines are presented in Sections 5 and 6. Lastly, Section 7 concludes the paper, summarizing vital findings and contributions, and simultaneously discussing potential avenues for future research.

2. Related work

2.1. Dynamic network representation learning

Networks in practical applications, such as social networks and transportation networks, often evolve continuously, exhibiting dynamic behaviors. Traditional static network representation learning methodologies (Hamilton, Ying, & Leskovec, 2017; Jiao, Guo, et al., 2022;

Jiao et al., 2023; Kipf & Welling, 2017b; Veličković et al., 2018) have overlooked this evolution, resulting in their ineffectiveness on dynamic networks. To address this issue, dynamic network representation learning has gained considerable attention, leading to the proposal of various dynamic network representation learning methodologies (Kazemi et al., 2020; Liu, Huang, Yu, & Dong, 2021; Ma, Guo, Ren, Tang, & Yin, 2020).

The primary aim of most dynamic network representation learning methodologies is to approximate or generate observed network characteristics. For instance, NetWalk (Yu et al., 2018) implements a specialized autoencoder model to learn vector representations for nodes in a dynamic network. It aspires to reduce the pairwise distance among nodes in each random walk, efficaciously capturing the structural information of the network. Furthermore, it integrates a dynamic clustering algorithm to detect network deviations. VGRNN (Hajiramezanali et al., 2019) maps each node in the network to a random vector in the latent space, representing an efficient dynamic network representation learning method that superiorly captures the potential variabilities in dynamic networks. Using generative adversarial networks (GAN) and recurrent networks, DynGAN (Maheshwari, Goyal, Hanawal, & Ramakrishnan, 2019) is capable of capturing both the temporal and structural information of dynamic networks. Employing GANs, it is capable of generating realistic dynamic network snapshots that preserve crucial network characteristics. DySAT (Sankar et al., 2020) generates dynamic node representations by applying joint self-attention mechanisms that operate on two dimensions: structural neighborhoods and temporal dynamics. It uses multiple attention heads in both structural and temporal attention layers to enable joint attention across different latent subspaces, thereby capturing variations in network structure. EvolveGCN (Pareja et al., 2020) employs RNN to enact temporal evolution of the graph model, embodying a model-focused adaptation strategy, diverging from conventional node-centric methodologies, and introducing greater versatility in input assimilation. The HNIP embedding model (Qiu et al., 2020) integrates a time exponential decay model to estimate the temporal proximity between nodes, exploring network historical information; it aspires to preserve high-order nonlinear information in dynamic networks. DGCN (Gao, Zhu, Zhang, Wang, & Li, 2022) uses RNN for the temporal adaptation of GCN's weight parameters, thereby effectively capturing global structural information across the dynamic graph's time steps. To counteract the reduced influence of directed neighbors, Dice similarity is invoked, offering a robust solution. HyperDNE (Huang et al., 2023) employs a dual-output sequential hypergraph, enabling the exploration of nodes and edges' collective characteristics and with a line graph neural network, assisting in the consolidation of social influence arising from social convergence degrees. Moreover, Pham et al. (2022) and Chen et al. (2021) introduce a method specifically tailored for link prediction, accounting for local structures in dynamic networks.

2.2. Contrastive learning on static network

Drawing inspiration from CL methods employed in image representation learning, certain CL methods for static network representation learning have been proposed (You et al., 2020; Zhu et al., 2021a). Deep Graph InfoMax (DGI) (Velickovic et al., 2019) fuses the potency of GNN with InfoMax-centric methods. It enhances the network by rearranging the nodes' features, thereby formulating a specific contrastive-based objective to amplify the mutual information (MI) between the node embeddings and a global summary embedding. Diverging from the utilization of the readout or corruption function to augment the network explicitly, GMI (Peng et al., 2020) extends DGI via the introduction of two contrastive objectives. These directly assess the MI between inputs and their respective representational nodes and edges. MVGRL (Hassani & Ahmadi, 2020), on the other hand, proficiently retains the global information of the input network by propagating the adjacency matrix from the original network. It constructs network views via uniformly sampling subgraphs and learns both node and graph-level representations by contrasting node representations with augmented network summary representations. Both DGI and MVGRL augment the original network in a fixed pattern, which might not be efficient when the attributes or the topology of nodes in the network are sparse. To overcome this, GCA (Zhu et al., 2021b) creates two network views through an adaptive network augmentation based on topology and attribute information, using contrastive-based loss to maximize the MI of node embeddings learned from the two augmented views.

Table 1
The qualitative comparison of the current literature.

Method	Data type of graph	Contrastive objective	Task level
DGI	Static	Node–global	Node
MVGRL	Static	Node–global	Node/Graph
GCA	Static	Node–node	Node
VGRNN	Dynamic	–	Node/Link
DySAT	Dynamic	–	Node/Link
E-GCN	Dynamic	–	Node/Link
HNIP	Dynamic	–	Node/Link
DGCN	Dynamic	–	Node/Link
HyperDNE	Dynamic	–	Node/Link
DNCL	Dynamic	Node–node	Node/Link

Table 1 identifies and juxtaposes crucial elements of related works with our DNCL model, including the data type of the graph, contrastive objective, and task-level properties. While some studies have explored the potential of contrastive learning for graphs, the majority concentrate on static graphs. Contrarily, our proposed method alleviates the challenges by implementing an intra-snapshots contrastive view and an inter-snapshots view. The former preserves the topological and attribute characteristics of each snapshot of the dynamic network, and the latter retains the temporal characteristics across snapshots during network evolution, also capturing non-sequence characteristics.

3. Preliminaries

In this segment, we furnish the formal characterization of the dynamic network and dynamic network representation learning. A summarization of major notations impacting this paper is enclosed in Supplementary Materials (Appendix. A).

3.1. Problem definition

A dynamic network is described as a network wherein the interconnections between nodes are not static, rather they evolve with time. It is predominantly demonstrated as a sequence of network snapshots procured at regular time intervals. In the scope of this study, we perceive any alterations in the network as changes in the edges, inclusive of scenarios when nodes are annexed or eliminated. Specifically, when nodes are freshly inducted or discarded, we regard them as isolated nodes void of interactions with other nodes. Consequently, for the entirety of this paper, we operate under the assumption of a consistent node quantity across all network snapshots.

Definition 1 (*Dynamic Network*)**.** A dynamic network, denoted as $\mathcal{G}$, can be represented as a sequence of network snapshots: $\mathcal{G} = (\mathcal{V}, \mathcal{E}) = \{G^1, G^2, \dots, G^T\}$. Here, $G^t = (\mathcal{V}, E^t)$ represents the network snapshot at time step t of $\mathcal{G}$, where t ranges from 1 to T. The set of all nodes in $\mathcal{G}$ is denoted as $\mathcal{V} = \{v_1, v_2, \dots, v_N\}$, where N represents the total number of nodes. The set of all links in $\mathcal{G}$ is given by $\mathcal{E} = \bigcup_{t=1}^{T} E^t$, where E^t represents the set of links at time step t.

The adjacency matrix of G^t is represented by $\mathbf{A}^t$, where each element $a^t_{i,j}$ represents the presence or absence of a link between nodes v_i and v_j at time t. If a link exists, then $a^t_{i,j} = a^t_{j,i} = 1$, otherwise $a^t_{i,j} = a^t_{j,i} = 0$. In addition, when considering node attributes, the attribute matrix of G^t is denoted by $\mathbf{X}^t$, which is an $N \times M$ matrix, with N representing

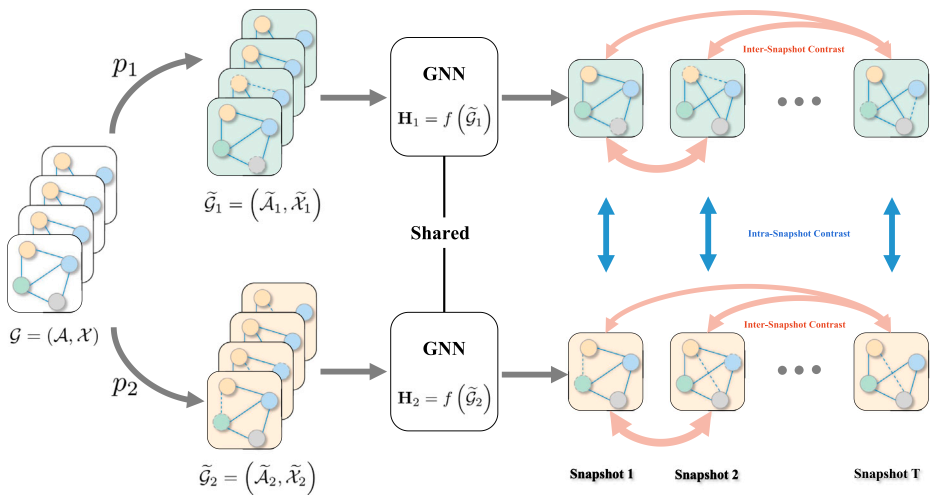
p_1
p_2
$\mathcal{G} = (\mathcal{A}, \mathcal{X})$
$\tilde{\mathcal{G}}_1 = (\tilde{\mathcal{A}}_1, \tilde{\mathcal{X}}_1)$
$\tilde{\mathcal{G}}_2 = (\tilde{\mathcal{A}}_2, \tilde{\mathcal{X}}_2)$
GNN
$\mathbf{H}_1 = f(\tilde{\mathcal{G}}_1)$
Shared
GNN
$\mathbf{H}_2 = f(\tilde{\mathcal{G}}_2)$
Inter-Snapshot Contrast
Intra-Snapshot Contrast
Inter-Snapshot Contrast
Snapshot 1
Snapshot 2
Snapshot T

Fig. 2. An overview of our proposed DNCL model. We first create two dynamic network views by augmenting the network structure and node attributes. Then, both views are input into a shared Graph Neural Network (GNN) encoder, which learns representations of the nodes. The model is trained using contrastive objectives, including the inter-snapshot contrast of node representations within the same views of different snapshots (red arrows) and the intra-snapshot contrast of node representations between two views of the same snapshots (blue arrows).

the number of nodes and M representing the dimension of the node attributes. It is important to note that the value of M is constant across time.

