Information Sciences 681 (2024) 121202

Contents lists available at ScienceDirect

Information Sciences

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

Inductive link prediction on temporal networks through causal inference

Zhiqiang Pan [a], Fei Cai [a,*], Wanyu Chen [b], Taihua Shao [a], Yupu Guo [a], Honghui Chen [a,*]

[a] Science and Technology on Information Systems Engineering Laboratory, National University of Defense Technology, Changsha, Hunan, China
[b] College of Electronic Countermeasures, National University of Defense Technology, Hefei, Anhui, China

ARTICLE INFO

Keywords:
Temporal networks
Inductive link prediction
Causal inference

ABSTRACT

The aim of inductive temporal link prediction is to forecast future edges associated with nodes unseen during training, which is a crucial task in the field of temporal network analysis. Existing methods mainly make predictions by learning from the node/edge attributes or investigating the node substructures. However, the deficiency of attributes limits the application scope of the attribute-aware methods, while the performance of the substructure-aware models is hindered by neglecting the node correlations or introducing structure bias. In addition, current inductive temporal link prediction methods struggle to generalize the learned network evolution pattern across different networks. To address these issues, we propose a Causal LInk Prediction (CLIP) framework for the inductive temporal link prediction task. Specifically, building upon the existing anonymous distance encoding strategy, we propose to eliminate the structure bias for estimating the true distance between nodes, which is achieved by the backdoor adjustment through the *do*-calculus, followed by decoupling the distance encoding vector to approximate the result. Moreover, to better adapt to the realistic scenarios, we further leverage the node substructures by considering the substructure features as the intervention on the basis of the true distance between nodes. In addition, our proposed approach achieves true inductive temporal link prediction by learning the universal evolution pattern across various temporal networks, which is accomplished through training on the synthetic dynamic graph data generated from the powerlaw cluster networks. We conduct extensive experiments on four real-world temporal networks, i.e., SuperUser, MathOverflow, AskUbuntu and StackOverflow, and the experimental results demonstrate that CLIP outperforms the baselines in terms of AP and AUC. In addition, experiments on the synthetic test graph data with various distributions showcase the remarkable generalization ability of CLIP.

1. Introduction

Graph networks have proven to be effective in modeling complex systems by regarding the contained elements as nodes and the interactions between them as edges [43]. Moreover, in real-world applications such as social media [42], recommender systems [37],

* Corresponding authors.
E-mail addresses: panzhiqiang@nudt.edu.cn (Z. Pan), caifei08@nudt.edu.cn (F. Cai), wanyuchen@nudt.edu.cn (W. Chen), shaotaihua13@nudt.edu.cn (T. Shao), guoyupu14@nudt.edu.cn (Y. Guo), chenhonghui@nudt.edu.cn (H. Chen).

https://doi.org/10.1016/j.ins.2024.121202
Received 22 October 2023; Received in revised form 11 July 2024; Accepted 13 July 2024

Available online 17 July 2024

etc, these systems consist of constantly changing nodes and evolving edges on the graph structure, leading to temporal networks [13]. While extensive research has been conducted on static networks [8], predicting the evolution of temporal networks still remains a challenging and important problem [3]. In this paper, we focus on the inductive temporal link prediction task, which aims to forecast the future inductive links associated with nodes unseen during training on temporal networks [41].

Previous methods primarily focus on leveraging the attributes brought by the nodes/edges or exploring the substructures of nodes. For instance, literatures [31,44,46] learn the node representations by mapping the node/edge attributes to the embedding space using non-linear deep learning models, such as graph neural networks (GNNs) [14,36]. On the other hand, the substructure-aware models either simultaneously extract the correlated substructures of the node pair to estimate their distance [41,12] or consider the structure features of nodes by sampling their individual substructures [48].

However, since the attributes of nodes/edges on temporal networks may be missing for various reasons, such as the privacy-protecting policy [19], the application scope of the attribute-aware methods is limited. Moreover, in the substructure-aware methods, the distance-based approaches suffer from the negative effects of structure bias, while the structure feature aware methods overlook the correlations between nodes. In addition, existing inductive temporal link prediction methods mainly focus on learning the network changing and predicting its future evolution within the same temporal network, failing to generalize the learned network evolution pattern across different networks.

To address the aforementioned challenges, we propose a novel framework called Causal LInk Predictor (CLIP) for inductive temporal link prediction. Specifically, building upon the state-of-the-art anonymous distance encoding method [41], we estimate the true distance between nodes by (1) first adopting the backdoor adjustment to remove the structure bias through the do-calculus [27], and (2) then further approximating the theoretical result by decoupling the distance encoding vector. After that, the individual substructure features of nodes are considered as the intervention on the basis of the true node distance, so as to leverage the node substructures for making predictions in realistic scenarios. In addition, to achieve true inductive temporal link prediction, we simulate the temporal networks by generating the synthetic training graph data using the powerlaw cluster graph [9] snapshots, on which our universal causal link predictor can be trained. Finally, the well-trained link predictor can be applied to various real-world temporal networks for generating accurate predictions.

To verify the effectiveness of our proposed CLIP framework, we conduct experiments on real-world temporal networks as well as synthetic test graph data with various distributions. The experimental results demonstrate that CLIP outperforms the baselines in terms of AP and AUC and meanwhile exhibits remarkable generalization ability.

Our contributions in this paper are summarized as follows:

1. We adopt the powerlaw cluster graph snapshots to simulate temporal networks for training, so as to achieve the true inductive temporal link prediction.
2. We utilize the backdoor adjustment to mitigate the structure bias in the existing anonymous distance encoding through the do-calculus, thus extracting the true node distance.
3. We take the individual node substructures into account by considering the substructure features as the intervention for making predictions on realistic temporal networks.
4. We conduct extensive experiments on four real-world temporal networks to prove the superiority of CLIP over the baselines. Additionally, we evaluate CLIP's performance on synthetic test graph data to validate its generalization ability.

The rest of this paper is organized as follows: We first review the literatures related to our research in Section 2. Next, we introduce the preliminaries in Section 3, and then detail our proposal in Section 4. After that, we give the experimental settings in Section 5, and then present the experimental results and conduct deep analysis in Section 6. Finally, we conclude our work and point out several possible future directions in Section 7.

2. Related work

In this section, we introduce the literatures related to our research mainly from two categories, i.e., temporal link prediction (see Section 2.1) and causal inference (see Section 2.2).

