Information Sciences 667 (2024) 120499

Contents lists available at ScienceDirect

Information Sciences

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

A deep contrastive framework for unsupervised temporal link prediction in dynamic networks

Pengfei Jiao [a,c], Xinxun Zhang [a], Zehao Liu [a], Long Zhang [b], Huaming Wu [b], Mengzhou Gao [a,*], Tianpeng Li [b], Jian Wu [a,*]

[a] Hangzhou Dianzi University, Hangzhou, 310018, China
[b] Tianjin University, Tianjin, 300350, China
[c] Data Security Governance Zhejiang Engineering Research Center, Hangzhou, China

ARTICLE INFO

Keywords:
Dynamic network
Unsupervised link prediction
Contrastive learning
Deep learning
Multi-step prediction

ABSTRACT

In dynamic networks, temporal link prediction aims to predict the appearance and disappearance of links in future snapshots based on the network structure we have observed. It also plays a crucial role in network analysis and predicting the behavior of the dynamic system. However, most existing studies only focus on supervised temporal link prediction problems, i.e., taking part of the links in future snapshots as supervised information. The ones that can solve the unsupervised temporal link prediction problem are mainly based on matrix decomposition, which lack the capability to automatically extract nonlinear spatial and temporal features from dynamic networks. The most challenging part of this problem is to extract the inherent evolution of the patterns hidden in dynamic networks in unsupervised ways. Inspired by the application and achievement of contrastive learning in network representation learning, we propose a novel deep Contrastive framework for unsupervised Temporal Link Prediction (CTLP). Our framework is based on a deep encoder-decoder architecture, which can capture the nonlinear structure and temporal features automatically and can predict future links of subsequent snapshots of dynamic networks in an unsupervised manner. Besides, CTLP could handle the multi-step temporal link prediction problem of dynamic networks through attenuation modeling across the snapshots. Extensive experiments on temporal link prediction show that our CTLP framework significantly outperforms state-of-the-art unsupervised methods, and even outperforms the supervised methods in some cases.

1. Introduction

A variety of complex systems can be characterized as networks, such as social networks [1], co-authorship networks [2], and protein-protein interaction networks [3]. In these networks, each node signifies an entity, while links denote the dependency relationships between pairs of nodes, representing collaborations or interactions in the real systems. In reality, numerous problems in systems and engineering fields can be formulated as link predictions in complex networks [2–4]. For instance, link prediction in a social network can be employed to anticipate the creation of new friendships [5]. Similarly, the prediction of potential collaborations between authors can be achieved through co-authorship networks, and the identification of new connections among proteins is usu-

* Corresponding authors.
E-mail addresses: mzgao@hdu.edu.cn (M. Gao), hzieewujian@163.com (J. Wu).

https://doi.org/10.1016/j.ins.2024.120499
Received 18 January 2022; Received in revised form 17 March 2024; Accepted 17 March 2024

Available online 21 March 2024

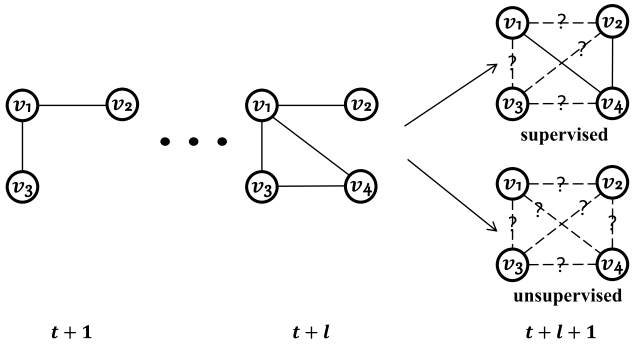

Fig. 1. The illustration of supervised and unsupervised temporal link prediction. In a dynamic network with $l+1$ consecutive snapshots, we expect to predict the links in the $(t+l+1)$-th snapshot. The supervised temporal link prediction takes two links in the $(t+l+1)$-th snapshot as supervised information, while for unsupervised temporal link prediction, the network topology information in the $t+l+1$-th snapshot is completely unknown when predicting.

ally uncovered using Protein-Protein Interaction (PPI) networks [6]. Generally, link prediction in complex networks aims to predict the appearance of latent or new links in the network based on the network part structure we have observed. It plays a crucial role in mining informative patterns of various complex networks, as it facilitates the construction and generation of the entire network [7].

Numerous methods have been introduced for link prediction in static networks, aiming to reveal latent or unobserved links in static networks [8–13]. However, considering the time-varying nature of real-world systems, it becomes evident that complex networks are dynamic and evolve over time. Specifically, these complex networks usually carry additional temporal information, involving the emergence and disappearance of nodes and edges as the network structure evolves across snapshots. For instance, in a social network, links will change with the behavior of social patterns of nodes [14], new links will be created (when people make new friends) and existing links may disappear. The links of a co-authorship network are also dynamically evolving since authors may frequently shift their research direction [15]. Compared with static networks, dynamic networks provide a more accurate depiction of complex systems due to their ability to capture evolving patterns. Consequently, link prediction in dynamic networks, or temporal link prediction, has garnered widespread attention as it offers an opportunity to analyze the dynamic evolution of the network structure.

Temporal link prediction aims to predict links in the snapshot at time $t+1$ utilizing the previous snapshots of the dynamic network and can be classified into two groups, supervised and unsupervised methods. In supervised temporal link prediction, a subset of links in the network at time $t+1$ is used as supervised information. Conversely, unsupervised temporal link prediction methods treat the network topology at time t+1 as unknown and predict links without relying on explicit supervised information. These two manners are illustrated in Fig. 1. Most existing research focuses on temporal link prediction problems in a supervised manner. The supervised temporal link prediction methods [16–21] can be broadly divided into two classes: direct temporal link prediction methods and network embedding-based temporal link prediction methods.

These two categories of methods are mainly based on deep learning. Deep learning models excel in automatically capturing the nonlinear structural and temporal features of networks at each timestamp in the process of dynamic network evolution. The direct temporal link prediction methods [17,22,18] treat the prediction of future links as the target of the model and utilize network reconstruction error as an objective function. The network embedding-based temporal link prediction methods [20,16,21] aim to learn the effective node embedding of the dynamic network, regarding link prediction as a downstream task. They treat temporal link prediction as a binary classification problem and utilize observed network structure at snapshot $t+1$ to train a prediction model. Unlike the current state of research on supervised methods, which has received widespread attention, only a few methods explore unsupervised temporal link prediction, and most of them are based on matrix decomposition [23]. However, matrix decomposition-based methods face limitations in automatically extracting effective spatial information and temporal dependencies of dynamic networks and fail to deal with the high complexity and nonlinearity of dynamic networks [7].

Motivated by the above facts, we propose a novel unsupervised temporal link prediction framework, which can automatically capture the complex nonlinear spatial features and temporal dependencies of dynamic networks, and perform link prediction in an unsupervised manner. To address the unsupervised challenge in temporal link prediction, we incorporate the contrastive loss into the temporal link prediction. Some recent works of static networks on contrastive learning [24–27] have made great achievements in the graph machine learning domain by maximizing mutual information (MI) between nodes and network representations. We maximize the structure information between future timestamps and past timestamps of dynamic networks to enable our model to make predictions of the future state of the network solely based on observed network historical data, eliminating the requirement for explicit supervision.

In addition, the majority of the recent temporal network representation learning algorithms [16,21,19] all typically employ a two-layer framework capable of extracting the nonlinear spatial–temporal features in dynamic networks. Specifically, the network structure information at each timestamp is extracted by a structure encoder, and then the network evolution pattern is obtained through another sequence model. The performance of these methods in temporal network prediction also proves the effectiveness of such a two-layer framework. Therefore, Our proposed **C**ontrastive unsupervised **T**emporal **L**ink **P**rediction (CTLP) framework adopts an encoder-decoder architecture with a two-layer setting. The encoder of CTLP consists of a structure encoder and a sequence model,

which allows the learned low-dimensional node representations of the dynamic network to capture more nonlinear spatial–temporal features. In addition, We utilize multiple identical decoders with different parameters to enable CTLP can solve the multi-step temporal link prediction problem, which means we can predict not only one future timestamp but several snapshots of the dynamic network. To simulate information sharing across different timestamps with increasing time spans, we implement an exponential decay function. Furthermore, considering that the sparsity of dynamic networks may severely reduce the performance of link prediction, we enhance the significance of existing links and enforce the model's focus on capturing the information on existing links more than nonexistent ones. Such a design allows CTLP can learn more effective embeddings for temporal link prediction. The main contributions of this paper can be summarized as follows:

1. We propose CTLP, a novel deep contrastive framework designed for temporal link prediction in an unsupervised manner. We innovatively apply the contrastive loss to temporal link prediction, and unsupervised predict future links by maximizing the structure information between different stages of network evolution.
2. Our proposed framework can adeptly handle the multi-step temporal link prediction problem in dynamic networks by employing multiple identical decoders with different parameters. Additionally, we incorporate an exponential decay function to simulate the potentially shared information of different network snapshots decay with time.
3. We conduct comprehensive experiments on six real-world dynamic network datasets, evaluating the superiority and excellent performance of our model in both single-step and multi-step temporal link prediction scenarios. The experiment results demonstrate that CTLP significantly outperforms state-of-the-art methods, validating its effectiveness in capturing complex temporal dynamics in dynamic networks.