Definition 2 (*Dynamic Network Representation Learning*)**.** Given a dynamic network $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ with the corresponding adjacency matrix sequence $\mathcal{A} = \{\mathbf{A}^1, \mathbf{A}^2, \ldots, \mathbf{A}^T\}$ and attribute matrix sequence $\mathcal{X} = \{\mathbf{X}^1, \mathbf{X}^2, \ldots, \mathbf{X}^T\}$, the purpose of dynamic network representation learning is to learn a mapping function $f(\mathcal{A}, \mathcal{X}) \in \mathbb{R}^{N\times d}$, which receives the graph structure sequence and feature sequence as input, and generates node embedding in low dimensionality, where d is the embedding dimension. Let $\mathbf{H} = f(\mathcal{A}, \mathcal{X})$ denote the learned representations of nodes, where $\mathbf{h}_i$ is the embedding of node v_i.

4. Methodology

4.1. Dynamic contrastive learning model

The proposed DNCL model follows the common graph contrastive learning paradigm and seeks to maximize the agreement of representations between different views (Hassani & Ahmadi, 2020; Zhu et al., 2021b). As shown in Fig. 2, unlike static networks, which only produce different views for a single network, we augment the dynamic network sequence to produce two different view snapshot sequences. Then, we adopt a GNN encoder to obtain the embeddings of the two dynamic network views, respectively. After that, we employ a contrastive objective to achieve the following three purposes:

1. Enforce the encoded embedding of each node in the same view but different snapshots (inter-snapshot) to agree with each other to maintain the local topology characteristics during the network evolution, and the correlation not limited to adjacent snapshots;
2. Enforce the encoded embedding of each node in the different views but the same snapshot (intra-snapshot) to agree with each other to maintain the local topology characteristics at the same time;
3. Ensure that the encoded embedding of each node is distinct from the embeddings of other nodes in different views and snapshots, allowing for discrimination between nodes and preserving the unique characteristics of each view and snapshot.

In our temporal multi-view network, each node v_i has an embedding $\mathbf{h}^j_{i;1}$ in view 1 at snapshot j that serves as the anchor. The embedding $\mathbf{h}^j_{i;2}$ of the same node in view 2 at snapshot j serves as the positive sample, while the embeddings of other nodes in both views are considered negative samples. To model the temporal relationships between the two views, we adopt the InfoNCE objective from van den Oord et al. (2018) and define an intra-snapshot pairwise objective for each positive pair. It can be defined as follows:

$$\ell(\mathbf{h}^j_{i;1}, \mathbf{h}^j_{i;2}) = - \log \frac{e^{\theta\left(\mathbf{h}^j_{i;1}, \mathbf{h}^j_{i;2}\right)/\tau}}{\underbrace{e^{\theta\left(\mathbf{h}^j_{i;1}, \mathbf{h}^j_{i;2}\right)/\tau}}_{\text{positive pair}} + \underbrace{\sum_{q\neq i}\left[e^{\theta\left(\mathbf{h}^j_{i;1}, \mathbf{h}^j_{q;2}\right)/\tau} + e^{\theta\left(\mathbf{h}^j_{i;1}, \mathbf{h}^j_{q;1}\right)/\tau}\right]}_{\text{negative pairs}}}, \tag{1}$$

where τ is a temperature hyper-parameter. We define the critic function $\theta(\mathbf{h}_1, \mathbf{h}_2) = s(g(\mathbf{h}_1), g(\mathbf{h}_2))$, where $s(\cdot, \cdot)$ is the L2 normalized dot product similarity and $g(\cdot)$ is a nonlinear projection to strengthen the expressive power of the critic function. In DNCL, the projection function $g(\cdot)$ is implemented with a two-layer perception model.

Considering the information shared between snapshots at different time steps may decrease as the time interval in the dynamic network increases, we further add an exponential decay function in our contrast objective to simulate the quantity of information sharing decays. Similarly, for any given node v_i, we consider its embedding $\mathbf{h}^j_{i;1}$, which is generated in one particular view at snapshot j, as the anchor. Here, j ranges from 1 to $T-1$. The embedding $\mathbf{h}^k_{i;1}$ of the same node generated at a subsequent snapshot k within the same view, with k ranging from $j+1$ to T, serves as the positive sample. On the other hand, the embeddings of the remaining nodes are treated as negative samples. This way, we can establish the inter-snapshot pairwise objective for each positive pair, which can be defined as follows:

$$\ell(\mathbf{h}^j_{i;1}, \mathbf{h}^k_{i;1}) = - \log \frac{e^{(1-|k-j|\gamma)} \cdot e^{\theta\left(\mathbf{h}^j_{i;1}, \mathbf{h}^k_{i;1}\right)/\tau}}{e^{\theta\left(\mathbf{h}^j_{i;1}, \mathbf{h}^k_{i;1}\right)/\tau} + \sum_{q\neq i}\left[e^{\theta\left(\mathbf{h}^j_{i;1}, \mathbf{h}^k_{q;1}\right)/\tau} + e^{\theta\left(\mathbf{h}^j_{i;1}, \mathbf{h}^j_{q;1}\right)/\tau}\right]}, \tag{2}$$

where γ is the temporal decay constant.

When the embeddings of nodes in two different views are used as anchors, $\ell\left(\mathbf{h}_{i;1}, \mathbf{h}_{i;2}\right)$ and $\ell\left(\mathbf{h}_{i;2}, \mathbf{h}_{i;1}\right)$ are defined similarly since two views are symmetrical. The contrastive objective to be maximized is defined as the average over all positive pairs. Therefore, the intra-snapshot loss of the same snapshot between two views can be formally defined as:

$$\mathcal{J}_{intra} = \sum_{j=1}^{T} \frac{1}{2N} \sum_{i=1}^{N} \left[\ell\left(\mathbf{h}_{i;1}^{j}, \mathbf{h}_{i;2}^{j}\right) + \ell\left(\mathbf{h}_{i;2}^{j}, \mathbf{h}_{i;1}^{j}\right) \right]. \tag{3}$$

The inter-snapshot loss of the same view but different snapshots can be formally defined as:

$$\mathcal{J}_{inter} = \sum_{j=1}^{T-1} \sum_{k=j+1}^{T} \frac{1}{2N} \sum_{i=1}^{N} \left[\ell(\mathbf{h}_{i;1}^{j}, \mathbf{h}_{i;2}^{k}) + \ell(\mathbf{h}_{i;2}^{k}, \mathbf{h}_{i;1}^{j}) \right]. \tag{4}$$

Therefore, the overall dynamic network contrastive loss is defined as:

$$\mathcal{J} = \alpha \cdot \left(\mathcal{J}_{inter} + \mathcal{J}'_{inter}\right) + \beta \mathcal{J}_{intra}, \tag{5}$$

where α and β are trade-off parameters used to balance the losses of inter-view and intra-view. $\mathcal{J}$inter and $\mathcal{J}$inter$'$ represent the inter-snapshot loss in two views, respectively. Without loss of generality, we set $\alpha = 0.5$ and $\beta = 1$ in our experiments.

4.2. Dynamic network augmentation

Visual representation learning has demonstrated that contrasting consistent and inconsistent views of images enhances encoder representations (Bachman, Hjelm, & Buchwalter, 2019). However, defining views on networks is a nontrivial task as networks lack standard augmentation methods, such as image cropping, rotation, and color distortion. Consequently, we consider two common augmentation approaches for networks: (1) structure-based augmentations that operate on network structure by removing edges, and (2) attribute-based augmentations that modify initial node attributes, such as masking or adding Gaussian noise. We create two distinct views of the snapshot sequence by augmenting the network snapshot sequence, which are denoted as $\widetilde{\mathcal{G}}_1 = p_1(\mathcal{G})$ and $\widetilde{\mathcal{G}}_2 = p_2(\mathcal{G})$. Specifically, to generate augmented views of the dynamic network in each snapshot, we randomly remove edges and mask node features. Higher probabilities are assigned to unimportant edges or features, while lower probabilities are given to important ones, as proposed in Zhu et al. (2021b). For the sake of model robustness, we use the same probability to remove edges or mask attributes.

To randomly remove edges in the original network snapshot, we define a masking matrix $\widetilde{\mathbf{R}} \in \{0,1\}^{N\times N}$. Each entry of the matrix is drawn from a Bernoulli distribution $\widetilde{\mathbf{R}}_{ij} \sim \mathcal{B}\left(1-p_r\right)$, where p_r is the probability of removing each edge. Given an input dynamic network $\mathcal{G}$ with its corresponding adjacency matrix sequence $\left\{\mathbf{A}^1, \mathbf{A}^2, \ldots, \mathbf{A}^T\right\}$, we compute the resulting adjacency matrix $\widetilde{\mathbf{A}}^t$ for time-step t as the Hadamard product of $\mathbf{A}^t$ and $\widetilde{\mathbf{R}}$, *i.e.*, $\widetilde{\mathbf{A}}^t = \mathbf{A}^t \circ \widetilde{\mathbf{R}}$.

Furthermore, to mask node features with zeros, we use the vector $\widetilde{\mathbf{m}} \in \{0,1\}^M$. Each dimension of the vector is drawn independently from a Bernoulli distribution with probability $1-p_m$, *i.e.*, $\widetilde{\mathbf{m}}_i \sim \mathcal{B}\left(1-p_m\right)$. Given an input dynamic network $\mathcal{G}$ with its corresponding attribute matrix sequence $\left\{\mathbf{X}^1, \mathbf{X}^2, \ldots, \mathbf{X}^T\right\}$, we compute the generated masked features $\widetilde{\mathbf{X}}^t$ of $\mathbf{X}^t$ as the vertically concatenated matrix of element-wise products between each node feature vector $\mathbf{x}_i^t$ and the masking vector $\widetilde{\mathbf{m}}$, which is calculated by:

$$\widetilde{\mathbf{X}}^t = \left[\mathbf{x}_1^t \circ \widetilde{\mathbf{m}}; \mathbf{x}_2^t \circ \widetilde{\mathbf{m}}; \cdots; \mathbf{x}_N^t \circ \widetilde{\mathbf{m}}\right]^\top, \tag{6}$$

where $[\cdot;\cdot]$ is the concatenation operator.