2.1. Temporal link prediction

According to the difference between the training and test data, the temporal link prediction task can be divided into transductive temporal link prediction and inductive temporal link prediction [13].

2.1.1. Transductive temporal link prediction

Transductive temporal link prediction aims to make predictions for future links within the test set which share the same nodes as the training set [15,10]. The typical transductive temporal link prediction methods mainly focus on learning the dynamic network embeddings [50,7,31,45]. For example, Zhou et al. [50] propose to enable the joint learning of embedding vectors for each vertex at different time steps by utilizing triads [11] to model dynamic changes in network structures. And Singer et al. [32] propose to preserve the static network embeddings through initialization and learn the network dynamics by combining the historical temporal embeddings, respectively. Moreover, Hajiramezanali et al. [7] propose a novel hierarchical variational model that extends the graph recurrent neural network (GRNN) [17] to capture the dynamics of both topology and node attributes in dynamic graphs by introducing

additional latent random variables to jointly model the hidden states of GRNN. Wang et al. [38] investigate the application of continual learning to incremental graph learning to tackle the challenges of data distribution shift and new pattern appearance over time. Furthermore, Sankar et al. [31] learn the node representations capturing the evolving graph structure dynamics through a combination of self-attention mechanisms [35] that operate on both the structural neighborhood and temporal dynamics. In addition, Yang et al. [45] address the problem of few-shot link prediction in dynamic networks and introduce a meta-learner that leverages hierarchical time interval-wise and node-wise adaptations to extract general knowledge for this task.

2.1.2. Inductive temporal link prediction

Inductive link prediction focuses on predicting the future links associated with the nodes newly appearing in the test set compared with the training set [5], where the main approaches concentrate on the node/edge features or the substructures of nodes [44,41,18]. For example, traditional methods such as Adamic-Adar (AA) [1] and preferential attachment (PA) [25] rely on the common neighbor mechanism to make predictions, which assumes that links are more likely to be created between nodes sharing many common neighbors. Moreover, Trivedi et al. [34] propose to consider the evolving dynamics of the communication and association processes, producing the embeddings capturing both temporal and structural information. Based on the self-attention mechanism, Xu et al. [44] propose the temporal graph attention (TGAT) layer, which efficiently aggregates features from the temporal-topological neighborhood and learns the interactions between time and features. And Rossi et al. [29] propose temporal graph networks (TGNs), which guarantee both high performance and computational efficiency by combining the memory modules and the graph-based operators. In addition, Wang et al. [41] propose the causal anonymous walk (CAW) which represents the network dynamics by automatically retrieving the network motifs [22] from the temporal random walks and designs an anonymization strategy to replace the node identities for inductively establishing the correlations between motifs. Furthermore, Jin et al. [12] propose the neural temporal walks (NTWs) extend CAW by introducing the spatiotemporal-biased random walks, and further utilize the neural ordinary differential equations [4] to explicitly model the temporal dependencies. In addition, Pan et al. [26] design an adaptive sampling strategy using the pretrained node embeddings, and consider both the structure of nodes and the distance between nodes for predicting future inductive edges.

However, the transductive temporal link prediction methods fail to generate predictions on the future links associated with the newly appearing nodes in the test set compared with the training set, limiting the application scope [34,41]. Moreover, the existing inductive temporal link prediction methods either neglect the correlations between nodes or easily receive the negative effect of the structure bias, disturbing the accurate modeling of the node correlations. In addition, previous inductive methods focus on the inductive setting which learns the network changing and infers the future inductive links within the same temporal network, failing to generalize the learned network evolution pattern across various networks.

2.2. Causal inference

Causal inference [27] is a scientific discipline that seeks to estimate the causal relationships between variables, which has been utilized to deal with the negative bias in extensive applications, including recommender systems [39], privacy protection [33], etc. For example, Wang et al. [39] propose a deconfounded recommender system (DecRS) to address the influence of confounding factors by employing a backdoor adjustment [27] with an approximation operator, and further design an inference strategy to dynamically regulate the backdoor adjustment based on the user's current status. Moreover, Ren et al. [28] analyze the effect of domain shift on deep neural networks (DNNs) induced by attacks using a domain-attack invariant causal model (DICM), and then propose a coherent causal invariant principle to help infer the human-like causal relations.

In addition, causal inference has also been investigated in the research area of link prediction. Specifically, Zhao et al. [49] explore the counterfactual link prediction (CFLP), which introduces a novel data augmentation-based method for generating counterfactual links and learns representations from both the observed and counterfactual links. However, such data augmentation based methods require generating additional counterfactual data, which in turn increases the time consumption cost for both data processing and model optimization, particularly in scenarios involving large-scale data.

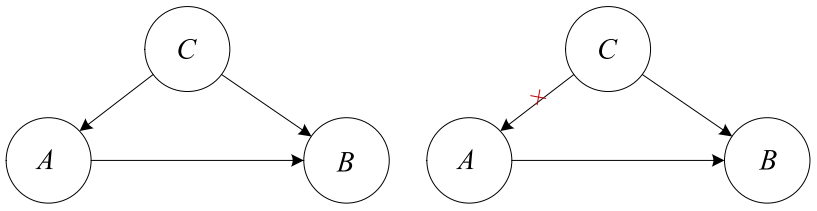
3. Preliminaries

In this section, we detail the preliminary knowledge related to our research, including the problem formulation, the introduction to causal inference and the pipeline of existing methods. Specifically, we first present the formulation of our investigated true inductive temporal link prediction task in Section 3.1. Then, we introduce causal inference, especially structural causal models (SCMs) and do-calculus, which are highly related to our research in Section 3.2. Finally, we provide an overview of the pipeline of the existing state-of-the-art inductive temporal link prediction methods in Section 3.3.

3.1. Problem formulation

Let $\mathcal{G} = \{\mathcal{V}, \mathcal{E}\}$ denote the temporal network, where $\mathcal{V}$ and $\mathcal{E}$ are the contained time-varying nodes and edges, respectively. Each edge $(u, v, t_{uv}) \in \mathcal{E}$ indicates that nodes u and v are connected at the timestamp t_{uv}. The temporal link prediction task aims to learn the temporal network evolution from the training nodes $\mathcal{V}_{tra}$ and edges $\mathcal{E}_{tra}$ in the training graph $\mathcal{G}_{tra} = \{\mathcal{V}_{tra}, \mathcal{E}_{tra}\}$, and then predict the future links $\mathcal{E}_{te}$ that may appear between the test nodes $\mathcal{V}_{te}$ in the test graph $\mathcal{G}_{te} = \{\mathcal{V}_{te}, \mathcal{E}_{te}\}$.