The remainder of the paper is arranged as follows. We first introduce the related works on network embedding and temporal link prediction in Section 2. In Section 2, we describe the proposed CTLP framework for unsupervised temporal link prediction in detail. The details and results of the experiment and further analysis are illustrated in Section 4. At last, we summarize the paper and point out its future direction in Section 5.

2. Related work

We initially explore some existing network embedding methods designed for static networks (each snapshot of a dynamic network can be regarded as a static network). Subsequently, we delve into a discussion of works dedicated to both supervised and unsupervised temporal link prediction.

2.1. Network embedding

Each snapshot of a dynamic network is commonly denoted as a static network. Network embedding or representation learning on static networks aims to learn a projection function that can embed the network into a latent representation space. The obtained embedding vectors contain the network's essential topological features and statistical properties. According to recent surveys [28], current network embedding methods for static networks are typically divided into three categories, i.e., matrix factorization, random walk, and deep learning-based.

Matrix Decomposition-based Methods: This category of methods typically denotes a similarity matrix (the adjacency matrix or high-order similarity matrix) and learns the embedding of nodes by reconstructing it. GraRep [29] utilized the k-step similarity matrix with a global feature for each node, it can also be used for weighted networks. Higher-order proximity embedding [30] preserved the asymmetric transitivity by considering the high-order similarity in the network. Modularized nonnegative matrix factorization (M-NMF) [31] can preserve the fine-grained structure of first-order and second-order proximities while also preserving the mesoscopic structure of communities, demonstrating superior performance in community discovery. Space-Invariant Projection (SIP) [32] constructs new node embedding based on matrix factorization, enabling fast embedding of new nodes in dynamic networks. As a result, this category of network embedding methods mainly utilizes a specific similarity matrix or a specific combination of different similarity matrices to represent the network structure. However, it's important to note that these methods face challenges of high computational complexity and fail to deal with large-scale data.

Random Walk-based Methods: Inspired by the Skip-Gram model, random walks-based methods learn node embedding based on path similarity, which is suitable for partially observable networks or a large entirety. DeepWalk [33] was the pioneering attempt that utilized a collection of short truncated random walks as the corpus in Skip-Gram. Each node in the network was treated as having its own vocabulary and maximized the likelihood of co-occurrence in fixed-length random walks to the learned embedding vectors. Furthermore, Node2vec [34] was proposed to select the node to be converted, which utilized breadth-first search while incorporating depth-first search to capture local similarity information and global information of each node. RAW-GNN [35] introduced a novel graph aggregation mechanism by utilizing breadth-first random walk search to extract homophily information and depth-first search to capture heterophily information, which guarantees the validity of network embedding.

Deep Learning-based Methods: Deep learning-based methods have gained prominence in learning network representations and mining meaningful patterns in recent years. Structural Deep Network Embedding (SDNE) [36] aims to capture the highly nonlinear features of the network using a deep autoencoder framework with Laplace regularization that optimizes the proximity of different orders. Kipf and Welling [37] proposed a graph convolutional network (GCN), which successfully incorporates the attribute characteristics of nodes into network embedding via the convolution operation on the spectral domain. Furthermore, GCN has been

extended to achieve network reconstruction and link prediction in [10]. GraphSAGE [38] was an inductive approach by stacking the aggregation layers to learn the node embedding, in which the node features can be sampled and aggregated only with some local neighborhoods.

2.2. Supervised temporal link prediction

The supervised temporal link prediction methods treat the subset of future network links as supervised data and can be categorized into two groups: direct prediction methods and network embedding-based methods.

Direct Temporal Link Prediction Methods: These methods take the prediction of future links or future adjacency matrix reconstruction as the target task. To predict temporal links, E-LSTM-D [17] proposed the encoder-decoder framework with a stacked long short-term memory (LSTM) [39] to extract historical information. DynGEM [22] initialized with the parameters from the previous time step to generate stable embeddings. By introducing a heuristic method, DynGEM can automatically expand layers when the size of the ancient is inappropriate. However, it faces limitations in handling situations where the network structure changes dramatically. Subsequently, dyngraph2vec [40] utilized an architecture constructed from dense and recurrent layers to learn the evolutionary structure of dynamic graphs and predict future links. Besides, motivated by the impressive performance of Generative Adversarial Networks (GANs) in image generation, Yang et al. [7] incorporated the GAN framework into dynamic networks called NetworkGAN, it could be used for both unweighted and weighted dynamic networks that combining GCN, TMF, and LSTM.

Network Embedding-based Temporal Link Prediction Methods: The main purpose of these methods is to learn dense embedding vectors of nodes and treat temporal link prediction as a downstream task. By employing joint self-attention across temporal dependencies and structural neighborhoods, DySAT [19] can generate dynamic node representations. To capture multi-faceted structural variations in the graph, DySAT integrated multiple attention heads within both the structural and temporal attention layers to facilitate joint attention across various latent subspaces. NetWalk [41] was a customized autoencoder model that learns vector representations for nodes. In each random walk, it minimized the pairwise distance among nodes. By introducing a dynamic clustering algorithm, NetWalk can detect network deviations. VGRNN [20] uniquely mapped each node to a random vector in the latent space, which can better capture the potential variability in dynamic networks. DynGAN [42] integrated generative adversarial networks with recurrent networks for the purpose of capturing both temporal and structural information. Qiu et al. [43] proposed a high-order nonlinear information preserving embedding model (HNIP), which can explore network historical information with a time exponential decay model to quantify the temporal proximity between nodes. HTGN [16] transformed dynamic networks into hyperbolic space by integrating hyperbolic-gated recursive neural networks and hyperbolic graph neural networks. This combination enables the model to capture dynamic behaviors over time while implicitly maintaining hierarchical relationships.

2.3. Unsupervised temporal link prediction

Yu et al. [23] introduced the TMF model, utilizing the matrix factorization method to explicitly characterize the temporal structure of the network. In this approach, the adjacency matrix of each network snapshot is decomposed into the product of a constant matrix and a polynomial function matrix with time as the independent variable. Notably, the parameter matrix of the polynomial function is shared across all time slices. So this method can predict the future links of dynamic networks in an unsupervised manner.

3. Methodology

We first formalize the problem definition of temporal link prediction, and then show the detailed process of our proposed CTLP framework. Furthermore, we show how CTLP handles multi-step link prediction.

3.1. Problem definition

Generally, we can utilize a sequence of network snapshots to represent a dynamic network. The temporal link prediction aims to predict the links of subsequent one or some snapshots based on previous observations of the dynamic network. The notations are listed in Table 1.

Definition 1. (*Dynamic Network*) $\mathcal{G} = \{G_1, G_2, \cdots, G_T\}$ is a sequence of network snapshots, where $G_t = (\mathcal{V}, \mathcal{E}_t)$ represents the t-th snapshot of the dynamic network. $\mathcal{V}$ is the set of all vertices and $\mathcal{E}_t \subseteq |\mathcal{V}| \times |\mathcal{V}|$ denotes the temporal links at snapshot G_t. $\mathbf{A}_t$ is the adjacency matrix of G_t with the element $a_{t;i,j} = 1$ if there is a link between v_i and v_j, otherwise $a_{t;i,j} = 0$.

Link prediction in static networks primarily involves identifying whether a link exists based on the observed partial network structure. Similarly, link prediction in dynamic networks leverages previous network snapshot information to mine the underlying evolutionary patterns of networks, further predicting the future status of the network.

Definition 2. (*Temporal Link Prediction*) Given a sequence of dynamic network adjacency matrix with length l, $\{\mathbf{A}_1, \mathbf{A}_2, \cdots, \mathbf{A}_l\}$, the direct temporal link prediction aims to train a projection function f which projects the input snapshots sequence to $\mathbf{A}_{l+1}$, i.e., $\mathbf{A}_{l+1} = f(\mathbf{A}_1, \mathbf{A}_2, \cdots, \mathbf{A}_l)$.

Table 1
Symbols and Their Definitions.