Our model produces two distinct views of the dynamic network $\mathcal{G}$ by simultaneously employing two methods. The generation of $\widetilde{\mathcal{G}}_1$ and $\widetilde{\mathcal{G}}_2$ is controlled by two hyperparameters, namely p_r and p_m. To ensure diversity in the generated contexts for the two views, we use two separate sets of hyperparameters, denoted by $\{p_{r,1}, p_{m,1}\}$ and $\{p_{r,2}, p_{m,2}\}$ for the augmentation process.

4.3. The GNN encoder

To extract the topology and temporal information of the dynamic network in two different views, we adopt a GNN encoder after corrupting the input snapshot sequence. The GNN encoder of DNCL consists of two parts: a structure layer and a temporal layer. The structure layer captures the nonlinear spatial or attribute features from the network snapshot at each time step, while the temporal layer captures the nonlinear temporal characteristics and evolving patterns. For the structure layer, we use a two-layer Graph Convolutional Network (GCN) (Kipf & Welling, 2017a), and for the temporal layer, we use a Gated Recurrent Unit (GRU) (Cho et al., 2014).

4.3.1. GCN-based structure layer

The GCN-based structure layer takes the feature matrix $\mathbf{X}$ as input and uses a localized first-order approximation to perform the spectral graph convolution operation based on the adjacency matrix $\mathbf{A}$. Formally, we define the operation of our GCN unit for snapshot t of the dynamic network as:

$$\mathbf{Z}^t = \mathrm{GCN}\left(\mathbf{X}^t, \mathbf{A}^t\right) = f\left(\hat{\mathbf{D}}^{-\frac{1}{2}}\hat{\mathbf{A}}\hat{\mathbf{D}}^{-\frac{1}{2}}\mathbf{X}^t\mathbf{W}_\mu\right), \tag{7}$$

where $\mathbf{X}^t$ and $\mathbf{A}^t$ are the attribute matrix and adjacency matrix of the network snapshot at time step t respectively. In the case of a network without node attributes, the matrix $\mathbf{X}^t$ is an identity matrix $\mathbf{I}$. The matrix $\hat{\mathbf{A}}$ is defined as $\mathbf{A}^t + \mathbf{I}$, where $\mathbf{I} \in \mathbb{R}^{N\times N}$ and $\mathbf{A}^t \in \mathbb{R}^{N\times N}$. The matrix $\hat{\mathbf{D}}$ represents the diagonal node degree matrix of $\hat{\mathbf{A}}$. The expression $\hat{\mathbf{D}}^{-\frac{1}{2}}\hat{\mathbf{A}}\hat{\mathbf{D}}^{-\frac{1}{2}}$ denotes the approximated graph convolution filter. The weight matrix is denoted as $\mathbf{W}_\mu$. The nonlinear activation function ReLU is represented as $f(\cdot)$. The output representation obtained by the GCN layer is denoted as $\mathbf{Z}^t$, where $\mathbf{Z}^t \in \mathbb{R}^{N\times d}$ and d is the embedding dimension. When the input consists of the adjacency matrix sequence $\left\{\mathbf{A}^1, \mathbf{A}^2, \ldots, \mathbf{A}^T\right\}$ and attribute matrix sequence $\left\{\mathbf{X}^1, \mathbf{X}^2, \ldots, \mathbf{X}^T\right\}$, the output is obtained as $\left\{\mathbf{Z}^1, \mathbf{Z}^2, \ldots, \mathbf{Z}^T\right\}$ through the GCN layer.

4.3.2. GRU-based temporal layer

As a type of RNN, the GRU is capable of handling long-term dependencies in sequence modeling problems. We use it to capture the temporal dependencies between different snapshots of dynamic networks. In the GRU layer, we take $\left\{z_i^1, z_i^2, \ldots, z_i^T\right\}$ as input, where $z^t i$ indicates the ith row of the tth snapshot embedding matrix $\mathbf{Z}^t$. We input the hidden state representation of a single GRU layer to merge the information of the former continuous T snapshots in the final embedding h_i^T. At time step t, the GRU computes the new hidden state as follows:

$$c_i^t = \sigma\left(z_i^t \mathbf{U}_c + h_i^{t-1}\mathbf{W}_c + b_c\right), \tag{8}$$

$$r_i^t = \sigma\left(z_i^t \mathbf{U}_r + h_i^{t-1}\mathbf{W}_r + b_r\right), \tag{9}$$

$$\widetilde{h}_i^t = \phi\left(z_i^t \mathbf{U}_h + \left(h_i^{t-1} \circ r_i^t\right)\mathbf{W}_h + b_h\right), \tag{10}$$

$$h_i^t = \left(1 - c_i^t\right) \circ h_i^{t-1} + c_i^t \circ \widetilde{h}_i^t, \tag{11}$$

where z_i^t is the input of GRU layer at time step t; c_i^t stands for the update gate, which determines how much past information is kept and how much new information is added; r_i^t stands for the reset gate, which is used to determine how much past information to forget; $\widetilde{h}_i^t$ is the candidate hidden state that uses the reset gate r_i^t to store the relevant information from the past; $\mathbf{U}_c$, $\mathbf{W}_c$, $\mathbf{U}_r$, $\mathbf{W}_r$, $\mathbf{U}_h$, $\mathbf{W}_h$ represent the weight matrices and b_c, b_r represent the biases; σ is a sigmoid function and ϕ is a hyperbolic tangent function. The whole input of GRU layer is $\left\{\mathbf{Z}^1, \mathbf{Z}^2, \ldots, \mathbf{Z}^T\right\}$, and the output is $\mathbf{H}^T \in \mathbb{R}^{N\times d}$.

The detailed training algorithm is summarized in Supplemental Materials (Appendix. B).

4.4. Complexity analysis

Assuming that all input and hidden features hold a dimension of d, T symbolizes the length of the network snapshot sequence, and N represents the total number of nodes, the time complexity of DNCL can be analyzed accordingly. The DNCL model incorporates several GCN and GRU modules, where each GCN module entails matrix multiplication and activation functions, resulting in a time complexity of $\mathcal{O}(TN^2d + Nd)$. In a similar vein, the GRU module involves linear and nonlinear transformations, along with cyclic calculations, yielding a time complexity of $\mathcal{O}(3T(Nd^2 + Nd))$. Consequently, the overall time complexity of DNCL is determined as follows:

$$\mathcal{O}(T(N^2d + Nd) + 3T(Nd^2 + Nd)). \tag{12}$$

Though the DNCL model employs mutual information, manages to cut down computational time through effective sampling strategies, and its primary overhead is attributed to the GCN and LSTM components. In contrast, both GCA and DGCN depict different computational overheads. GCA, being an adaptive contrast representation learning approach for static graph scenarios, leverages a priori knowledge to spur graph generation, yielding two views for concurrent operations. This does not substantially inflate its time overhead, which is primarily committed to the GCN operations, with a complexity of $\mathcal{O}(N^2d)$. DGCN, unlike GCA, computes Dice similarities across all nodes, and via GCN, the complexity is depicted as $\mathcal{O}(T(N^2C\log C + N^2d))$, wherein C symbolizes the average degree of a node. Consequently, DNCL displays superior experimental results compared to similar methods, without ramping up its computational time.

The model complexity bears a linear proportion with the network size and relies on the model structure. Therefore, when the network size (N) undergoes a substantial escalation, the complexity of the model does not rise proportionally. Whilst DNCL encapsulates a significant number of parameters, it can accomplish a test within seconds thanks to GPU acceleration.

4.5. Theorem and proof

We give theoretical justification behind DNCL from two aspects, *i.e.*, MI maximization and the defined loss.

Theorem 1. *For two random variables $\mathbf{H}_1, \mathbf{H}_2 \in \mathbb{R}^d$ being the embedding of one snapshot in the two views or two snapshots in one view, with their joint distribution denoted as $p(\mathbf{H}_1, \mathbf{H}_2)$, our objective $\mathcal{J}$ is a lower bound of MI between encoder input $\mathbf{X}$ and node representations in two network views or two network snapshots $\mathbf{H}_1, \mathbf{H}_2$. Formally,*

$$\mathcal{J} \le I(\mathbf{X}; \mathbf{H}_1, \mathbf{H}_2). \tag{13}$$

Proof. We first demonstrate the connection between $\mathcal{J}$ and the InfoNCE objective (van den Oord et al., 2018), which is defined as:

$$I_{\mathrm{NCE}}(\mathbf{H}_1; \mathbf{H}_2) \triangleq \mathbb{E}_{\Pi_i p(\mathbf{h}_{i;1}, \mathbf{h}_{i;2})}\left[\frac{1}{N}\sum_{i=1}^{N}\log\frac{e^{\theta(\mathbf{h}_{i;1}, \mathbf{h}_{i;2})}}{\frac{1}{N}\sum_{j=1}^{N} e^{\theta(\mathbf{h}_{i;1}, \mathbf{h}_{j;2})}}\right], \tag{14}$$

where $\theta(\mathbf{x}, \mathbf{y}) = s(g(\mathbf{x}), g(\mathbf{y}))$. We further define $\rho_r(\mathbf{h}_{i;1}) = \sum_{j\neq i}^{N} e^{\theta(\mathbf{h}_{i;1}, \mathbf{h}_{j;1})/\tau}$ and $\rho_c(\mathbf{h}_{i;1}) = \sum_{j=1}^{N} e^{\theta(\mathbf{h}_{i;1}, \mathbf{h}_{j;2})/\tau}$ for convenience of notation. $\rho_r(\mathbf{h}_{i;2})$ and $\rho_c(\mathbf{h}_{i;2})$ can be defined similarly. Then, the objective $\mathcal{J}$ could be rewritten as follows:

$$\mathcal{J} = \mathbb{E}_{\Pi_i p(\mathbf{h}_{i;1}, \mathbf{h}_{i;2})}\left[\frac{1}{N}\sum_{i=1}^{N}\log\frac{e^{\theta(\mathbf{h}_{i;1}, \mathbf{h}_{i;2})/\tau}}{\sqrt{\rho_c(\mathbf{h}_{i;1}) + \rho_r(\mathbf{h}_{i;1})}} \cdot \frac{1}{\sqrt{\rho_c(\mathbf{h}_{i;2}) + \rho_r(\mathbf{h}_{i;2})}}\right].$$