Most existing temporal link prediction methods focus on investigating transductive temporal link prediction where $\mathcal{V}_{te} \subseteq \mathcal{V}_{tra}$, which indicates that all the test nodes appear in the training phase. Moreover, some literatures [44,41,12] take the inductive temporal



(a) An example of the fork structure. (b) do-calculus on the fork structure.

Fig. 1. Examples of the fork structure and do-calculus.

link prediction into consideration, where the test nodes contain newly appearing nodes that are not included in the training set, i.e., $\mathcal{V}_{te} \nsubseteq \mathcal{V}_{tra}$.

However, existing inductive temporal link prediction methods learn the network evolution in a temporal network and then predict the future state within the same network, failing to generalize the learned network evolution pattern across different networks. Consequently, we propose to investigate the true inductive temporal link prediction task by learning the parameters of the universal link predictor from the synthetic training graph data and then predicting the future state of various temporal networks during their corresponding evolution processes.

3.2. Causal inference

3.2.1. Structural causal models

Unlike machine learning which learns the correlation between variables by fitting the training data, causal inference aims to discover the causality between variables. The representative investigated causal framework is the structural causal models (SCMs) proposed by Pearl et al. [27]. Specifically, SCMs build the causal graphs by involving the variables and their causal relationships via a directed acyclic graph (DAG), and then use the structural functions to calculate the value of each variable based on its parent nodes. In addition, there are three types of DAGs, including chain, fork and collider. Here, we introduce the fork structure in detail, considering its high relevance to our research. Fig. 1a presents an example of the fork structure, where variable C is a confounder affecting both variables A and B. When estimating the correlations from variable A to variable B, simply omitting variable C will lead to biased correlation estimation.

3.2.2. do-calculus

To remove the impact of the confounder C on the correlation estimation from variable A to variable B, the optimal method is to conduct the randomized experiments by dividing the individuals into the treatment group and the control group with different values of variable A. However, the randomized experiments are generally expensive, making them hard or even impossible to implement [6]. Thus, it is necessary to estimate the causal correlation from variable A to variable B with only the observed data. A possible solution for achieving this goal is to conduct the intervention on variable A using do-calculus [27]. Specifically, as shown in Fig. 1b, do-calculus blocks the impact of A's parent node C, then the impact of variable A on variable B can be obtained as follows:

$$P(B|\mathrm{do}(C)) = \sum_{A} P(B|C,A)P(A), \tag{1}$$

where the causal correlation of variable A on variable B can be obtained as the weighted sum of the conditioned causal effect. The application of do-calculus to our research problem can be referred in Section 4.3 for more details.

3.3. Inductive temporal link prediction pipeline

The existing state-of-the-art inductive temporal link prediction pipeline is mainly constituted of three components, i.e., temporal substructure extraction (see Section 3.3.1), link probability calculation (see Section 3.3.2, implemented by anonymous distance encoding and aggregation function), and model optimization (see Section 3.3.3).

3.3.1. Temporal substructure extraction

Given the source node u and target node v, the neighbors of u and v can be sampled to extract their corresponding substructures. Specifically, the neighbors are sampled by backtracking the edge timestamps, i.e., merely the neighbors appearing before the current timestamp can be sampled. Then, for $v_x \in \{u, v\}$, we can obtain various temporal walks on the graph structure, forming the substructure of v_x with different hops of neighbors, which can be denoted as $\mathcal{N}_{v_x} = \{\mathcal{N}^0_{v_x}, \mathcal{N}^1_{v_x}, \ldots, \mathcal{N}^L_{v_x}\}$. And the neighbors at the l-hop can be represented as $\mathcal{N}^l_{v_x} = \{v^l_1, v^l_2 \ldots, v^l_M\}$, where M is the neighbor number at each layer. In addition, following reference [41], the neighbors $\mathcal{N}^0_{v_x}$ at the 0-hop are all padded using the anchor node v_x itself.

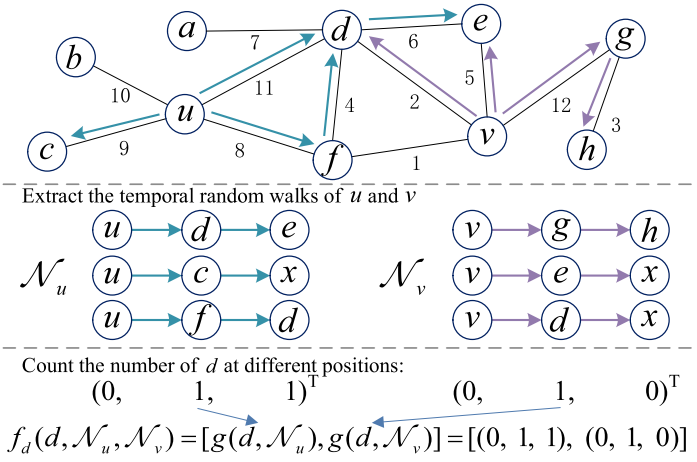


Fig. 2. The anonymous distance encoding of an example neighbor d, where x indicates the padded neighbor.

3.3.2. Link probability calculation

After extracting the substructure of u and v as $\mathcal{N}_u$ and $\mathcal{N}_v$, respectively, the probability of a future link formulation between u and v can be estimated by encoding nodes u and v as well as their corresponding neighbors of multihops. By denoting the link predictor as the function Φ, which can be implemented by modeling the graph structure [48], measuring the distance between nodes [41,12], considering the time features [44], and so on, the link probability calculation process can be formulated as follows:

$$y_{uv} = \Phi(u, v, \mathcal{N}_u, \mathcal{N}_v), \tag{2}$$

where $y_{uv} \in \mathbb{R}$ is the predicted score between nodes u and v.

Specifically, to achieve this goal, the state-of-the-art link predictor focuses on estimating the distance between nodes on the temporal graph structure in two steps, i.e., anonymous distance encoding [41] and aggregation function.