Symbol	Definition
$\mathcal{V}$, $\|\mathcal{V}\|$	the set and the number of nodes in the dynamic network
$\mathbf{A}_t$	the adjacency matrix of the t-th network snapshot
W	the weight matrix of the neural network
b	the bias of the neural network
l	the number of the input snapshots
k	the number of the output snapshots in multi-step prediction
$\mathbf{A}'_{l+1}$	the predicted adjacency matrix of the $l+1$-th network snapshot
d	the number of embedding dimensions
$\mathbf{Z}_t$	the $\|\mathcal{V}\| \times d$ structure embedding matrix of t-th network snapshot
$\mathbf{H}_l$	the $\|\mathcal{V}\| \times d$ embedding matrix of the l input snapshots
C_{z_t}	the cell states at snapshot t of LSTM
f_{z_t}	the trigger value of the forget gate at snapshot t of LSTM
o_{z_t}	the trigger value of the output gate at snapshot t of LSTM
u_{z_t}	the trigger value of the update gate at snapshot t of LSTM
$\tilde{C}_{z_t}$	the new estimated candidate state at snapshot t of LSTM
σ	the sigmoid function

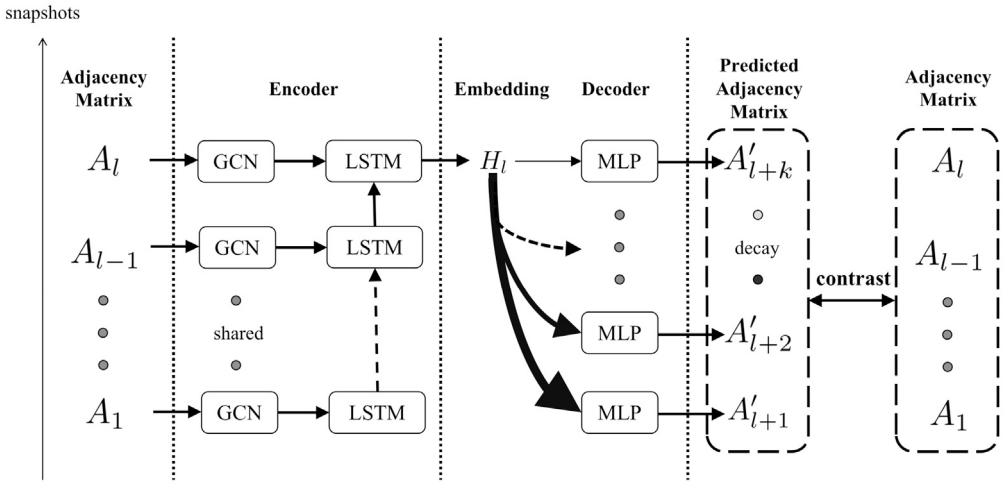

Fig. 2. Overall architecture of the CTLP framework, consisting of an encoder and a decoder. The encoder is composed of a shared GCN model and an LSTM model. The decoder layer is composed of multiple MLP. The predicted adjacency matrices are contrasted with the input adjacency matrices.

Definition 3. (*Network Embedding-based Temporal Link Prediction*) Given a sequence of dynamic network adjacency matrix with length l, $\{\mathbf{A}_1, \mathbf{A}_2, \cdots, \mathbf{A}_l\}$, the network embedding based temporal link prediction firstly learns the network embedding $\mathbf{Z}_l$ by a mapping function g, i.e., $\mathbf{Z}_l = g(\mathbf{A}_1, \mathbf{A}_2, \cdots, \mathbf{A}_l)$, and then learns another projection function f that maps the embedding to the $\mathbf{A}_{l+1}$, i.e., $\mathbf{A}_{l+1} = f(\mathbf{Z}_l)$

Definition 4. (*Multi-step Temporal Link Prediction*) Given a sequence of dynamic network adjacency matrix with length l, $\{\mathbf{A}_1, \mathbf{A}_2, \cdots, \mathbf{A}_l\}$, the multi-step temporal link prediction aims to train a projection function which projects the input snapshots sequence to the output sequence, i.e., $\mathbf{A}_{l+1}, \mathbf{A}_{l+2}, \cdots, \mathbf{A}_{l+k} = f(\mathbf{A}_1, \mathbf{A}_2, \cdots, \mathbf{A}_l)$.

Unsupervised temporal link prediction means that when we train our prediction model, that maps $\{\mathbf{A}_1, \mathbf{A}_2, \cdots, \mathbf{A}_l\}$ to $\mathbf{A}_{l+1}$ or $\{\mathbf{A}_{l+1}, \mathbf{A}_{l+2}, \cdots, \mathbf{A}_{l+k}\}$, we don't utilize any network data from $\mathbf{A}_{l+1}$ or $\{\mathbf{A}_{l+1}, \mathbf{A}_{l+2}, \cdots, \mathbf{A}_{l+k}\}$ as the supervised information.

3.2. CTLP framework

The overall architecture of the CTLP framework is depicted in Fig. 2. Our unsupervised temporal link prediction framework adopts the encoder–decoder architecture, where the encoder is employed to extract the nonlinear temporal dependencies and network structure information, and the decoder then maps the extracted network features back to the original representing space. Such encoder–decoder architecture can handle structural and temporal nonlinearities. Therefore, CTLP a well-designed encoder-decoder model can effectively capture the spatial and temporal features of dynamic networks, and perform link prediction in a unified manner.

We adopt a structure encoder to extract the nonlinear spatial features and use a sequence model to capture the nonlinear temporal features and evolutionary patterns, as our encoder. The structure encoder can be GCN [37], VGAE [10], Autoencoder, etc., and the sequence model can adopt an RNN-based model, such as TCN, Transformer-based model, and others. In this paper, we utilize GCN and LSTM, respectively. An LSTM cell could be treated as a simple layer, where the terms with subscript f represent parameters of the forget gate, parameters of the input gate are denoted with the terms with subscripts i and C, and o represent the output gate's parameters.

3.2.1. Graph convolutional network layer

A typical GCN layer takes the feature matrix $\mathbf{X}$ as input and applies localized first-order approximation to perform the spectral graph convolution operation based on the adjacency matrix $\mathbf{A}$. The operation of our GCN unit for the t-th snapshot of the dynamic network is defined as:

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

where $\mathbf{I}$ denotes an identity matrix, $\mathbf{A}_t$ represents the adjacency matrix of t-th snapshot in the dynamic network and $\hat{\mathbf{A}} = \mathbf{A}_t + \mathbf{I}$, $\mathbf{I} \in \mathbb{R}^{|\mathcal{V}|\times|\mathcal{V}|}$ and $\mathbf{A}_t \in \mathbb{R}^{|\mathcal{V}|\times|\mathcal{V}|}$; $\hat{\mathbf{D}}$ is the degree matrix of $\hat{\mathbf{A}}$; $\hat{\mathbf{D}}^{-\frac{1}{2}}\hat{\mathbf{A}}\hat{\mathbf{D}}^{-\frac{1}{2}}$ is the approximated graph convolution filter; $\mathbf{W}_\mu$ denotes the learned parameters; $f(\cdot)$ is the nonlinear activation function; and $\mathbf{Z}_t$ is the node embedding of t-th snapshot obtained by GCN, $\mathbf{Z}_t \in \mathbb{R}^{|\mathcal{V}|\times d}$ and d is the dimension of embedding, $\mathbf{Z}_t$ represents the vector representations for the $|\mathcal{V}|$ nodes at snapshot G_t. When the input is the adjacency matrix sequence of l snapshots $\left\{\mathbf{A}_1, \mathbf{A}_2, \cdots, \mathbf{A}_l\right\}$, the output is $\left\{\mathbf{Z}_1, \mathbf{Z}_2, \cdots, \mathbf{Z}_l\right\}$ obtained through the shared GCN layer.

3.2.2. Long short-term memory layer

The temporal dependencies in dynamic networks can not be captured by the structure encoder. LSTM is capable of handling scenarios with long-term temporal dependencies. Therefore, We introduce the hidden state representation extracted through a single LSTM layer, integrating feature information from previous consecutive l snapshots into the ultimate embedding $h_{l;i}$. In the LSTM layer, we take $\left\{z_{1;i}, z_{2;i}, \cdots, z_{l;i}\right\}$ as input, $z_{t;i}$ indicates the i-th row of $\mathbf{Z}_t$. The formalized process of the LSTM is as follows:

$$h_{z_{t;i}} = o_{z_{t;i}} * \tanh\left(C_{z_{t;i}}\right), \tag{2}$$

$$o_{z_{t;i}} = \sigma_{z_{t;i}}\left(W_o\left[y_{h_{t-1;i}}, z_{t;i}\right] + b_o\right), \tag{3}$$

$$C_{z_{t;i}} = f_{z_{t;i}} * C_{z_{t-1;i}} + i_{z_{t;i}} * \tilde{C}_{z_{t;i}}, \tag{4}$$

$$\tilde{C}_{z_{t;i}} = \tanh\left(W_C \cdot \left[h_{z_{t-1;i}}, z_{t;i} + b_c\right]\right), \tag{5}$$

$$u_{z_{t;i}} = \sigma\left(W_u \cdot \left[h_{z_{t-1;i}}, z_{t;i}\right] + b_u\right), \tag{6}$$

$$f_{z_{t;i}} = \sigma\left(W_f \cdot \left[h_{z_{t-1;i}}, z_{t;i} + b_f\right]\right), \tag{7}$$

where C_{z_t} is the cell state of LSTM, f_{z_t}, $_{z_t}$, and u_{z_t} represent the values triggering the forget gate, output gate, and update gate, respectively. $\tilde{C}_{z_t}$ is the candidate state, W and b represents the weight matrix and bias vector, respectively. σ is the sigmoid activation function. The input of LSTM layer is $\left\{\mathbf{Z}_1, \mathbf{Z}_2, \cdots, \mathbf{Z}_l\right\}$, and the output is $\mathbf{H}_l \in \mathbb{R}^{|\mathcal{V}|\times d}$, where d is the dimension of embedding.