Replacing by the notation of ρ_c, the InfoNCE estimator I_{NCE} can be written as follows:

$$I_{\mathrm{NCE}}(\mathbf{H}_1, \mathbf{H}_2) = \mathbb{E}_{\Pi_i p(\mathbf{h}_{i;1}, \mathbf{h}_{i;2})}\left[\frac{1}{N}\sum_{i=1}^{N}\log\frac{e^{\theta(\mathbf{h}_{i;1}, \mathbf{h}_{i;2})/\tau}}{\rho_c(\mathbf{h}_{i;1})}\right].$$

Therefore, we obtain

$$\begin{aligned}2\mathcal{J} = {} & I_{\mathrm{NCE}}(\mathbf{H}_1, \mathbf{H}_2) - \mathbb{E}_{\Pi_i p(\mathbf{h}_{i;1}, \mathbf{h}_{i;2})}\left[\frac{1}{N}\sum_{i=1}^{N}\log\left(1 + \frac{\rho_r(\mathbf{h}_{i;1})}{\rho_c(\mathbf{h}_{i;1})}\right)\right] \\ & + I_{\mathrm{NCE}}(\mathbf{H}_2, \mathbf{H}_1) - \mathbb{E}_{\Pi_i p(\mathbf{h}_{i;1}, \mathbf{h}_{i;2})}\left[\frac{1}{N}\sum_{i=1}^{N}\log\left(1 + \frac{\rho_r(\mathbf{h}_{i;2})}{\rho_c(\mathbf{h}_{i;2})}\right)\right] \\ & \le I_{\mathrm{NCE}}(\mathbf{H}_1, \mathbf{H}_2) + I_{\mathrm{NCE}}(\mathbf{H}_2, \mathbf{H}_1).\end{aligned}$$

Referring to Poole, Ozair, van den Oord, Alemi, and Tucker (2019), the InfoNCE estimator is a lower bound of the true MI, *i.e.*,

$$I_{\mathrm{NCE}}(\mathbf{H}_1, \mathbf{H}_2) \le I(\mathbf{H}_1; \mathbf{H}_2).$$

So, we can get

$$2\mathcal{J} \le I(\mathbf{H}_1; \mathbf{H}_2) + I(\mathbf{H}_2; \mathbf{H}_1) = 2I(\mathbf{H}_1; \mathbf{H}_2),$$

finally, we get the following inequality

$$\mathcal{J} \le I(\mathbf{H}_1; \mathbf{H}_2). \tag{15}$$

Referring to Edwards (2008), which states that, for all random variables $\mathbf{X}, \mathbf{Y}, \mathbf{Z}$ satisfying the Markov relation $\mathbf{X} \to \mathbf{Y} \to \mathbf{Z}$, the inequality $I(\mathbf{X}; \mathbf{Z}) \le I(\mathbf{X}; \mathbf{Y})$ holds. We find that $\mathbf{X}, \mathbf{H}_1, \mathbf{H}_2$ satisfy the relation $\mathbf{H}_1 \leftarrow \mathbf{X} \to \mathbf{H}_2$. Since $\mathbf{H}_1$ and $\mathbf{H}_2$ are conditionally independent after observing $\mathbf{X}$, the relation is Markov equivalent to $\mathbf{H}_1 \to \mathbf{X} \to \mathbf{H}_2$, which leads to $I(\mathbf{H}_1; \mathbf{H}_2) \le I(\mathbf{H}_1; \mathbf{X})$. We further observe that the relation $\mathbf{X} \to (\mathbf{H}_1, \mathbf{H}_2) \to \mathbf{H}_1$ holds, therefore, it follows that $I(\mathbf{X}; \mathbf{H}_1) \le I(\mathbf{X}; \mathbf{H}_1, \mathbf{H}_2)$. Combining the two inequalities yields the required inequality as follows:

$$I(\mathbf{H}_1; \mathbf{H}_2) \le I(\mathbf{X}; \mathbf{H}_1, \mathbf{H}_2). \tag{16}$$

Following Eqs. (15) and (16), we finally get the inequality:

$$\mathcal{J} \le I(\mathbf{X}; \mathbf{H}_1, \mathbf{H}_2), \tag{17}$$

end of the proof. □

Theorem 2. *When we use the identity function as projection function g, measure nodes' similarity by simply taking the inner product, and assume that positive pairs are far closer than negative pairs, i.e., $\mathbf{h}_{i;1}^{\top}\mathbf{h}_{j;2} \ll \mathbf{h}_{i;1}^{\top}\mathbf{h}_{i;2}$ and $\mathbf{h}_{i;1}^{\top}\mathbf{h}_{j;1} \ll \mathbf{h}_{i;1}^{\top}\mathbf{h}_{i;2}$, minimizing the pairwise objective $\ell(\mathbf{h}_{i;1}, \mathbf{h}_{i;2})$ coincides with maximizing the triplet loss, as given in the sequel*

$$\begin{aligned}-\ell(\mathbf{h}_{i;1}, \mathbf{h}_{i;2}) \propto 4N\tau + \sum_{j\neq i}\Big[&\|\mathbf{h}_{i;1} - \mathbf{h}_{i;2}\|^2 - \left\|\mathbf{h}_{i;1} - \mathbf{h}_{j;2}\right\|^2 \\ &+ \|\mathbf{h}_{i;1} - \mathbf{h}_{i;2}\|^2 - \left\|\mathbf{h}_{i;1} - \mathbf{h}_{j;1}\right\|^2\Big].\end{aligned} \tag{18}$$

Proof. Based on these assumptions, we can rearrange the pairwise objective as follows:

$$\begin{aligned}-\ell(\mathbf{h}_{i;1}, \mathbf{h}_{i;2}) &= -\log\frac{e^{\left(\mathbf{h}_{i;1}^{\top}\mathbf{h}_{i;2}/\tau\right)}}{\sum_{k=1}^{N} e^{\left(\mathbf{h}_{i;1}^{\top}\mathbf{h}_{k;2}/\tau\right)} + \sum_{k\neq i}^{N} e^{\left(\mathbf{h}_{i;1}^{\top}\mathbf{h}_{k;1}/\tau\right)}} \\ &= \log\left(1 + \sum_{k\neq i}^{N} e^{\frac{\mathbf{h}_{i;1}^{\top}\mathbf{h}_{k;2} - \mathbf{h}_{i;1}^{\top}\mathbf{h}_{i;2}}{\tau}} + \sum_{k\neq i}^{N} e^{\frac{\mathbf{h}_{i;1}^{\top}\mathbf{h}_{k;1} - \mathbf{h}_{i;1}^{\top}\mathbf{h}_{i;2}}{\tau}}\right).\end{aligned} \tag{19}$$

By Taylor expansion of first order, we obtain:

$$\begin{aligned}-\ell(\mathbf{h}_{i;1}, \mathbf{h}_{i;2}) &\approx \sum_{k\neq i}^{N}\left(e^{\frac{\mathbf{h}_{i;1}^{\top}\mathbf{h}_{k;2} - \mathbf{h}_{i;1}^{\top}\mathbf{h}_{i;2}}{\tau}} + e^{\frac{\mathbf{h}_{i;1}^{\top}\mathbf{h}_{k;1} - \mathbf{h}_{i;1}^{\top}\mathbf{h}_{i;2}}{\tau}}\right) \\ &\approx 2 + \frac{1}{\tau}\sum_{k\neq i}^{N}\left(\mathbf{h}_{i;1}^{\top}\mathbf{h}_{k;2} - \mathbf{h}_{i;1}^{\top}\mathbf{h}_{i;2} + \mathbf{h}_{i;1}^{\top}\mathbf{h}_{k;1} - \mathbf{h}_{i;1}^{\top}\mathbf{h}_{i;2}\right)\end{aligned}$$

Table 2
Statistical Details of Datasets. Including the number of snapshots, nodes, edges, and clusters in each dataset.

Attribute	#Snapshot	#Node	#Edge	#Cluster
Enron	12	151	3,513	–
Cell	10	400	5,124	–
UCI	10	1,604	25,190	–
B-OTC	7	5,051	40,516	–
COLAB	10	315	943	–
Ellip1	6	1,089	6,269	–
Ellip2	6	2,047	12,768	–
Ellip3	7	3,519	27,942	–
Ellip4	6	4,328	32,148	–
KIT	12	195	14,236	25
SBM1	8	1,280	81,661	4
SBM2	8	2,560	325,215	5
SBM3	8	5,120	653,093	6

$$
\begin{aligned}
&= 2 - \frac{1}{2\tau} \sum_{k\neq i}^{N} \Big(\left\|\mathbf{h}_{i;1} - \mathbf{h}_{k;2}\right\|^2 - \left\|\mathbf{h}_{i;1} - \mathbf{h}_{i;2}\right\|^2 \\
&\qquad + \left\|\mathbf{h}_{i;1} - \mathbf{h}_{k;1}\right\|^2 - \left\|\mathbf{h}_{i;1} - \mathbf{h}_{i;2}\right\|^2 \Big) \\
&\propto 4\tau + \sum_{k\neq i}^{N} \Big(\left\|\mathbf{h}_{i;1} - \mathbf{h}_{i;2}\right\|^2 - \left\|\mathbf{h}_{i;1} - \mathbf{h}_{k;2}\right\|^2 \\
&\qquad + \left\|\mathbf{h}_{i;1} - \mathbf{h}_{i;2}\right\|^2 - \left\|\mathbf{h}_{i;1} - \mathbf{h}_{k;1}\right\|^2 \Big),
\end{aligned}
$$

which concludes the proof. □

5. Experimental setup

This section offers an overview of the experimental setup, which encompasses the datasets used, the employed comparison methods, the evaluation metrics used to measure task performance, and a detailed outline of the parameter settings utilized in the experiments.