Anonymous distance encoding The anonymous distance encoding focuses on encoding each neighbor of nodes u or v, i.e., $v_i \in \mathcal{N}_u \cup \mathcal{N}_v$, to measure the distance between u and v by regrading each neighbor v_i as an intermediate node. Specifically, for neighbor v_i, the encoding process can be formulated as follows:

$$\mathbf{a}_i = [f_d(v_i, u), f_d(v_i, v)] = [g(v_i, \mathcal{N}_u), g(v_i, \mathcal{N}_v)], \tag{3}$$

where $\mathbf{a}_i \in \mathbb{R}^{2(L+1)}$ is the distance encoding vector of v_i. And for $v_x \in \{u, v\}$, $f_d(v_i, v_x)$ measures the overall distance between neighbor v_i and node v_x, which is achieved by $g(v_i, \mathcal{N}_{v_x})$ that calculates the number of v_i appearing at positions with different steps away from v_x (i.e., at different hops in the node substructure of v_x). This can be formulated as follows:

$$g(v_i, \mathcal{N}_{v_x})[l] = |\{v_j | v_j \in \mathcal{N}^l_{v_x}, v_i = v_j\}|, \forall l = 0, 1, \ldots, L, \tag{4}$$

where v_j is a neighbor at the l-th hop, and $g(v_i, \mathcal{N}_{v_x}) \in \mathbb{R}^{L+1}$ denotes the vector measuring the distance between v_i and v_x.

We present an example of the anonymous distance encoding in Fig. 2 by illustrating the encoding process of neighbor d. As shown in Fig. 2, $g(d, \mathcal{N}_u)$ reflects the distribution of neighbor d appearing at positions with various hops away from the source node u, which provides an overall estimation of the distance between u and d. Similarly, $g(d, \mathcal{N}_v)$ measures the distance between the target node v and neighbor d. Thus, by regarding neighbor d as the intermediate node, we can achieve the goal of estimating the distance between the source node u and target node v in the temporal graph structure.

Aggregation function The aggregation function, which we denote as f_θ, aims to fuse the distance information measured by neighbors at various hops to provide an overall estimation of the distance between nodes u and v. Specifically, the distance encoding vector $\mathbf{a}_i$ of neighbor $v_i \in \mathcal{N}_u \cup \mathcal{N}_v$ is first inputted into a multilayer perceptron (MLP) to obtain v_i's distance representation as follows:

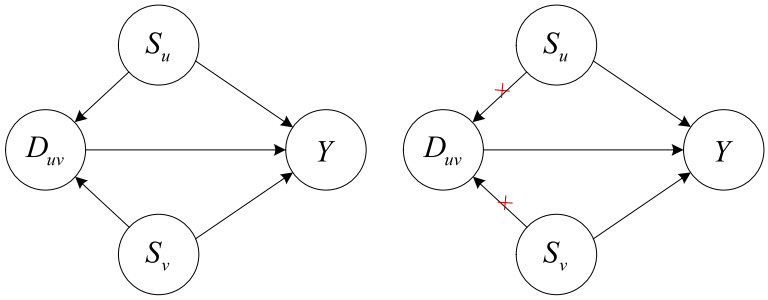
$$\mathbf{d}_i = \text{MLP}(\mathbf{a}_i) = \mathbf{W}_2(\sigma(\mathbf{W}_1(\mathbf{a}_i))), \forall v_i \in \mathcal{N}_{v_x}, \tag{5}$$

where $\mathbf{d}_i \in \mathbb{R}^d$ is the transformed vector of v_i, d is the vector dimension, $\mathbf{W}_1 \in \mathbb{R}^{d \times 2(L+1)}$ and $\mathbf{W}_2 \in \mathbb{R}^{d \times d}$ are the trainable parameters, and σ denotes the ReLu function.

After that, we fuse the distance measured by various neighbors using the graph neural networks (GNNs) to generate the final distance representation of $v_x \in \{u, v\}$ as follows:

$$\mathbf{d}_{v_x} = f_{GNN}(\mathbf{d}_i | v_i \in \mathcal{N}_{v_x}), \tag{6}$$

where the final distance representation of u and v can be represented as $\mathbf{d}_u \in \mathbb{R}^d$ and $\mathbf{d}_v \in \mathbb{R}^d$, respectively.



(a) Conventional inductive temporal link prediction. (b) Deconfounded inductive temporal link prediction.

Fig. 3. Causal graphs for inductive temporal link prediction.

Finally, the probability $y_{uv} \in \mathbb{R}$ of a future link formation between nodes u and v can be generated by inputting their representations $\mathbf{d}_u$ and $\mathbf{d}_v$ into another MLP as follows:

$$y_{uv} = \text{MLP}([\mathbf{d}_u, \mathbf{d}_v]) = \mathbf{W}_4(\sigma(\mathbf{W}_3([\mathbf{d}_u, \mathbf{d}_v]))), \tag{7}$$

where $\mathbf{W}_3 \in \mathbb{R}^{d \times d}$ and $\mathbf{W}_4 \in \mathbb{R}^{1 \times d}$ are the trainable parameters.

3.3.3. Model optimization

Finally, we employ the cross-entropy function as the optimization objective for optimizing the learnable parameters in the link predictor function Φ as follows:

$$L = \sum_{(u,v,t_{uv}) \in \mathcal{E}_{tra}} -\log(\sigma(y_{uv})) - \log(\sigma(1 - y_{uv'})), \tag{8}$$

where $(u, v, t_{uv}) \in \mathcal{E}_{tra}$ is the temporal edge observed in the training set, v' indicates the negative sample, and y_{uv} and $y_{uv'}$ are the prediction scores of the positive edge (u, v, t_{uv}) and the negative edge $(u, v', t_{uv'})$, respectively. Finally, we apply the Back-Propagation Through Time (BPTT) algorithm [30] to learn the trainable parameters in Φ.

4. Approach

In this section, we detail our proposal by performing causal analysis on the conventional methods and subsequently adopt the backdoor adjustment as well as the causal intervention to improve the performance of the true inductive temporal link prediction. Specifically, we first construct the causal graphs for inductive temporal link prediction in Section 4.1. Then, we analyze the impact of the structure bias on the conventional methods in Section 4.2. After that, we propose our Causal LInk Predictor (CLIP) model in Section 4.3.