3.2.3. Decoder layer

Given the input sequences $\left\{\mathbf{A}_1, \mathbf{A}_2, \cdots, \mathbf{A}_l\right\}$, we have obtained the temporal embedding $\mathbf{H}_l$ which is used for predicting the future links that be represented as $\mathbf{A}_{l+1}$ at time $t+1$. Subsequently, a two-layer fully connected neural network is employed as the decoder to predict future links. Specifically as follows:

$$\mathbf{A}'_{l+1} = \sigma\left(W_\phi H_l + b_\phi\right), \tag{8}$$

where W_ϕ and b_ϕ are the weight matrix and bias vector of the decoder.

3.3. Contrastive loss

Contrastive loss [44] is employed to make the past snapshots and the predicted snapshots of dynamic networks share as much information as possible. The contrastive loss aims to distinguish the links belonging to the same node at different snapshots from the links belonging to the other nodes. For any node v_i, the links belong to node v_i at t-th snapshot can be defined as the i-th row $a_{t;i}$ of $\mathbf{A}_t$, and $a_{t;i}$ is treated as the anchor, the links belong to node v_i at predicted $l+1$ th snapshot, $a'_{l+1;i}$, composes the positive sample, and the negative samples naturally consist of other nodes in the predicted snapshots. To enhance the decoder's reconstruction capability in the numerous zero elements in $\mathbf{A}_t$ resulting from the sparsity of the network, more attention should be paid to existing links during backpropagation. So we introduce a weighting matrix $\mathbf{P}$ to penalize the incorrect prediction of links, $\mathbf{P}_{ij} = \beta$ for $\mathbf{A}_{t;ij} > 0$, else 1, where β is a hyperparameter to control the effect of penalizing incorrect prediction. The loss function of predicting the links at snapshot $l+1$ based on the past l snapshots of the dynamic network can be expressed as:

$$\mathcal{L} = \sum_{t=1}^{l} \sum_{i=1}^{N} -\log \left(\frac{sim(a'_{l+1;i}, a_{t;i} \odot p_i)}{\sum_{j=1}^{N} sim(a'_{l+1;j}, a_{t;i} \odot p_i)} \right), \tag{9}$$

where $a_{l+1;i}$ is the i-th row of the predicted adjacency matrix $\mathbf{A}_{l+1}$, $a_{t;i}$ is the i-th row of $\mathbf{A}_t$ in the input sequence $\{\mathbf{A}_1, \mathbf{A}_2, \cdots, \mathbf{A}_l\}$ and p_i is the i-th row of the penalty matrix P, $\odot$ is the Hadamard product and sim is a similarity function.

3.4. Multi-step temporal link prediction

For the multi-step link prediction problem, we employ k decoders with the same but not shared parameters to obtain the adjacency matrices of k future snapshots, where k represents the length of predicted steps. We are able to obtain the predicted sequence, $\left\{\mathbf{A}'_{l+1}, \mathbf{A}'_{l+2}, \cdots, \mathbf{A}'_{l+k}\right\}$ through the k decoders, where l denotes the length of the input sequence.

Considering the structure information shared between different snapshots will decrease with time, we incorporate an exponential decay function into the loss function for multi-step prediction to simulate the influence of information sharing decays with time. The loss function of predicting the links of future k snapshots based on the past l snapshots of the dynamic network can be expressed as:

$$\mathcal{L}_m = \sum_{t'=l+1}^{l+k} \sum_{t=1}^{l} \sum_{i=1}^{N} \theta(t'-t) - \log \left(\frac{sim(a'_{t';i}, a_{t;i} \odot p_i)}{\sum_{j=1}^{N} sim(a'_{t';j}, a_{t;i} \odot p_i)} \right), \tag{10}$$

where θ is the decay constant.

3.5. Complexity analysis

The computational complexity of optimizing CTLP is primarily determined by the weight parameters within the framework. Similar to some other deep learning methods based on contrastive learning, we assume that a simple CTLP framework consists of a structure encoder layer with m_1 units, the hidden size of an LSTM layer is m_2 and k decoder layers, then the complexity of our model is:

$$\mathcal{O}(n) \sim n\left(m_1 + km_2\right) + 4\left(m_1 m_2 + m_2^2 + m_2\right), \tag{11}$$

where n is the number of all nodes. The computational complexity varies based on changes in the model's structure. Despite the numerous parameters, CTLP can still quickly complete training and testing with the assistance of GPU acceleration.

4. Experimental setup

4.1. Datasets

Our experiments are performed on six real-world dynamic network datasets. We divide each dynamic network dataset into multiple consecutive snapshots. Table 2 presents the comprehensive statistics of these datasets, including the number of nodes, edges, and snapshots in each dynamic network. Further details are listed below.

1) **HS11 and HS12** [45]: These are two social networks that record 77,602 contact events between 242 individuals at a French school.
2) **ENRON** [46]: The dataset comprises email communications among employees at Enron Inc. spanning from January 1999 to July 2002. In order to control the scale of the network, in this paper, our analysis specifically focuses on email communications between executives.
3) **Cellphone** [47]: This network is constructed based on cellphone call records within the fictional Paraiso movement, spanning a ten-day period in June 2006. Each node denotes a distinct cellphone, and an edge is established between two cellphones when a phone call is made between them.
4) **UCIMsg** [48]: The dataset comprises an online social network with private messages exchanged at the University of California. If one student sends a message to another student, an edge is created between them.
5) **Bitcoin** [48]: The network represents a trust relationship among individuals who engage in Bitcoin trading on the Bitcoin Alpha platform. Due to the anonymity of Bitcoin users, it is essential to track their reputations to safeguard against interactions with dishonest and high-risk users.

4.2. Baseline methods

We select two effective static network embedding methods and several temporal link prediction methods as our baselines and they are introduced as follows:

Algorithm 1 Contrastive unsupervised temporal link prediction algorithm.

Input: GCN model g_μ, LSTM model g_ψ, adjacency matrix sequence of l snapshots $\{\mathbf{A}_1, \mathbf{A}_2, \cdots, \mathbf{A}_l\}$, number of predicted snapshots k, k decoders $\{g_{\phi;1}, g_{\phi;2}, \cdots, g_{\phi;k}\}$.
Output: Node embedding of different snapshots

for $epoch \leftarrow 1, 2, \cdots$ **do**:
  for $t = 1$ *to* l **do**:
    Obtain the spatial embedding of each input snapshot, $\mathbf{Z}_t = g_\mu(\mathbf{A}_t)$
  end for
  Obtain the temporal embedding of l input snapshots, $\mathbf{H}_l = g_\psi(\mathbf{Z}_1, \mathbf{Z}_2, \cdots, \mathbf{Z}_l)$
  for $t' = l + 1$ *to* $l + k$ **do**:
    Generate links of the future snapshot, $\mathbf{A}'_{t'} = g_{\phi;t'}(\mathbf{H}_l)$
  end for
  Compute the contrastive loss $\mathcal{L}_m$ with Eq. (10)
  Update parameters of model with Adam to minimize $\mathcal{L}_m$
end for

Table 2
Statistical Details of the Datasets.

Datasets	#Node	#Edge	#Snapshot
HS11	126	424	7
HS12	180	538	8
ENRON	151	292	12
Cellphone	400	512	10
UCIMsg	1899	20296	10
Bitcoin	3,783	24,186	7

Table 3
Average MAP Scores of Single-Step Link Prediction on Six Datasets.