5.1. Datasets

We conduct experiments on seven real-world dynamic networks and three synthetic dynamic networks. The datasets utilized in our study are characterized by edge sizes that vary in magnitude from one hundred to over one hundred thousand. These datasets are derived from a variety of domains, encompassing social, industrial, and synthetic fields. The deliberate selection of these diverse datasets facilitates a more comprehensive demonstration of the robustness and universal applicability of our proposed models. The detailed statistics are reported in Table 2 and give their detailed description in Supplemental Materials (Appendix. C).

5.2. Baseline methods

To evaluate the performance of DNCL, we compare it against six network representation learning methods. For static network representation learning, we independently learn embeddings of each snapshot and subsequently aggregate them by computing their average. They including GCN (Kipf & Welling, 2017a), GCA (Zhu et al., 2021b), VGRNN (Hajiramezanali et al., 2019), DySAT (Sankar et al., 2020), EvolveGCN(E-GCN) (Sankar et al., 2020), HNIP (Qiu et al., 2020), DGCN (Gao et al., 2022) and HyperDNE (Huang et al., 2023). We give their detailed introduction in Supplemental Materials (Appendix. D).

To ensure a fair comparison, we employ default parameter settings for all baseline methods. Additionally, we use the same embedding dimension settings as DNCL on each dataset. The input snapshot sequence length, denoted as T, is set to 3. However, it is worth noting that HNIP does not support attribute graphs, so we only compare its performance with that of other methods in the link prediction and clustering tasks.

5.3. Evaluation metrics and parameters settings

In our experiments, we use Average Precision (AP) and Area under the ROC Curve as metrics for link prediction tasks, F1 score as the metric for node classification tasks, and Normalized Mutual Information (NMI) as the metric for clustering tasks.

AP is a commonly used metric for binary classification problems that averages the precision over predicted links. It can be defined as follows:

$$
AP = \frac{\sum_k p@k \cdot \mathbb{I}\left\{E_{pred}(k) \in E_{gt}\right\}}{\left|\left\{k : E_{pred}(k) \in E_{gt}\right\}\right|}, \tag{20}
$$

where $p@k = \frac{\left|E_{pred}(k) \cap E_{gt}\right|}{k}$, which represents the fraction of correct predictions in the top k predictions and k is the ranking position of the prediction results, which is used to calculate the weighted sum of the prediction results of different ranking positions when calculating AP and calculate the corresponding weight according to whether the prediction results are consistent with the real category; E_{pred} and E_{gt} represent the predicted and ground truth links, respectively. A higher AP value indicates that the model can make accurate predictions for a larger proportion of links.

AUC is a metric that measures the model's ability to distinguish between true and false positive rates at various thresholds. It can be computed by considering u independent comparisons between the existing and nonexistent links, where u' times the existing link gets a higher score than the nonexistent link and u'' times they receive the same score. Then, the AUC is calculated as:

$$
AUC = \frac{u' + 0.5u''}{u}. \tag{21}
$$

To mitigate the impact of sparsity, we randomly sample the same number of existing edges for nonexistent edges before computing the AUC.

NMI is a metric commonly used to detect the effectiveness of network partitions. For two different divisions of A and B, the NMI is defined as follows:

$$
NMI = \frac{-2\sum_{i=1}^{C_A}\sum_{j=1}^{C_B} C_{ij} \cdot \log\left(\frac{C_{ij}\cdot N}{C_{i\cdot}\cdot C_{\cdot j}}\right)}{\sum_{i=1}^{C_A} C_{i\cdot} \cdot \log\left(\frac{C_{i\cdot}}{N}\right) + \sum_{j=1}^{C_B} C_{\cdot j} \cdot \log\left(\frac{C_{\cdot j}}{N}\right)}, \tag{22}
$$

where N represents the number of nodes, C denotes the confusion matrix, and C_{ij} represents the number of nodes belonging to community i in division A and also belonging to community j in division B. C_A and C_B represent the number of communities in divisions A and B, respectively. Furthermore, $C_{i\cdot}$ and $C_{\cdot j}$ represent the sums of the elements in the matrix C. The larger value of NMI means more similarity between A and B. Especially when $NMI = 1$, it indicates that A and B are the same division of the network.

For DNCL, in the parameter sensitivity section, denoted as Section 6.5, we illustrate the parameters applied in the DNCL model. The configuration is as follows: the node representation dimension is fixed at 64, the trade-off parameters α and β are set to 0.5 and 1 respectively, and the temperature parameter τ stands at 0.5. Additionally, for the generation of $\widetilde{\mathcal{G}}_1$ and $\widetilde{\mathcal{G}}_2$, the parameters $p_{r,1}, p_{m,1}$ and $p_{r,2}, p_{m,2}$ are set as 0,0 and 0.1, 0.1 accordingly. With these parameters, the model yields optimum results. Moreover, fluctuations in the performance of DNCL are minimal with variant parameter settings, thereby validating the robustness of our model.

6. Experimental results

We verified the effect of DNCL on three different tasks: link prediction, node classification and clustering respectively, and the experimental results will be introduced separately in this section.

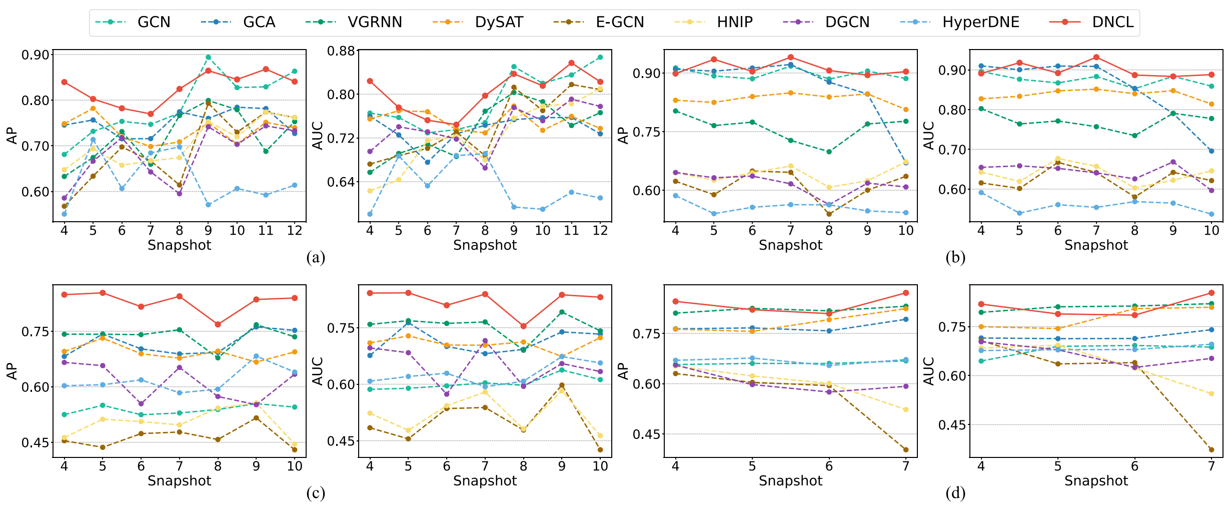

Fig. 3. Comparison of link prediction results based on the AP and AUC scores on four real-world networks. (a) Enron. (b) Cell. (c) UCI. (d) B-OTC.

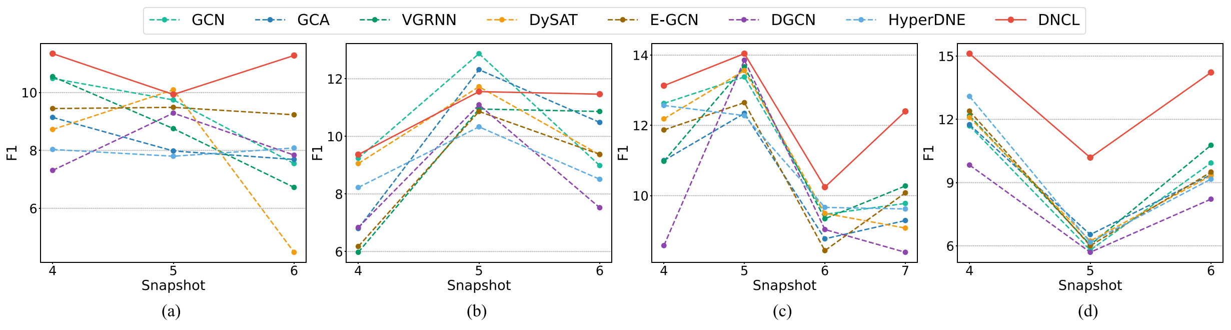

Fig. 4. Comparison of node classification results based on the F1 score on four real-world networks. (a) Ellip1. (b) Ellip2. (c) Ellip3. (d) Ellip4.

Table 3
Average AP and AUC scores (%) of link prediction on four real-world networks.