4.1. Causal view of inductive temporal link prediction

To alleviate the structure bias involved in inductive temporal link prediction, we first construct a causal graph for analyzing the relations between the elements in conventional inductive temporal link prediction methods as shown in Fig. 3a. Specifically, the causal graph for conventional methods consists of four variables, i.e., D_{uv}, S_u, S_v and Y, whose meanings are illustrated as follows:

- D_{uv} denotes the true distance between the source node u and the target node v;
- S_u and S_v are the substructures of nodes u and v extracted from the temporal network, respectively;
- Y denotes the existence of a future link (u, v, t_{uv}) between nodes u and v at the timestamp t_{uv}, i.e., 1 or 0.

In addition, the edges in the causal graph denote the causal relations between the corresponding variables. In detail,

- $\{D_{uv}, S_u, S_v\} \rightarrow Y$ denotes that the link formulation probability is simultaneously decided by the true distance between u and v as well as influenced by the substructures of u and v;
- $\{S_u, S_v\} \rightarrow D_{uv}$ denotes that the substructures of nodes u and v impact the true distance estimation between them.

Based on the causal theory [27], S_u and S_v are identified as the confounding variables between D_{uv} and Y. Specifically, there are two causal paths starting from $\{S_u, S_v\}$ and ending at Y, i.e., $\{S_u, S_v\} \rightarrow Y$ and $\{S_u, S_v\} \rightarrow D_{uv} \rightarrow Y$. The first path indicates that in addition to the true distance D_{uv} between nodes, the node substructures affect the link formulation probability Y. Moreover, in the second path, the substructures S_u and S_v first influence the distance and then indirectly affect the link formulation probability Y.

This makes the estimation of the impact of D_{uv} on Y suffer from the structure bias introduced by S_u and S_v. For example, as stated in reference [47], compared with nodes with less structural complexity, nodes with abundant substructure tend to have a shorter distance to other nodes.

4.2. Causal analysis on conventional methods

Due to existing confounders S_u and S_v, previous conventional inductive temporal link prediction methods suffer from the bad effects of structure bias. Specifically, the conditional probability $P(Y|D_{uv})$ can be calculated as follows:

$$\begin{aligned} & P(Y|D_{uv}) \\ & \overset{(1)}{=} \sum_{S_u,S_v} P(Y,S_u,S_v|D_{uv}), \\ & \overset{(2)}{=} \sum_{S_u,S_v} P(Y|D_{uv},S_u,S_v)P(S_u,S_v|D_{uv}), \\ & \overset{(3)}{=} \sum_{S_u,S_v} P(Y|D_{uv},S_u,S_v)P(D_{uv}|S_u,S_v)P(S_u,S_v), \end{aligned} \tag{9}$$

where (1) is the definition of marginal distribution; (2) and (3) are due to Bayes' theorem.

From the theoretical result, we can intuitively observe that the term $P(D_{uv}|S_u,S_v)$ indicates that the estimation of the influence of the true node distance D_{uv} on the link prediction probability Y will receive the bias from the confounder, i.e., the substructures S_u and S_v, resulting in an inaccurate estimation of the influence of D_{uv} on Y.

4.3. Causal LInk predictor (CLIP)

To address the existing structural bias in conventional methods, we first estimate the true distance between nodes using the backdoor adjustment in Section 4.3.1, where the results are further approximated in Section 4.3.2. Finally, on the basis of the obtained true node distance, we further consider the node substructures using the structure intervention for making accurate predictions in real-world temporal networks in Section 4.3.3.

4.3.1. Backdoor adjustment

To address the influence of the confounding variables S_u and S_v, we propose to estimate the causal effect of the true node distance D_{uv} on the prediction score Y. Theoretically, such an aim can be achieved through the experiments by collecting intervened data and forcibly adjusting the distance between nodes to eliminate the confounding effect. However, implementing such experiments is expensive, especially in large scale scenarios. Moreover, this also poses a potential risk of negatively impacting the user experience.

Thus, we rely on the causal technique known as backdoor adjustment [27], which allows for the estimation of the causal effect using the observed data. Specifically, we perform the do-calculus [27] on the distance D_{uv}, which removes the impact of D_{uv}'s parent nodes, achieving an accurate estimation of the impact of D_{uv} on Y. The intervened causal graph is presented in Fig. 3b. Denoting the causal graphs in Fig. 3a and Fig. 3b as G and G', respectively, the influence of D_{uv} on Y can be calculated as follows:

$$\begin{aligned} & P(Y|do(D_{uv})) \\ & \overset{(1)}{=} P_{G'}(Y|D_{uv}), \\ & \overset{(2)}{=} \sum_{S_u,S_v} P_{G'}(Y|D_{uv},S_u,S_v)P_{G'}(S_u,S_v|D_{uv}), \\ & \overset{(3)}{=} \sum_{S_u,S_v} P_{G'}(Y|D_{uv},S_u,S_v)P_{G'}(S_u,S_v), \\ & \overset{(4)}{=} \sum_{S_u,S_v} P(Y|D_{uv},S_u,S_v)P(S_u,S_v), \end{aligned} \tag{10}$$

where (1) is due to the backdoor criterion as the only backdoor path $D_{uv} \leftarrow \{S_u,S_v\} \rightarrow Y$ has been blocked by $do(D_{uv})$; (2) is due to Bayes' theorem; (3) is because D_{uv} is independent with S in G'; (4) $P_{G'}(Y|D_{uv},S_u,S_v) = P(Y|D_{uv},S_u,S_v)$ is because the causal mechanism $\{D_{uv},S_u,S_v\} \rightarrow Y$ is not changed when cutting off $\{S_u,S_v\} \rightarrow D_{uv}$, and $P(S_u,S_v) = P_{G'}(S_u,S_v)$ since $\{S_u,S_v\}$ has the same prior on G and G'.

Compared with Eq. (9), we can observe that Eq. (10) lacks the bias term $P(D_{uv}|S_u,S_v)$, which indicates that the structure bias from S_u and S_v to D_{uv} is blocked by the do-calculus. Thus, Eq. (10) can provide an accurate estimation of the impact of D_{uv} on Y. In addition, the same conclusion can also be intuitively observed from Fig. 3b, where we can see that the path $\{S_u,S_v\} \rightarrow D$ is cut by the do-calculus.