Methods	HS11	HS12	ENRON	Cellphone	UCIMsg	Bitcoin
DeepWalk	0.0982	0.1013	0.1210	0.0689	0.0320	0.0127
LINE	0.0933	0.1152	0.1227	0.0128	0.0219	0.0207
DynAE	0.3373	0.3676	0.4187	0.6983	0.0726	0.0084
DynRNN	0.1496	0.2008	0.2740	0.4003	0.0493	0.0706
DynAERNN	0.1966	0.3196	0.5163	0.7128	**0.1603**	0.0840
DySAT	0.1242	0.0683	0.0713	0.0532	0.0442	0.0896
VGRNN	0.2537	0.2322	0.2709	0.2163	0.0609	0.0542
TMF	0.1961	0.1592	0.1968	0.3159	0.0718	0.0585
CTLP	**0.3655**	**0.3850**	**0.4395**	**0.8172**	0.1338	**0.1043**

1) **DeepWalk** [33]: It is the first network embedding technique that utilizes random walks on networks, the embedding vectors can reveal the latent similarity of pair nodes.
2) **LINE** [49]: This is a static network embedding method that utilizes a semi-supervised method to extract non-linear features of the network, and analyzes local and global structures of networks using different orders proximity.
3) **DynAE, DynRNN and DynAERNN** [40]: The three methods are direct temporal link prediction methods. DynAE is a deep autoencoder with multiple fully connected layers, DynRNN adopts RNN layers as the encoder and decoder layers of the autoencoder, and DynAERNN combines the fully connected layers with RNN layers as the encoder.
4) **DySAT** [19]: It is an innovative neural model that can learn effective node embeddings and capture the dynamic network evolution information through self-attention incorporating structural neighborhood and temporal dynamics.
5) **VGRNN** [20]: This model is hierarchical and variational, incorporating extra latent variables to extract the hidden states of dynamic networks with a graph recurrent neural network. It effectively captures changes in both network topology and node attributes in dynamic networks.
6) **TMF** [23]: This is an unsupervised temporal link prediction method, which utilizes matrix factorization to explicitly model the evolving features of network snapshots over time.

Among them, DeepWalk, LINE, DynAE, DynRNN, and DynAERNN are tailored for single-step link prediction problems, while DySAT, VGRNN, and TMF, are capable of addressing multi-step link prediction tasks.

4.3. Implementation details

For illustration, we use l to represent the length of the input snapshots and k to represent the length of the output snapshots in the multi-step link prediction task.

Table 4
Average AUC Scores of Single-Step Link Prediction on Six Datasets.

Methods	HS11	HS12	ENRON	Cellphone	UCIMsg	Bitcoin
DeepWalk	0.6611	0.6940	0.7621	0.6057	0.6846	0.5253
LINE	0.6346	0.6510	0.6931	0.5690	0.6992	0.5344
DynAE	0.5684	0.6251	0.6230	0.6576	0.5139	0.5008
DynRNN	0.5795	0.6335	0.6942	0.5981	0.5399	0.5853
DynAERNN	0.5812	0.6744	0.7655	0.8589	0.5874	0.5827
DySAT	0.4908	0.4949	0.5017	0.4858	0.4948	0.4837
VGRNN	**0.7813**	**0.8422**	0.8079	0.8650	0.7293	**0.6450**
TMF	0.5520	0.7338	0.6887	0.6259	0.5605	0.5238
CTLP	0.7224	0.8127	**0.8240**	**0.8682**	**0.7687**	0.6071

4.3.1. Implementation details of baseline methods

For static network embedding baselines, i.e., DeepWalk and LINE, we first obtain the embedding of the current snapshot and subsequently predict the links of the next snapshot through an inner product decoder [10]. For direct temporal link prediction methods, i.e., DynAE, DynRNN and DynAERNN, during training, for the input of the $1, 2, \cdots, l$ snapshot, we use the links at the $l+1$-th snapshot as the supervision information, while in the testing process, we utilize $2, 3, \cdots, l+1$ snapshot as the input to predict the links at the $l+2$-th snapshot. For network embedding-based temporal link prediction methods, i.e., DySAT and VGRNN, during the training process, for the input of $1, 2, \cdots, l$ snapshot, in the single-step link prediction, we use the $2, 3, \cdots, l+1$ snapshot as the reconstruction target. In the testing process, we use the $2, 3, \cdots, l+1$ snapshot as the input to obtain the embedding of the $l+2$-th snapshot and then utilize the obtained embedding to predict the links in the $l+2$-th snapshot by an inner product decoder. While in the multi-step link prediction, during training, we use the $1+k, 2+k, \cdots, l+k$ snapshot as the reconstruction target, and use the obtained embedding of each snapshot to predict the links by an inner product decoder. For all baseline methods, we adopt the default parameter configurations as outlined in their original papers. As for the embedding dimension, we employ the same settings as our framework on each dataset.

4.3.2. Implementation details of CTLP framework

For all datasets, Adam optimization algorithm with the learning rate 0.001 is utilized to optimize the parameters of the model. And the early stopping is employed to mitigate overfitting. For HS11, HS12, ENRON, and Cellphone datasets, we do not add the penalty matrix since these networks are dense. For UCIMsg and Bitcoin datasets, we set $\beta = 60, 100$ of the penalty matrix, respectively. The embedding dimension d is set to 64 for three small datasets, i.e., HS11, HS12, and ENRON, $d = 128$ for Cellphone, UCIMsg, and Bitcoin datasets. For HS11, HS12, ENRON, and Cellphone datasets, we utilize the exponential function of vector inner product as the similarity measure function sim, while for larger UCIMsg and Bitcoin datasets, we only apply vector inner product.

4.4. Evaluation metrics

Mean Average Precision (MAP) and Area Under the Curve (AUC) are utilized as our evaluation metrics. The MAP calculates the average precision across all nodes and can be defined as:

$$\mathrm{MAP} = \frac{\sum_i AP(i)}{|\mathcal{V}|}, \tag{12}$$

where $AP(i) = \frac{\sum_k p@k(i) \cdot \mathbb{I}\left\{E_{pred_i}(k) \in E_{gt_i}\right\}}{\left|\left\{k : E_{pred_i}(k) \in E_{gt_i}\right\}\right|}$ and $p@k(i) = \frac{\left|E_{pred_i}(k) \cap E_{gt_i}\right|}{k}$, E_{pred_i} and E_{gt_i} denote the predicted links and existing links that are connected to node i, respectively. A higher MAP value suggests that the model can provide more accurate predictions for the majority of nodes.

In u independent comparisons, if the score of existing links is higher than the score of non-existent links u' times and they receive the same score u'' times, then we can conclude:

$$\mathrm{AUC} = \frac{u' + 0.5u''}{u}. \tag{13}$$

To mitigate the impact of sparsity, we randomly sample the same number of non-existent links as existing links within the network before calculation.

5. Experimental results

5.1. Single-step link prediction

All methods are evaluated for single-step link prediction tasks on dynamic networks using MAP and AUC. Specifically, we set the input snapshot length $l = 3$. For all snapshots in each dataset, we partition them into multiple groups of input and target output with the sliding window size 4 and step size 1. We take the average MAP and AUC scores of all input groups as the evaluation index.