Method	Metric	Enron	Cell	UCI	B-OTC	COLAB
GCN	AP	78.88	86.31	53.81	66.16	73.18
	AUC	78.99	85.26	60.30	67.84	73.45
GCA	AP	75.11	89.76	71.75	77.01	79.88
	AUC	73.63	87.40	71.21	72.05	76.23
VGRNN	AP	72.03	75.92	73.71	82.10	88.77
	AUC	73.45	77.11	75.39	80.93	86.21
DySAT	AP	73.35	83.36	69.30	78.35	90.40
	AUC	75.18	83.73	70.77	77.69	87.25
E-GCN	AP	69.31	61.16	46.36	55.78	87.53
	AUC	74.31	62.42	50.21	58.89	83.88
HNIP	AP	70.53	64.03	50.31	59.99	67.18
	AUC	72.35	63.83	52.13	63.63	69.34
DGCN	AP	68.09	61.71	61.26	60.50	68.19
	AUC	73.83	64.24	65.02	66.50	72.04
HyperDNE	AP	62.62	55.69	61.81	66.82	57.53
	AUC	63.25	55.94	62.69	68.29	60.30
DNCL	AP	**82.68** (± 0.12)	**91.27** (± 0.33)	**83.02** (± 0.11)	**82.32** (± 0.22)	**90.74** (± 0.07)
	AUC	**80.33** (± 0.14)	**89.91** (± 0.43)	**82.32** (± 0.12)	**81.21** (± 0.23)	**87.72** (± 0.17)

6.1. Performance of link prediction

This task aims to predict whether any two nodes will interact with each other in the future. For example, predict whether there is an edge between any two nodes in the fourth snapshot based on the information of the first three snapshots. We employ the garnered nodes to signify the connection probability between two nodes via a layer of multi-layer perceptron (MLP) and sigmoid functions. We experiment on the Enron, Cell, UCI, B-OTC and COLAB networks. We use AP and AUC scores to evaluate all the methods. For all snapshots in each dataset, we split them into multiple groups of input with the step size 1. We differ from DySAT and HyperDNE in dataset partitioning in that we input only the first three snapshots in link prediction to predict the subsequent snapshots, instead of using the first T snapshots for prediction and T+1 snapshots as in DySAT or HyperDNE. We take the average AP and AUC scores of all input groups for evaluation. The results are presented in Table 3 and Fig. 3. Table 3 shows the average value of AP and AUC on all snapshots, and Fig. 3 reveals the value of AP and AUC on each snapshot. We can see that the DNCL model outperforms the baseline methods. DySAT often achieves comparable performance to DNCL in different datasets. One possible reason is that DySAT can jointly model structural and temporal information by aggregators (*e.g.*, multi-head attention). Noticed that the results of the static graph representation learning model GCA are superior to some temporal methods, which also illustrates the effectiveness of contrastive learning from the side.

6.2. Performance of node classification

The aim of this task is to predict the node class in the future. The intuition is that a model that captures the evolution of dynamic

Table 4
Average F1 scores (%) of node classification on four datasets.

Method	Ellip1	Ellip2	Ellip3	Ellip4
GCN	9.26	10.37	11.31	9.14
GCA	8.27	9.87	10.35	9.22
VGRNN	8.68	9.26	11.07	9.66
DySAT	7.77	10.05	11.08	9.21
E-GCN	9.39	8.81	10.76	9.30
DGCN	8.14	8.48	9.96	7.91
HyperDNE	7.97	9.02	11.03	9.47
DNCL	**11.17** (± 2.13)	**10.83** (± 1.54)	**12.55** (± 1.41)	**13.19** (± 1.22)

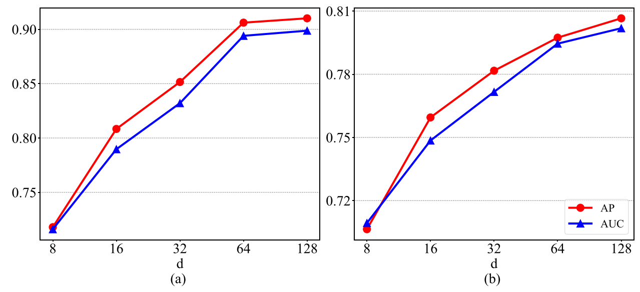

Fig. 5. Influence of embedding dimension for AP and AUC scores on two datasets. (a) Enron. (b) Cell.

networks should also be able to reflect changes in node classes within its learned embeddings. To this end, we train a classifier using node representations obtained from the sequence of node attributes. The dataset is split into training, validation and testing sets, with proportions of 50%, 20% and 30%, respectively. The micro average F1 score is used as the evaluation metric. The results of all methods on four Elliptic datasets are presented in Table 4 and Fig. 4, where Table 4 shows the average value of F1 scores on all snapshots, Fig. 4 reveals the value of F1 score on each snapshot. The results indicate that the DNCL model significantly outperforms the baseline methods, particularly in Ellip4, which is the largest and noisiest of the four networks. These findings suggest that our model is more robust than those that estimate or generate ground-truth network features in challenging environments.

6.3. Performance of clustering

The aim of this task is to predict the node cluster label in the future, *i.e.*, exploring changes in clusters in dynamic networks. We experiment on the KIT and three synthetic networks. Once we have learned node representations, we apply K-Means to cluster all nodes. The NMI score is used as the evaluation metric. From Table 5, we can see that DNCL still outperforms all baseline methods, especially on the dataset SBM3 with 5,120 nodes. The random-walk-based HNIP is competitive with GCN-based models (*e.g.*, E-GCN) in clustering tasks, which means that only considering neighbor information is not enough for node representation learning.

6.4. Ablation study

To verify the validity of our proposed contrast between different views in DNCL, *i.e.*, inter-snapshot loss and intra-snapshot loss, we conduct an ablation study by independently removing the different items in the overall contrastive loss as shown in Eq. (5). DNCL-1 refers to the model without the inter-snapshot loss (*i.e.*, $\mathcal{J}_{inter}$) in the topology perturbation view, DNCL-2 refers to the model without the inter-snapshot loss (*i.e.*, $\mathcal{J}'_{inter}$) in the node feature perturbation view, and DNCL-3 refers to the model without the intra-snapshot loss (*i.e.*, $\mathcal{J}_{intra}$).

The experimental results are presented in Table 6, which clearly demonstrate the effectiveness of both inter-snapshot and intra-snapshot

Table 5
Average NMI scores (%) of clustering on four datasets.

Method	KIT	SBM1	SBM2	SBM3
GCN	63.05	11.69	35.90	25.68
GCA	71.06	9.68	30.57	37.99
VGRNN	55.60	10.72	36.58	42.20
DySAT	60.50	11.33	41.12	41.07
E-GCN	71.76	12.54	37.12	41.65
HNIP	68.82	13.12	36.17	35.90
DGCN	55.26	10.19	37.27	40.13
HyperDNE	55.93	13.43	38.52	42.04
DNCL	**76.01** (± 2.17)	**17.10** (± 1.32)	**42.12** (± 1.11)	**55.93** (± 1.73)

Table 6
Ablation study.

		DNCL-1	DNCL-2	DNCL-3	DNCL
AP	Enron	80.61	80.90	83.02	**83.63**
	Cell	88.35	88.72	92.67	**91.17**
	UCI	79.00	78.60	80.04	**82.97**
	B-OTC	72.90	73.10	80.34	**83.42**
F1	Ellip1	9.39	7.86	8.47	**11.07**
	Ellip2	8.97	9.04	8.32	**10.80**
	Ellip3	10.71	11.04	11.31	**12.45**
	Ellip4	8.71	8.89	8.85	**13.18**
NMI	KIT	73.20	73.85	75.81	**75.91**
	SBM1	11.30	10.54	16.38	**17.01**
	SBM2	35.28	33.56	37.76	**42.03**
	SBM3	46.77	42.28	38.58	**55.82**

contrast in improving the performance of the proposed DNCL model on all datasets. Specifically, the ablation study shows that removing inter-snapshot contrast in the view of topology perturbation has a greater impact on link prediction and node classification tasks, while removing intra-snapshot contrast has a greater impact on clustering tasks. This observation suggests that network topology information plays a more important role in the learning of node representations than node attributes. These results provide further evidence of the validity of the proposed contrastive learning framework in DNCL, highlighting the importance of leveraging both inter-snapshot and intra-snapshot contrast in dynamic network representation learning.

6.5. Parameter sensitivity analysis

We conduct experiments for analyzing the sensitivity of several hyper-parameters in this subsection.

6.5.1. Embedding dimension

To analyze the influence of the hidden features learned by DNCL, we set the embedding dimensions d for each dataset as 8, 16, 32, 64, and 128, respectively. The results are presented in Fig. 5. As expected, the performance improves as the value of d increases. However, when the node embedding dimension exceeds 64, the growth rate of the performance is limited. Considering the storage space consumption, we adopt $d = 64$ for all experiments in this paper.

6.5.2. Inter-snapshot and intra-snapshot contrast weights

In this experiment, we investigate the influence of the inter-snapshot contrast parameter α on model performance by fixing β to 1 and varying α from 0.0 to 1.0. The results are presented in Fig. 6. While the impact of α on link prediction and clustering tasks is not as significant as on node classification, increasing α consistently improves model performance across all tasks. Additionally, we explore the influence of the intra-snapshot contrast parameter β by fixing α to 0.5 and varying β from 0.0 to 1.0. The results are presented in Fig. 7. We observe that changes in β have a significant impact on all three tasks, suggesting that the comparison between different views within the same snapshot is crucial for nodes to learn more discriminative representations.

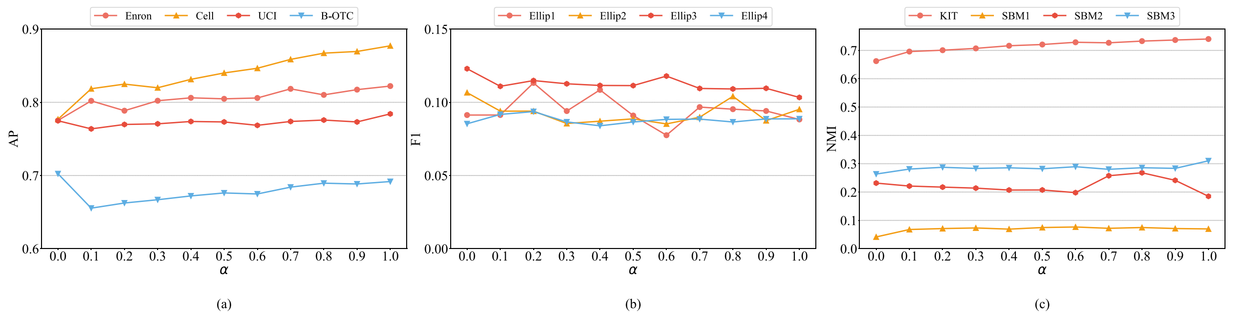

Fig. 6. Influence of α with $\beta = 1$ on different tasks, respectively. (a) Link prediction. (b) Node classification. (c) Clustering.