4.3.2. Backdoor adjustment approximation

From Eq. (10), we can accurately obtain the impact of D_{uv} on Y. However, there are no known values of Y subject to different substructures of S_u and S_v in the observed data, making the calculation of Eq. (10) intractable. Thus, it is essential to conduct an approximation of Eq. (10) using the observed data. Specifically, building upon the state-of-the-art anonymous distance encoding method introduced in Section 3.3.2, the approximation can be formulated as follows:

$$
\begin{aligned}
&P(Y|do(D_{uv})) \\
&\overset{(1)}{=} \sum_{S_u,S_v} P(Y|D_{uv},S_u,S_v)P(S_u,S_v), \\
&\overset{(2)}{=} \sum_{S_u,S_v} f_\theta([f_d(\mathcal{N}_u,u),f_d(\mathcal{N}_u,v)]+[f_d(\mathcal{N}_v,u),f_d(\mathcal{N}_v,v)])P(S_u,S_v), \\
&\overset{(3)}{=} \sum_{S_u,S_v} f_\theta([f_d(\mathcal{N}_u,u),\mathbf{0}]+[\mathbf{0},f_d(\mathcal{N}_u,v)]+[f_d(\mathcal{N}_v,u),\mathbf{0}]+[\mathbf{0},f_d(\mathcal{N}_v,v)])P(S_u,S_v), \\
&\overset{(4)}{=} \sum_{S_u,S_v} f_\theta([f_d(\mathcal{N}_u,u),\mathbf{0}]+[\mathbf{0},f_d(\mathcal{N}_{uv},v)]+[f_d(\mathcal{N}_{uv},u),\mathbf{0}]+[\mathbf{0},f_d(\mathcal{N}_v,v)])P(S_u,S_v), \\
&\overset{(5)}{=} \sum_{S_u,S_v} f_\theta([f_d(\mathcal{N}_{uv},u),f_d(\mathcal{N}_{uv},v)]+f_d(\mathcal{N}_u,u)+f_d(\mathcal{N}_v,v))P(S_u,S_v), \\
&\overset{(6)}{=} \sum_{S_u,S_v} f_\theta([f_d(\mathcal{N}_{uv},u),f_d(\mathcal{N}_{uv},v)])P(S_u,S_v)+\sum_{S_u} f_\theta(f_d(\mathcal{N}_u,u))P(S_u) \\
&\quad +\sum_{S_v} f_\theta(f_d(\mathcal{N}_v,v))P(S_v), \\
&\overset{(7)}{\approx} f_\theta([f_d(\mathcal{N}_{uv},u),f_d(\mathcal{N}_{uv},v)])+C_u+C_v.
\end{aligned}
\tag{11}
$$

We explain this derivation step by step as follows:

- (1) is obtained from the backdoor adjustment, i.e., Eq. (10);
- (2) is because the impact of the biased distance considering D_{uv} as well as S_u and S_v, can be modeled by the distance estimation detailed in Section 3.3.2. Here, $f_d(\mathcal{N}_u,u)$ indicates the overall distance between $\mathcal{N}_u$ and u, which fuses the distance between each neighbor $v_i \in \mathcal{N}_u$ and node u measured by the function $f_d(v_i,u)$ described by Eq. (3), and the illustrations are similar for $f_d(\mathcal{N}_u,v)$, $f_d(\mathcal{N}_v,u)$ and $f_d(\mathcal{N}_v,v)$;
- (3) indicates that under the condition f_θ is linear, the encoding vectors $[f_d(\mathcal{N}_u,u),f_d(\mathcal{N}_u,v)]$ for node u can be separated into two parts $[f_d(\mathcal{N}_u,u),\mathbf{0}]$ and $[\mathbf{0},f_d(\mathcal{N}_u,v)]$, which measure the distance between u's neighbors $\mathcal{N}_u$ and nodes u and v, respectively, and a similar operation can also be conducted on node v by separating $[f_d(\mathcal{N}_v,u),f_d(\mathcal{N}_v,v)]$ to $[f_d(\mathcal{N}_v,u),\mathbf{0}]$ and $[\mathbf{0},f_d(\mathcal{N}_v,v)]$;
- (4) is because when generating the distance encoding vectors for node u, on the side of node v, $[\mathbf{0},f_d(\mathcal{N}_u,v)]$ can be simplified to $[\mathbf{0},f_d(\mathcal{N}_{uv},v)]$ considering that only the common neighbors $\mathcal{N}_{uv}$ of nodes u and v contribute to the encoding while the other positions in $f_d(\mathcal{N}_u,v)$ related to the neighbor set $\mathcal{N}_u-\mathcal{N}_{uv}$ are all zero;
- (5) is because the two elements $[f_d(\mathcal{N}_{uv},u),\mathbf{0}]$ and $[\mathbf{0},f_d(\mathcal{N}_{uv},v)]$ respectively estimating the distance of the common neighbors $\mathcal{N}_{uv}$ from nodes u and v can be combined as $[f_d(\mathcal{N}_{uv},u),f_d(\mathcal{N}_{uv},v)]$, and the zero matrices in $[f_d(\mathcal{N}_u,u),\mathbf{0}]$ and $[\mathbf{0},f_d(\mathcal{N}_v,v)]$ can be omitted;
- (6) is because S_v is independent of $f_\theta(f_d(\mathcal{N}_u,u))$, and S_u is independent of $f_\theta(f_d(\mathcal{N}_v,v))$, respectively;
- (7) is because no observed data about various substructures of nodes u and v are available, thus we regard the observed value $f_\theta([f_d(\mathcal{N}_{uv},u),f_d(\mathcal{N}_{uv},v)])$ to approximate its unbiased value, and the expectation of variables $f_\theta(f_d(\mathcal{N}_u,u))$ and $f_\theta(f_d(\mathcal{N}_v,v))$ are both constants, which we denote as C_u and C_v, respectively.

Notably, there are two limitations to the above derivations: (1) In steps 3-6, the reorganization of the anonymous distance encoding vector requires that the aggregation function f_θ be linear. However, non-linear functions with strong modeling ability can also be adopted by controlling the resulted error to a small value as analyzed in reference [39]. Here, we adopt the linear functions, i.e., removing the non-linear function σ in Eq. (5) as well as Eq. (7), and implement the f_{GNN} in Eq. (6) using GraphSAGE with the mean pooling. We would like to leave the investigation of applying non-linear functions to our future work. (2) In step 7, approximating the expectation of $f_\theta([f_d(\mathcal{N}_{uv},u),f_d(\mathcal{N}_{uv},v)])$ by adopting its observed value introduces bias. However, considering that the common neighbor number of the source node u and target node v is generally far smaller than the total neighbor numbers in their whole node substructures $\mathcal{N}_u$ and $\mathcal{N}_v$ as stated in reference [40], the introduced bias is limited. Generally, considering that no various substructures of nodes u and v are observed, this approximate estimation with a small bias incorporated is practical in real-world scenarios.