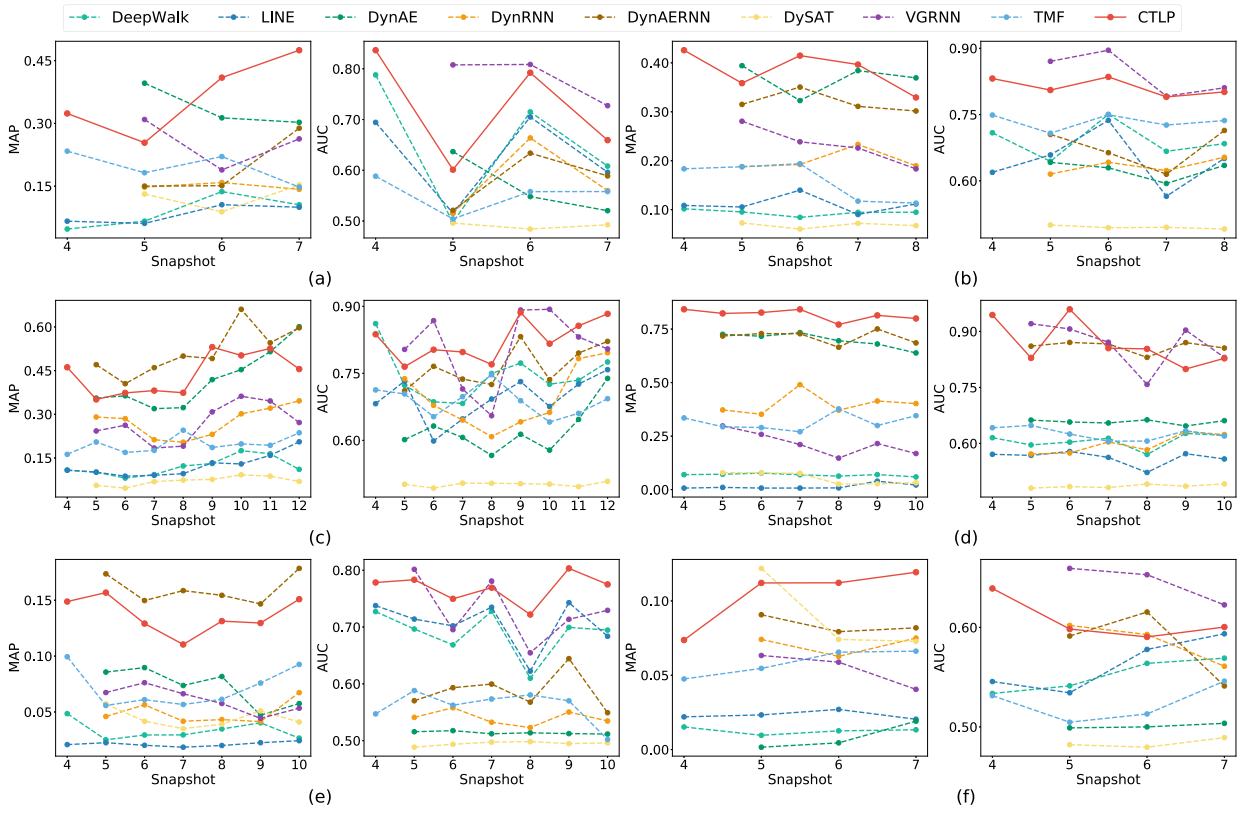

Fig. 3. Single-step link prediction results in terms of MAP and AUC values of different methods across various real-world networks (a) HS11. (b) HS12. (c) ENRON. (d) Cellphone. (e) Ucimsg. (h) Bitcoin.

Table 3 and Table 4 present the results of single-step link prediction. Notably, the CTLP framework consistently outperforms baseline methods.

As shown in Fig. 3, considering that the varying patterns can help predict the structure, our framework consistently outperforms the unsupervised method TMF on all six datasets. This is because TMF is based on matrix factorization and struggles to capture complex temporal patterns of dynamic networks. Moreover, our method exhibits superior performance compared to all supervised methods on ENRON and Cellphone datasets. Across the HS11, HS12, and Bitcoin datasets, our method demonstrates a higher average MAP value than supervised methods, with the average AUC value being only slightly lower by a few percentage points compared to the embedding-based method VGRNN. On Ucimsg dataset, the average MAP value of our method is slightly below that of the direct temporal link prediction method DynAERNN, but our method excels with an average AUC value nearly 20%.

5.2. *Multi-step link prediction*

We adopt MAP and AUC to evaluate some methods on HS12, ENRON, Cellphone, and UCIMsg networks in a multi-step link prediction task. To be specific, we set the input snapshot length $l = 3$ and the output snapshot length $k = 3$. For each dataset, we partition all snapshots into multiple groups of input and target output with the sliding window size 6 and step size 1. We take the average MAP and AUC scores of all input groups as the evaluation metrics. The results of multi-step link prediction are presented in Fig. 4. It is evident that the CTLP framework significantly outperforms the baseline methods. Furthermore, the performance of CTLP on multi-step temporal links surpasses that of single-step link prediction, indicating the effectiveness of our framework in capturing network evolution patterns. We visualize the prediction results of the fifth snapshot of the ENRON dataset in Fig. 5 to specifically demonstrate the effectiveness of our method. Successful predictions are denoted by blue links, while unsuccessful predictions are represented by red links. It is evident that the prediction results of CTLP are significantly better than other baseline methods.

5.3. *Parameter sensitivity analysis*

We first study the impact of the embedding dimensions with different values on the single-step prediction task. Subsequently, we investigate the effect of the decay constant θ and the length l of historical snapshots on the multi-step prediction performance.

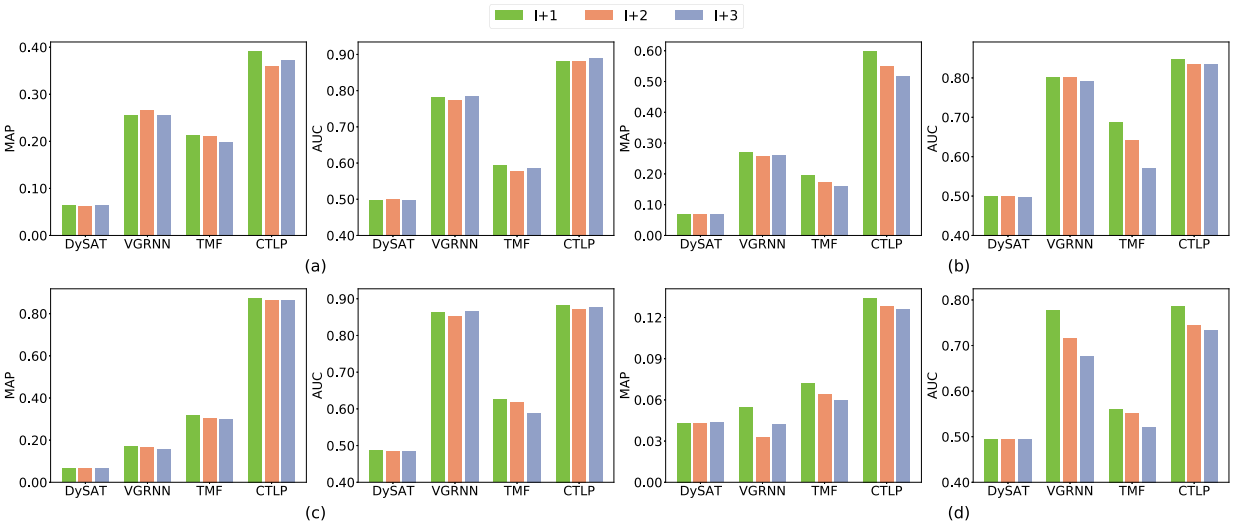

Fig. 4. Multi-step link prediction results in terms of MAP and AUC values of different methods across various real-world networks (a) HS12. (b) ENRON. (c) Cellphone. (d) Ucimsg.

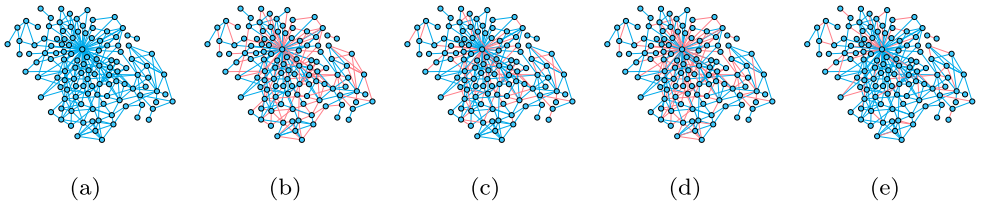

Fig. 5. The real links in the fifth snapshot of ENRON network and the links predicted by different methods in the multi-step prediction task, the blue ones are successfully predicted and the red ones are failed. (a) The real links. (b) DySAT, 51.27% links successfully predicted. (c) VGRNN, 71.52% links successfully predicted. (d) TMF, 56.49% links successfully predicted. (e) CTLP, 80.02% links successfully predicted.

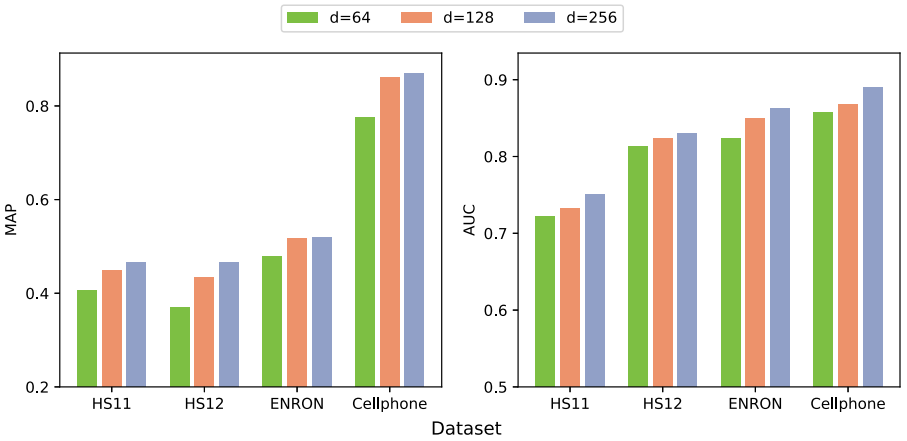

Fig. 6. The impact of embedding dimension on MAP and AUC values, where the embedding dimension is set to 64, 128, and 256 for each dataset, respectively.

5.3.1. Embedding dimension d

We vary the embedding dimension across 64, 128, and 256 and reported the corresponding MAP and AUC results on different dynamic networks in Fig. 6. It can be found that higher embedding dimensions generally yield better performance. This improvement can be ascribed to the lower-dimensional vector potentially losing more effective spatio-temporal information compared to a higher-dimensional one. However, it is noteworthy that if d exceeds a certain value, it may not provide additional contributions.