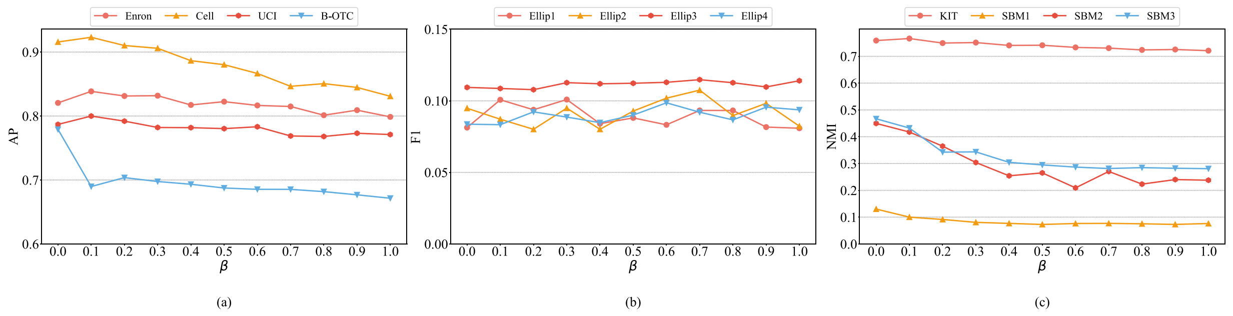

Fig. 7. Influence of β with $\alpha = 0.5$ on different tasks, respectively. (a) Link prediction. (b) Node classification. (c) Clustering.

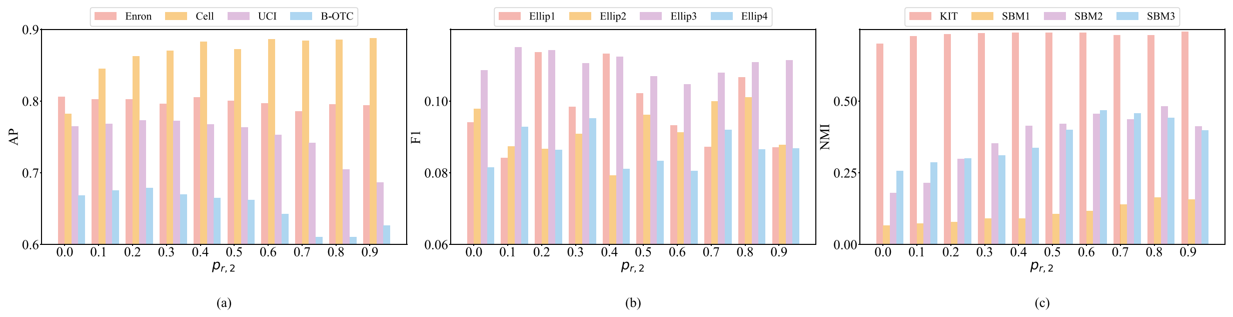

Fig. 8. Influence of $p_{m,2}$ with $p_{m,1} = 0$ on different tasks, respectively. (a) Link prediction. (b) Node classification. (c) Clustering.

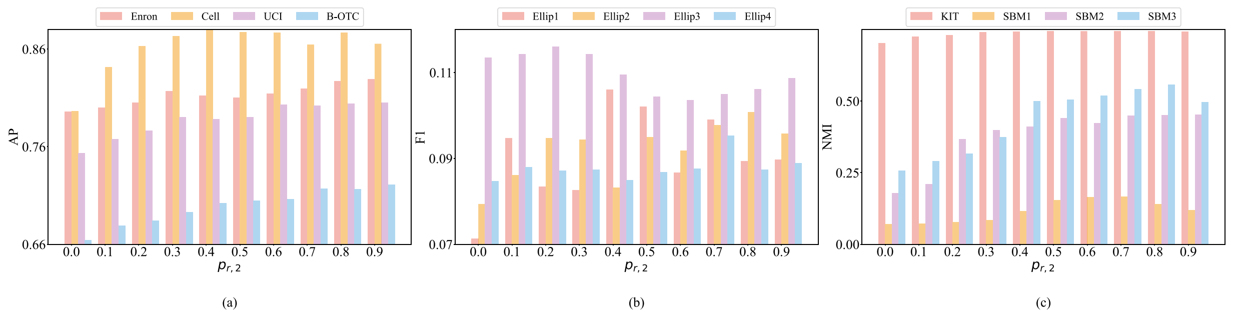

Fig. 9. Influence of $p_{r,2}$ with $p_{r,1} = 0$ on different tasks, respectively. (a) Link prediction, (b) Node classification and (c) Clustering.

6.5.3. The topology and node features mask weights

To explore the influence of the different view generation mechanisms, we fixed $p_{m,1}$ to 0 and then varied $p_{m,2}$ from 0.0 to 0.9, and fixed $p_{r,1}$ to 0 and then varied $p_{r,2}$ from 0.0 to 0.9, respectively. The results are presented in Figs. 8 and 9. The results show that masking more edges or node features improves the performance of clustering tasks more significantly.

7. Conclusion and future work

In spite of numerous dynamic network representation learning methodologies proposed in recent years, the majority are restricted to generative or estimated objectives, hyper-focusing on network details. Consequently, correlated models might lack robustness when networks scale and graph features become sparse. To rectify this, we proposed a contrastive learning method that maximizes mutual information to learn representations. Our method is tailored specifically for dynamic networks, which are typically represented as sequences of network snapshots. By incorporating both inter-snapshot and intra-snapshot contrast, our approach efficaciously captures both the structural (attribute) information and the temporal dimension of the network. We introduce the DNCL model grounded on this contrastive strategy, and our experimental results in various downstream tasks demonstrate its efficacy in overcoming the limitations of extant methods. Notwithstanding the strengths of our method, it is not devoid of limitations. Firstly, it targets discrete-time dynamic networks, potentially neglecting the fine-grained information contained within their continuous-time counterparts. Secondly, our method is limited to levering local neighborhood data, which may pose a barrier to accessing higher-order and densely semantic information.

In the future, we aim to broaden the application of DNCL to encompass continuous-time dynamic networks beyond discrete-time ones. This could be achieved by integrating techniques such as random point processes and temporal random walks. Additionally, while our model currently relies primarily on local neighborhood information, we acknowledge the limitations of intra-snapshot contrast and inter-snapshot contrast. We hence plan to explore the integration of higher-order structural information—such as motifs, communities, network homogeneity, and sub-graphs to bolster the diversity of our comparative views and obtain a more comprehensive understanding of the dataset. Furthermore, we intend to probe into methods such as pre-training and various levels of sampling to enhance the scalability and efficiency of the algorithm for larger dynamic networks.

CRediT authorship contribution statement

Pengfei Jiao: Funding acquisition, Methodology, Supervision, Writing – original draft, Writing – review & editing, Conceptualization. **Hongjiang Chen:** Methodology, Resources, Software, Writing – original draft, Formal analysis, Visualization. **Huijun Tang:** Data curation, Investigation, Methodology, Writing – review & editing. **Qing Bao:** Investigation, Visualization, Writing – review & editing. **Long Zhang:** Investigation, Writing – review & editing. **Huaming Wu:** Conceptualization, Funding acquisition, Investigation, Supervision, Writing – review & editing.

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.

Data availability

Data will be made available on request.

Acknowledgments

This work was supported in part by the National Natural Science Foundation of China under Grants 62372146 and 62071327, in part by the Fundamental Research Funds for the Provincial Universities of Zhejiang Grant GK229909299001-008, in part by the Zhejiang Provincial Natural Science Foundation of China under Grant LDT23F01012F01, in part by the National Social Science Fund of China under Grant 22FKSB042, and in part by the Hangzhou Artificial Intelligence Major Scientific and Technological Innovation Project under Grant 2022AIZD00161.

Appendix A. Supplementary data

Supplementary material related to this article can be found online at https://doi.org/10.1016/j.neunet.2024.106240.

References

Bachman, P., Hjelm, R. D., & Buchwalter, W. (2019). Learning representations by maximizing mutual information across views. In *Advances in neural information processing systems 32* (pp. 15509–15519).

Chen, H., Jiao, P., Tang, H., & Wu, H. (2023). Temporal graph representation learning with adaptive augmentation contrastive. In *Joint European conference on machine learning and knowledge discovery in databases* (pp. 683–699). Springer.

Chen, T., Kornblith, S., Norouzi, M., & Hinton, G. E. (2020). A simple framework for contrastive learning of visual representations. In *Proceedings of the 37th international conference on machine learning: vol. 119*, (pp. 1597–1607). PMLR.

Chen, J., Zhang, J., Xu, X., Fu, C., Zhang, D., Zhang, Q., et al. (2021). E-LSTM-D: A Deep Learning Framework for Dynamic Network Link Prediction. *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, [ISSN: 2168-2216] *51*(6), 3699–3712. http://dx.doi.org/10.1109/tsmc.2019.2932913.

Cho, K., van Merrienboer, B., Gülçehre, Ç., Bahdanau, D., Bougares, F., Schwenk, H., et al. (2014). Learning phrase representations using RNN encoder-decoder for statistical machine translation. In *Proceedings of the 2014 conference on empirical methods in natural language processing* (pp. 1724–1734). ACL, http://dx.doi.org/10.3115/v1/d14-1179.

Du, L., Wang, Y., Song, G., Lu, Z., & Wang, J. (2018). Dynamic network embedding: An extended approach for skip-gram based network embedding. In *IJCAI, vol. 2018* (pp. 2086–2092).

Edwards, S. (2008). Thomas M. Cover and Joy A. Thomas, elements of information theory (2nd ed.), John Wiley & Sons, Inc. (2006). *Information Processing &amp; Management*, *44*(1), 400–401. http://dx.doi.org/10.1016/j.ipm.2007.02.009.

Gao, C., Zhu, J., Zhang, F., Wang, Z., & Li, X. (2022). A novel representation learning for dynamic graphs based on graph convolutional networks. *IEEE Transactions on Cybernetics*.