4.3.3. Structure intervention

By adopting the backdoor adjustment through the do-calculus to remove the structure bias in the anonymous distance encoding and approximating the result through decoupling the encoding vector, we can obtain the true distance between the source node u and target node v. Then, to guarantee accurate predictions of future inductive links on real-world temporal networks, we propose to take the substructures of nodes u and v into consideration.

Specifically, assuming the influence score of the structure bias of S_u and S_v on the link prediction probability is $\tilde{s}_u$ and $\tilde{s}_v$, respectively, we aim to make predictions with such bias. To achieve this target, here we do the intervention of $S_u = \tilde{s}_u$ and $S_v = \tilde{s}_v$ for prediction as follows:

$$\begin{aligned} &P(Y|do(D_{uv} = d_{uv}), do(S_u = \tilde{s}_u, S_v = \tilde{s}_v)), \\ &= P(Y = 1|d_{uv}, \tilde{s}_u, \tilde{s}_v), \\ &= d_{uv} + \tilde{s}_u + \tilde{s}_v, \end{aligned} \tag{12}$$

where d_{uv} can be obtained by Eq. (11), $\tilde{s}_u$ and $\tilde{s}_v$ denote the influence scores of the structure bias of nodes u and v, respectively, which we propose to model using the graph neural networks (GNNs). Specifically, we refer to the degrees of nodes in the node substructure for modeling the fine-grained substructure features of $v_x \in \{u, v\}$. In detail, we first embed the degree deg_i of each neighbor $v_i \in \mathcal{N}_{v_x}$ as follows:

$$\mathbf{d}_i = \text{Embedding}(deg_i), \tag{13}$$

where $\mathbf{d}_i \in \mathbb{R}^d$ is the generated embedding vector, d is the embedding dimension, and deg_i is the degree of neighbor v_i.

Then, we adopt GNNs to fuse the embeddings of multihop neighbors for obtaining the final structure representation of node v_x as $\mathbf{d}_{v_x} \in \mathbb{R}^d$ as follows:

$$\mathbf{d}_{v_x} = f_{GNN}(\mathbf{d}_i | v_i \in \mathcal{N}_{v_x}), \tag{14}$$

where various GNNs such as GCN [14] and GAT [36] can all be adopted, here we apply GraphSAGE with the mean pooling for simplification to simultaneously avoid introducing additional parameters and accelerate the training as well as inference processes.

Finally, we can generate the influence score of the structure bias of v_x on the link prediction probability as $\tilde{s}_{v_x} \in \mathbb{R}$ by inputting $\mathbf{d}_{v_x}$ into an MLP as follows:

$$\tilde{s}_{v_x} = \text{MLP}(\mathbf{d}_{v_x}) = \mathbf{W}_6(\sigma(\mathbf{W}_5(\mathbf{d}_{v_x}))), \tag{15}$$

where $\mathbf{W}_5 \in \mathbb{R}^{d \times d}$ and $\mathbf{W}_6 \in \mathbb{R}^{1 \times d}$ are the trainable parameters, and σ denotes the ReLu function.

5. Experiments

In this section, we first list the research questions to guide the experiments in Section 5.1, and then introduce the used datasets in Section 5.2. After that, the models discussed in this article are summarized in Section 5.3. Finally, the experimental setup is given in Section 5.4.

5.1. Research questions

(RQ1) Can our proposed CLIP model achieve better performance than the baselines on real-world temporal networks?
(RQ2) How is the generalization ability of CLIP, and can CLIP consistently exhibit strong performance on the generated synthetic test graph data with various distributions?
(RQ3) How is the impact of the non-linearities in CLIP on prediction performance?
(RQ4) How does CLIP perform across different time periods of temporal networks, and can CLIP accurately estimate the true distance between nodes?
(RQ5) What is the impact of the node substructure size on the performance of CLIP?

5.2. Datasets

To train the CLIP model, we generate the synthetic training graph data, which consists of snapshots of the powerlaw cluster graphs with the powerlaw degree distribution and approximate average clustering as outlined in reference [9]. Specifically, during the synthetic graph generation, we set the probability of adding a triangle after adding a random edge (i.e., p) and the number of random edges to add for each new node (i.e., m) to 0.4 and 4, respectively. In addition, we simulate the evolution of realistic temporal networks by adopting a graph snapshot number of 10 and setting the node number in the i-th snapshot to ie^3.

Table 1
Statistics of the real-world datasets used in our experiments.

Statistics	SuperUser	MathOverflow	AskUbuntu	StackOverflow
# Nodes	14,912	4,563	8,374	17,181
# Edges	156,205	100,705	72,919	265,001
Ave Degree	10.48	22.07	8.71	15.42
Timespan	360 days	360 days	360 days	90 days

To examine the utility of CLIP, we adopt four real-world temporal networks collected from the stack exchange websites, i.e., SuperUser,[1] MathOverflow,[2] AskUbuntu[3] and StackOverflow.[4] Moreover, we extract the data for a timespan of 360 days from SuperUser, MathOverflow and AskUbuntu for experiments, while the timespan for StackOverflow is set to 90 days considering its extremely large size. The statistics of the preprocessed real-world datasets are summarized in Table 1. In addition, the test data of each day in the three datasets SuperUser, MathOverflow as well as AskUbuntu and that of every 6 h in StackOverflow are defined as a time period, thus generating 360 time periods on all four datasets. Furthermore, on each dataset, we generate three different snapshot-based temporal networks by ranging the snapshot timespan in 30 periods, 7 periods and 1 period (producing 12, approximately 51 and 360 snapshots, respectively), and simultaneously introduce the original graph data where the temporal edges appear continuously.

In addition, to test the generalization of CLIP, we also include the synthetic test graph data with various distributions for performance evaluation in experiments. Specifically, the synthetic test graph data are generated in the same way as the training data, except that we range the parameters p (i.e., the probability of adding a triangle after adding a random edge) and m (i.e., the number of random edges to add for each new node) in the synthetic test graph data generation in $\{0.3, 0.4, 0.5\}$ and $\{2, 4, 6\}$, respectively.