Table 5
Average MAP Scores of Multi-Step Link Prediction on HS11 and HS12 Datasets with Different θ Values.

θ value	HS11			HS12		
	$l+1$	$l+2$	$l+3$	$l+1$	$l+2$	$l+3$
$\theta=0.2$	0.3278	0.4187	0.4068	0.4033	0.3882	0.3697
$\theta=0.4$	0.3324	0.4155	0.4079	0.4023	0.3572	0.3698
$\theta=0.6$	0.3381	0.4252	0.4032	0.3978	0.3820	0.3706
$\theta=0.8$	0.3342	0.4046	0.3793	0.4198	0.3932	0.3779
$\theta=1.0$	0.3626	0.4199	0.4083	0.4075	0.3679	0.3620

Table 6
Average MAP Scores of Multi-Step Link Prediction on ENRON and Cellphone Datasets with Different θ Values.

θ value	ENRON			Cellphone		
	$l+1$	$l+2$	$l+3$	$l+1$	$l+2$	$l+3$
$\theta=0.2$	0.5788	0.5227	0.4789	0.8717	0.8640	0.8605
$\theta=0.4$	0.5810	0.5196	0.4810	0.8726	0.8674	0.8618
$\theta=0.6$	0.5778	0.5265	0.4848	0.8762	0.8662	0.8573
$\theta=0.8$	0.5859	0.5362	0.4990	0.8713	0.8632	0.8571
$\theta=1.0$	0.5761	0.5232	0.4870	0.8762	0.8634	0.8601

Table 7
Average MAP Scores of Multi-Step Link Prediction on ENRON and Cellphone Datasets with Different l Values.

l value	ENRON			Cellphone		
	start	median	end	start	median	end
$l=1$	0.6036	-	-	0.8071	-	-
$l=2$	0.5986	0.5298	-	0.8661	0.8651	-
$l=3$	0.5985	0.5502	0.5190	0.8744	0.8644	0.8643
$l=4$	0.5392	0.5074	0.4941	0.8705	0.8687	0.8618
$l=5$	0.5096	0.5093	0.5026	0.8590	0.8796	0.8552

5.3.2. Decay constant θ

To simulate the decay of information sharing over time in the loss function of multi-step prediction, we employ an exponential decay function, with the decay constant θ regulating the extent of information shared between different snapshots. We evaluate the performance of CTLP across various θ values. The decay constant ranges from 0.2 to 1.0, and for each value of θ, the length of input snapshots l is 3 and the length of output snapshots k is 3. Table 5 and Table 6 show the experimental results. Observing the results across different datasets, we note that varying θ values yield different outcomes. Specifically, on HS11 and Cellphone datasets, the predicted MAP value increases with the rise of θ. Conversely, on HS12 and Enron datasets, the maximum MAP value is achieved when $\theta=0.8$. This is because the amount of information shared by different snapshots of various networks is different, which also conforms to the general rule in the real world.

5.3.3. Length of historical snapshots l

To assess the influence of historical snapshot input length on model performance, we train CTLP using different input lengths. The input length ranges from 1 to 5. Subsequently, we assess the variation in average MAP values on ENRON and Cellphone datasets for the multi-step link prediction task. We set the length of predicted snapshots equal to the input length, i.e., $l=k$. The experimental results are presented in Table 7, showcasing the predicted start, median, and end snapshot results, respectively. The results indicate that as we predict farther into the future, the effectiveness of the predictions tends to decrease. Despite this, our framework consistently demonstrates strong capability in addressing the challenge of predicting longer snapshot sequences.

5.4. Ablation experiments

We further perform a multi-step link prediction task on the HS11, HS12, ENRON, and Cellphone datasets. We compare CTLP with three models degraded by removing the structure encoder GCN (denoted as CTLP†), replacing the GCN module with two fully connected layers (denoted as CTLP††) and removing the sequence model LSTM (denoted as CTLP‡). Table 8 displays the average MAP values of these models. The results demonstrate a sharp decrease in MAP scores when discarding the GCN module or LSTM module. This is because discarding structure encoder GCN or sequence model LSTM will lead the framework to lose the structural or temporal features of dynamic networks. When utilizing fully connected layers (FCN) as the structure encoder, the results are

Table 8
Ablation Study of Multi-Step Link Prediction on HS12 and Cellphone Datasets.

Methods	HS12			Cellphone		
	$l+1$	$l+2$	$l+3$	$l+1$	$l+2$	$l+3$
CTLP	**0.3915**	**0.3603**	**0.3726**	**0.8744**	**0.8644**	**0.8643**
CTLP†	0.1795	0.1706	0.1850	0.2461	0.2419	0.2360
CTLP††	0.3752	0.3531	0.3599	0.2950	0.2870	0.2882
CTLP‡	0.2854	0.2622	0.2592	0.2374	0.2283	0.2194

better than not using any structure encoder but inferior to employing GCN, which indicates the effectiveness of GCN in capturing the structure feature of each network snapshot.

6. Conclusion

Unlike previous studies that only focus on supervised temporal link prediction in dynamic networks, in this paper, we propose a deep contrastive framework called CTLP to address the challenge of unsupervised temporal link prediction in dynamic networks. The CTLP framework is based on deep encoder-decoder architecture, which can capture the complex nonlinear spatial features and temporal dependencies of dynamic networks. Moreover, our framework could handle the multi-step temporal link prediction problem of dynamic networks. Additionally, we perform extensive experiments and compare CTLP with state-of-the-art link prediction approaches on six real-world dynamic network datasets to verify the superiority of CTLP. Experimental results demonstrate that the CTLP framework significantly outperforms unsupervised methods, and even exceeds the supervised methods in some cases.

As for our future work, more attention will be focused on optimizing the contrast form that can capture more meaningful evolution characteristics of dynamic networks. Furthermore, we will also attempt to expand CTLP to more large-scale dynamic networks. Besides, we could also fuse node features to learn more effective dynamic network embedding and further enhance the performance of the model in temporal link prediction.

CRediT authorship contribution statement

Pengfei Jiao: Supervision, Methodology, Conceptualization. **Xinxun Zhang:** Writing – review & editing, Writing – original draft. **Zehao Liu:** Writing – review & editing, Writing – original draft. **Long Zhang:** Visualization, Validation, Methodology, Conceptualization. **Huaming Wu:** Writing – original draft. **Mengzhou Gao:** Writing – original draft, Supervision. **Tianpeng Li:** Writing – review & editing. **Jian Wu:** Writing – review & editing, Writing – original draft.

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.

Acknowledgement

This work was supported in part by the National Natural Science Foundation of China under Grants 62372146 and 62003120, in part by the Fundamental Research Funds for the Provincial Universities of Zhejiang Grant GK229909299001-008, and in part by the Zhejiang Provincial Natural Science Foundation of China under Grant LDT23F01015F01, and in part by the Hangzhou Artificial Intelligence Major Scientific and Technological In-novation Project under Grant 2022AIZD0114.

References

[1] S.P. Borgatti, M.G. Everett, J.C. Johnson, F. Agneessens, Analyzing Social Networks, SAGE Publications, Limited, 2024.
[2] J. Gehrke, P. Ginsparg, J.M. Kleinberg, Overview of the 2003 KDD cup, SIGKDD Explor. 5 (2) (2003) 149–151.
[3] A. Theocharidis, S. Van Dongen, A.J. Enright, T.C. Freeman, Network visualization and analysis of gene expression data using biolayout express 3d, Nat. Protoc. 4 (10) (2009) 1535.
[4] Y. Zhang, J. Chen, Z. Cheng, X. Shen, J. Qin, Y. Han, Y. Lu, Edge propagation for link prediction in requirement-cyber threat intelligence knowledge graph, Inf. Sci. 653 (2024) 119770.
[5] S. Wasserman, K. Faust, Social Network Analysis: Methods and Applications, Cambridge University Press, 1994.
[6] G.A. Pavlopoulos, A. Wegener, R. Schneider, A survey of visualization tools for biological network analysis, BioData Min. 1 (2008) 1–11.
[7] M. Yang, J. Liu, L. Chen, Z. Zhao, X. Chen, Y. Shen, An advanced deep generative framework for temporal link prediction in dynamic networks, IEEE Trans. Cybern. 50 (12) (2020) 4946–4957.
[8] Z. Wang, Y. Chai, C. Sun, X. Rui, H. Mi, X. Zhang, P.S. Yu, A weighted symmetric graph embedding approach for link prediction in undirected graphs, IEEE Trans. Cybern. 54 (2) (2024) 1037–1047.