Gehrke, J., Ginsparg, P., & Kleinberg, J. M. (2003). Overview of the 2003 KDD cup. *SIGKDD Explorations*, *5*(2), 149–151. http://dx.doi.org/10.1145/980972.980992.

Goyal, P., Chhetri, S. R., & Canedo, A. (2020). dyngraph2vec: Capturing network dynamics using dynamic graph representation learning. *Knowledge-Based Systems*, *187*, http://dx.doi.org/10.1016/j.knosys.2019.06.024.

Goyal, P., Kamra, N., He, X., & Liu, Y. (2018). DynGEM: Deep embedding method for dynamic graphs. CoRR abs/1805.11273 arXiv:1805.11273.

Grover, A., & Leskovec, J. (2016). Node2vec: Scalable feature learning for networks. In *Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining* (pp. 855–864). New York, NY, USA: ACM, http://dx.doi.org/10.1145/2939672.2939754.

Hajiramezanali, E., Hasanzadeh, A., Narayanan, K. R., Duffield, N., Zhou, M., & Qian, X. (2019). Variational graph recurrent neural networks. In *Advances in neural information processing systems 32* (pp. 10700–10710).

Hamilton, W. L., Ying, R., & Leskovec, J. (2017). Inductive representation learning on large graphs. In *Proceedings of the 31st international conference on neural information processing systems* (pp. 1025–1035). Red Hook, NY, USA: Curran Associates Inc., ISBN: 9781510860964.

Han, Y., Huang, G., Song, S., Yang, L., Wang, H., & Wang, Y. (2021). Dynamic neural networks: A survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *44*(11), 7436–7456.

Hassani, K., & Ahmadi, A. H. K. (2020). Contrastive multi-view representation learning on graphs. In *Proceedings of the 37th international conference on machine learning: vol. 119*, (pp. 4116–4126). PMLR.

He, K., Fan, H., Wu, Y., Xie, S., & Girshick, R. B. (2020). Momentum contrast for unsupervised visual representation learning. In *2020 IEEE/CVF conference on computer vision and pattern recognition* (pp. 9726–9735). IEEE, http://dx.doi.org/10.1109/CVPR42600.2020.00975.

Huang, J., Lu, T., Zhou, X., Cheng, B., Hu, Z., Yu, W., et al. (2023). HyperDNE: Enhanced hypergraph neural network for dynamic network embedding. *Neurocomputing*, *527*, 155–166. http://dx.doi.org/10.1016/j.neucom.2023.01.039.

Huang, Y., Shang, J., Lin, B. Y., Fu, L., & Wang, X. (2018). Dynamic detection of communities and their evolutions in temporal social networks. In S. A. McIlraith, & K. Q. Weinberger (Eds.), *Proceedings of the thirty-second AAAI conference on artificial intelligence* (pp. 8089–8090). AAAI Press.

Jiao, P., Guo, X., Pan, T., Zhang, W., Pei, Y., & Pan, L. (2022). A survey on role-oriented network embedding. *IEEE Transactions on Big Data*, *8*(4), 933–952. http://dx.doi.org/10.1109/TBDATA.2021.3131610.

Jiao, P., Li, T., Wu, H., Wang, C.-D., He, D., & Wang, W. (2022). HB-DSBM: Modeling the dynamic complex networks from community level to node level. *IEEE Transactions on Neural Networks and Learning Systems*.

Jiao, P., Li, T., Xie, Y., Wang, Y., Wang, W., He, D., et al. (2021). Generative evolutionary anomaly detection in dynamic networks. *IEEE Transactions on Knowledge and Data Engineering*, 1. http://dx.doi.org/10.1109/TKDE.2021.3129057.

Jiao, P., Tian, Q., Zhang, W., Guo, X., Jin, D., & Wu, H. (2023). Role discovery-guided network embedding based on autoencoder and attention mechanism. *IEEE Transactions on Cybernetics*, *53*(1), 365–378. http://dx.doi.org/10.1109/TCYB.2021.3094893.

Kazemi, S. M., Goel, R., Jain, K., Kobyzev, I., Sethi, A., Forsyth, P., et al. (2020). Representation learning for dynamic graphs: A survey. *Journal of Machine Learning Research*, *21*(70), 1–73.

Kipf, T. N., & Welling, M. (2016). Variational graph auto-encoders. In *NIPS Workshop on Bayesian Deep Learning*.

Kipf, T. N., & Welling, M. (2017a). Semi-supervised classification with graph convolutional networks. In *5th international conference on learning representations, ICLR 2017, Toulon, France, April 24-26, 2017, conference track proceedings*. OpenReview.net.

Kipf, T. N., & Welling, M. (2017b). Semi-supervised classification with graph convolutional networks. In *International conference on learning representations*.

Liu, Z., Huang, C., Yu, Y., & Dong, J. (2021). Motif-preserving dynamic attributed network embedding. In *Proceedings of the web conference 2021* (pp. 1629–1638).

Ma, Y., Guo, Z., Ren, Z., Tang, J., & Yin, D. (2020). Streaming graph neural networks. In *Proceedings of the 43rd international ACM SIGIR conference on research and development in information retrieval* (pp. 719–728).

Maheshwari, A., Goyal, A., Hanawal, M. K., & Ramakrishnan, G. (2019). Dyngan: Generative adversarial networks for dynamic network embedding. In *Graph representation learning workshop at neurIPS*.

Manessi, F., Rozza, A., & Manzo, M. (2020). Dynamic graph convolutional networks. *Pattern Recognition*, *97*, Article 107000.

van den Oord, A., Li, Y., & Vinyals, O. (2018). Representation learning with contrastive predictive coding. CoRR abs/1807.03748 arXiv:1807.03748.

Pareja, A., Domeniconi, G., Chen, J., Ma, T., Suzumura, T., Kanezashi, H., et al. (2020). EvolveGCN: Evolving graph convolutional networks for dynamic graphs. In *The thirty-fourth AAAI conference on artificial intelligence* (pp. 5363–5370). AAAI Press.

Peng, Z., Huang, W., Luo, M., Zheng, Q., Rong, Y., Xu, T., et al. (2020). Graph representation learning via graphical mutual information maximization. In *WWW '20: the web conference 2020* (pp. 259–270). ACM / IW3C2, http://dx.doi.org/10.1145/3366423.3380112.

Perozzi, B., Al-Rfou, R., & Skiena, S. (2014). DeepWalk: online learning of social representations. In *The 20th ACM SIGKDD international conference on knowledge discovery and data mining* (pp. 701–710). ACM, http://dx.doi.org/10.1145/2623330.2623732.

Pham, P., Nguyen, L. T. T., Nguyen, N. T., Pedrycz, W., Yun, U., & Vo, B. (2022). Comgcn: Community-driven graph convolutional network for link prediction in dynamic networks. *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, *52*(9), 5481–5493. http://dx.doi.org/10.1109/TSMC.2021.3130149.

Poole, B., Ozair, S., van den Oord, A., Alemi, A., & Tucker, G. (2019). On variational bounds of mutual information. In *Proceedings of the 36th international conference on machine learning*: *vol. 97*, (pp. 5171–5180). PMLR.

Qiu, Z., Hu, W., Wu, J., Liu, W., Du, B., & Jia, X. (2020). Temporal network embedding with high-order nonlinear information. In *The thirty-fourth AAAI conference on artificial intelligence* (pp. 5436–5443). AAAI Press.

Sankar, A., Wu, Y., Gou, L., Zhang, W., & Yang, H. (2020). DySAT: Deep neural representation learning on dynamic graphs via self-attention networks. In *WSDM '20: the thirteenth ACM international conference on web search and data mining* (pp. 519–527). ACM, http://dx.doi.org/10.1145/3336191.3371845.

Tang, J., Qu, M., Wang, M., Zhang, M., Yan, J., & Mei, Q. (2015). LINE: Large-scale information network embedding. In *Proceedings of the 24th international conference on world wide web* (pp. 1067–1077). ACM, http://dx.doi.org/10.1145/2736277.2741093.

Theocharidis, A., Van Dongen, S., Enright, A. J., & Freeman, T. C. (2009). Network visualization and analysis of gene expression data using BioLayout express 3D. *Nature protocols*, *4*(10), 1535.

Veličković, P., Cucurull, G., Casanova, A., Romero, A., Liò, P., & Bengio, Y. (2018). Graph attention networks. In *International conference on learning representations*.

Velickovic, P., Fedus, W., Hamilton, W. L., Liò, P., Bengio, Y., & Hjelm, R. D. (2019). Deep graph infomax. In *7th international conference on learning representations*.

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

You, Y., Chen, T., Sui, Y., Chen, T., Wang, Z., & Shen, Y. (2020). Graph contrastive learning with augmentations. *Advances in Neural Information Processing Systems*, *33*, 5812–5823.

Yu, W., Cheng, W., Aggarwal, C. C., Zhang, K., Chen, H., & Wang, W. (2018). NetWalk: A flexible deep embedding approach for anomaly detection in dynamic networks. In *Proceedings of the 24th ACM SIGKDD international conference on knowledge discovery & data mining* (pp. 2672–2681). ACM, http://dx.doi.org/10.1145/3219819.3220024.

Zhou, L., Yang, Y., Ren, X., Wu, F., & Zhuang, Y. (2018). Dynamic network embedding by modeling triadic closure process. In *Proceedings of the thirty-second AAAI conference on artificial intelligence* (pp. 571–578). AAAI Press.

Zhu, Y., Xu, Y., Yu, F., Liu, Q., Wu, S., & Wang, L. (2021a). Graph contrastive learning with adaptive augmentation. In *Proceedings of the web conference 2021* (pp. 2069–2080).

Zhu, Y., Xu, Y., Yu, F., Liu, Q., Wu, S., & Wang, L. (2021b). Graph contrastive learning with adaptive augmentation. In *WWW '21: the web conference 2021* (pp. 2069–2080). ACM / IW3C2, http://dx.doi.org/10.1145/3442381.3449802.