[9] J. Ding, L. Jiao, J. Wu, F. Liu, Prediction of missing links based on community relevance and ruler inference, Knowl.-Based Syst. 98 (2016) 200–215.
[10] T.N. Kipf, M. Welling, Variational graph auto-encoders, CoRR, arXiv:1611.07308 [abs].
[11] A. Agibetov, Neural graph embeddings as explicit low-rank matrix factorization for link prediction, Pattern Recognit. 133 (2023) 108977.
[12] Y. Wang, H. Wang, W. Lu, Y. Yan Hygge, Hyperbolic graph attention network for reasoning over knowledge graphs, Inf. Sci. 630 (2023) 190–205.
[13] Y. Xiu, X. Liu, K. Cao, B. Chen, W.K.V. Chan, An extended self-representation model of complex networks for link prediction, Inf. Sci. 662 (2024) 120254.
[14] Z. Qiu, J. Wu, W. Hu, B. Du, G. Yuan, P.S. Yu, Temporal link prediction with motifs for social networks, IEEE Trans. Knowl. Data Eng. 35 (3) (2023) 3145–3158.
[15] H. Hou, H. Kretschmer, Z. Liu, The structure of scientific collaboration networks in Scientometrics, Scientometrics 75 (2) (2008) 189–202.
[16] M. Yang, M. Zhou, H. Xiong, I. King, Hyperbolic temporal network embedding, IEEE Trans. Knowl. Data Eng. 35 (11) (2023) 11489–11502.
[17] J. Chen, J. Zhang, X. Xu, C. Fu, D. Zhang, Q. Zhang, Q. Xuan, E-LSTM-D: a deep learning framework for dynamic network link prediction, IEEE Trans. Syst. Man Cybern. Syst. 51 (6) (2021) 3699–3712.
[18] S. Bonner, A.A. Abarghouei, P.T.G. Jackson, J. Brennan, I. Kureshi, G. Theodoropoulos, A.S. McGough, B. Obara, Temporal neighbourhood aggregation: predicting future links in temporal graphs via recurrent variational graph convolutions, in: 2019 IEEE International Conference on Big Data (Big Data), IEEE, 2019, pp. 5336–5345.
[19] A. Sankar, Y. Wu, L. Gou, W. Zhang, H. Yang Dysat, Deep neural representation learning on dynamic graphs via self-attention networks, in: Proceedings of the 13th International Conference on Web Search and Data Mining, ACM, 2020, pp. 519–527.
[20] E. Hajiramezanali, A. Hasanzadeh, K.R. Narayanan, N. Duffield, M. Zhou, X. Qian, Variational graph recurrent neural networks, in: Annual Conference on Neural Information Processing Systems, 2019, pp. 10700–10710.
[21] T. Li, W. Wang, P. Jiao, Y. Wang, R. Ding, H. Wu, L. Pan, D. Jin, Exploring temporal community structure via network embedding, IEEE Trans. Cybern. 53 (11) (2023) 7021–7033.
[22] P. Goyal, N. Kamra, X. He, Y. Liu, Dyngem: deep embedding method for dynamic graphs, CoRR, arXiv:1805.11273 [abs].
[23] W. Yu, C.C. Aggarwal, W. Wang, Temporally factorized network modeling for evolutionary network analysis, in: Proceedings of the Tenth ACM International Conference on Web Search and Data Mining, ACM, 2017, pp. 455–464.
[24] Z. Peng, W. Huang, M. Luo, Q. Zheng, Y. Rong, T. Xu, J. Huang, Graph representation learning via graphical mutual information maximization, in: WWW '20: The Web Conference 2020, Taipei, Taiwan, April 20-24, 2020, ACM / IW3C2, 2020, pp. 259–270.
[25] Y. Zhu, Y. Xu, F. Yu, Q. Liu, S. Wu, L. Wang, Graph contrastive learning with adaptive augmentation, in: WWW '21: The Web Conference 2021, Virtual Event / Ljubljana, Slovenia, April 19-23, 2021, ACM / IW3C2, 2021, pp. 2069–2080.
[26] K. Hassani, A.H.K. Ahmadi, Contrastive multi-view representation learning on graphs, in: Proceedings of the 37th International Conference on Machine Learning, ICML 2020, 13-18 July 2020, Virtual Event, in: Proceedings of Machine Learning Research, vol. 119, PMLR, 2020, pp. 4116–4126.
[27] Y. Yu, X. Wang, M. Zhang, N. Liu, C. Shi, Provable Training for Graph Contrastive Learning, Advances in Neural Information Processing Systems, vol. 36, Curran Associates, Inc., 2023, pp. 50327–50345.
[28] S. Khoshraftar, A. An, A survey on graph representation learning methods, ACM Trans. Intell. Syst. Technol. 15 (1) (2024) 19:1–19:55.
[29] S. Cao, W. Lu, Q. Xu, Grarep: learning graph representations with global structural information, in: Proceedings of the 24th ACM International on Conference on Information and Knowledge Management, ACM, 2015, pp. 891–900.
[30] M. Ou, P. Cui, J. Pei, Z. Zhang, W. Zhu, Asymmetric transitivity preserving graph embedding, in: Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, ACM, 2016, pp. 1105–1114.
[31] X. Wang, P. Cui, J. Wang, J. Pei, W. Zhu, S. Yang, Community preserving network embedding, Proc. AAAI Conf. Artif. Intell. 31 (2017) 203–209.
[32] Y. Zhang, H. Wang, J. Zhao, Space-invariant projection in streaming network embedding, Inf. Sci. 649 (2023) 119637.
[33] B. Perozzi, R. Al-Rfou, S. Skiena, Deepwalk: online learning of social representations, in: Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, ACM, 2014, pp. 701–710.
[34] A. Grover, J. Leskovec, node2vec: scalable feature learning for networks, in: Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, ACM, 2016, pp. 855–864.
[35] D. Jin, R. Wang, M. Ge, D. He, X. Li, W. Lin, W. Zhang, RAW-GNN: random walk aggregation based graph neural network, in: IJCAI, ijcai.org, 2022, pp. 2108–2114.
[36] D. Wang, P. Cui, W. Zhu, Structural deep network embedding, in: Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, San Francisco, CA, USA, August 13-17, 2016, ACM, 2016, pp. 1225–1234.
[37] T.N. Kipf, M. Welling, 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, 2017.
[38] W.L. Hamilton, Z. Ying, J. Leskovec, Inductive representation learning on large graphs, in: Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA, 2017, pp. 1024–1034.
[39] R. Pascanu, Ç. Gülçehre, K. Cho, Y. Bengio, How to construct deep recurrent neural networks, in: 2nd International Conference on Learning Representations, ICLR 2014, Banff, AB, Canada, April 14-16, 2014, Conference Track Proceedings, 2014.
[40] P. Goyal, S.R. Chhetri, A. Canedo, dyngraph2vec: capturing network dynamics using dynamic graph representation learning, Knowl.-Based Syst. 187 (2020) 104816.
[41] W. Yu, W. Cheng, C.C. Aggarwal, K. Zhang, H. Chen, W. Wang, 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, KDD 2018, London, UK, August 19-23, 2018, ACM, 2018, pp. 2672–2681.
[42] A. Maheshwari, A. Goyal, M.K. Hanawal, G. Ramakrishnan, Dyngan: generative adversarial networks for dynamic network embedding, in: Graph Representation Learning Workshop at NeurIPS, 2019.
[43] Z. Qiu, W. Hu, J. Wu, W. Liu, B. Du, X. Jia, Temporal network embedding with high-order nonlinear information, in: The Thirty-Fourth AAAI Conference on Artificial Intelligence, AAAI 2020, New York, NY, USA, February 7-12, 2020, AAAI Press, 2020, pp. 5436–5443.
[44] A. van den Oord, Y. Li, O. Vinyals, Representation learning with contrastive predictive coding, CoRR, arXiv:1807.03748 [abs].
[45] J. Stehlé, N. Voirin, A. Barrat, C. Cattuto, L. Isella, J.-F. Pinton, M. Quaggiotto, W. Van den Broeck, C. Régis, B. Lina, et al., High-resolution measurements of face-to-face contact patterns in a primary school, PLoS ONE 6 (8) (2011) e23176.
[46] B. Klimt, Y. Yang, Introducing the enron corpus, in: CEAS 2004 - First Conference on Email and Anti-Spam, July 30-31, 2004, Mountain View, California, USA, 2004.
[47] F. Folino, C. Pizzuti, An evolutionary multiobjective approach for community discovery in dynamic networks, IEEE Trans. Knowl. Data Eng. 26 (8) (2014) 1838–1852.
[48] L. Jure, A. Krevl, Snap datasets: Stanford large network dataset collection, Retrieved December 2021 from http://snap.stanford.edu/data.
[49] J. Tang, M. Qu, M. Wang, M. Zhang, J. Yan, Q. Mei, LINE: large-scale information network embedding, in: Proceedings of the 24th International Conference on World Wide Web, WWW 2015, Florence, Italy, May 18-22, 2015, ACM, 2015, pp. 1067–1077.