5.3. Model summary

The following baselines are selected for comparison with CLIP to validate its effectiveness: (1) CN (Common Neighbor) [23] predicts the link probability between nodes based on the number of common neighbors. (2) AA (Adamic-Adar) [1] considers the degrees of common neighbors on the basis of CN, where larger degrees indicate a higher similarity between nodes. (3) PA (Preferential Attachment) [25] follows the mechanism by which links are more likely to be generated between nodes with larger degrees. (4) LP (Local Path) [24] takes into account the 3-hop paths in measuring the distance between nodes. (5) DE-GNN [16] enhances GNNs by incorporating the structure-related distance encoding as additional node features or for controlling the message aggregation. (6) CAW-N [41] proposes to consider the distance between nodes using the designed anonymous distance encoding strategy for making predictions. (7) In addition, we include a variant denoted as CLIP_D of CLIP for performance comparison in RQ2 and RQ3, which relies on the true distance between nodes estimated by Eq. (11).

5.4. Experimental setup

To achieve the true inductive temporal link prediction, we first generate the synthetic training graph data by combining the snapshots of the powerlaw cluster graphs with increasing sizes. More details about the synthetic data generation are presented in Section 5.2. Then, the links in each generated graph snapshot are randomly separated into the training set and validation set with the proportions of 70% and 30%, which are utilized for learning the model parameters and tuning the hyper-parameters, respectively.

Then, for the heuristic baselines, i.e., CN, AA, PA and LP, considering that no trainable parameters are contained, we directly follow their settings by adopting the full historical neighbors for CN, AA as well as PA and measuring the 3-hop paths for LP, respectively. Moreover, for the deep learning based methods, i.e., two neural baselines PGNN and CAW as well as our proposal CLIP, we adopt the Adam optimizer for learning the trainable parameters with setting the learning rate to $1e^{-4}$ for all three models. And we set the batch size and the embedding dimension to 64 and 100, respectively. Furthermore, we range the hop number and the hop length in the node substructure extraction in $\{4, 8, 16, 32\}$ and $\{1, 2, 3, 4\}$ for finding the optimal hyper-parameter combination. In addition, we adopt the early stopping strategy during training, which adopts a total training epoch number of 20 and stops training when the performance on the validation set does not improve for more than 3 epochs. Finally, the learned model with the highest AUC metric on the validation set is selected for testing the prediction performance on the test set. All the experiments are conducted using PyTorch 1.8.1 on a Ubuntu 18.04 server with NVIDIA GeForce RTX 3090 GPU with 24 GB memories.

In addition, to facilitate the reproducibility of the results in our work, we will release the code and data used to run the experiments in this paper after publication.

[1] https://snap.stanford.edu/data/sx-superuser.html.
[2] https://snap.stanford.edu/data/sx-mathoverflow.html.
[3] https://snap.stanford.edu/data/sx-askubuntu.html.
[4] https://snap.stanford.edu/data/sx-stackoverflow.html.

Table 2
Overall performance of CLIP and the baselines in terms of AP and AUC (mean in percentage over 5 trial runs). The results of the best performing baseline and the best performer in each column are underlined and boldfaced, respectively.

	Method	SuperUser		MathOverflow		AskUbuntu		StackOverflow	
		AP	AUC	AP	AUC	AP	AUC	AP	AUC
30-period	CN	$58.43_{\pm 0.0302}$	$56.92_{\pm 0.0216}$	$63.85_{\pm 0.0599}$	$63.16_{\pm 0.0841}$	$59.04_{\pm 0.0534}$	$57.23_{\pm 0.0573}$	$56.35_{\pm 0.0142}$	$55.24_{\pm 0.0227}$
	AA	$61.51_{\pm 0.0193}$	$60.56_{\pm 0.0171}$	$66.76_{\pm 0.0759}$	$65.12_{\pm 0.0845}$	$61.72_{\pm 0.0445}$	$60.39_{\pm 0.0431}$	$58.76_{\pm 0.0184}$	$58.30_{\pm 0.0223}$
	PA	$\underline{63.18_{\pm 0.0000}}$	$56.23_{\pm 0.0000}$	$\underline{68.37_{\pm 0.0000}}$	$61.59_{\pm 0.0000}$	$\underline{62.83_{\pm 0.0000}}$	$54.27_{\pm 0.0000}$	$\underline{64.06_{\pm 0.0000}}$	$58.99_{\pm 0.0000}$
	LP	$62.10_{\pm 0.0385}$	$\underline{61.75_{\pm 0.0478}}$	$66.92_{\pm 0.0583}$	$\underline{66.17_{\pm 0.0479}}$	$61.87_{\pm 0.0393}$	$\underline{60.81_{\pm 0.0456}}$	$59.50_{\pm 0.0390}$	$59.47_{\pm 0.0473}$
	PGNN	$59.34_{\pm 0.0435}$	$61.44_{\pm 0.0365}$	$60.25_{\pm 0.0442}$	$63.77_{\pm 0.0265}$	$57.75_{\pm 0.0708}$	$59.95_{\pm 0.0680}$	$58.15_{\pm 0.0591}$	$\underline{59.71_{\pm 0.0525}}$
	CAW	$56.12_{\pm 0.0557}$	$55.26_{\pm 0.0707}$	$56.95_{\pm 0.0640}$	$56.21_{\pm 0.0568}$	$55.52_{\pm 0.0906}$	$54.79_{\pm 0.1336}$	$53.47_{\pm 0.0583}$	$53.87_{\pm 0.0438}$
	CLIP	$\mathbf{69.50}_{\pm 0.0497}$	$\mathbf{71.36}_{\pm 0.0540}$	$\mathbf{71.74}_{\pm 0.0232}$	$\mathbf{74.89}_{\pm 0.0237}$	$\mathbf{73.79}_{\pm 0.0351}$	$\mathbf{73.86}_{\pm 0.0309}$	$\mathbf{73.12}_{\pm 0.0400}$	$\mathbf{72.69}_{\pm 0.0332}$
7-period	CN	$59.81_{\pm 0.0326}$	$58.49_{\pm 0.0534}$	$65.59_{\pm 0.0170}$	$65.31_{\pm 0.0222}$	$60.90_{\pm 0.0395}$	$59.11_{\pm 0.0688}$	$57.21_{\pm 0.0165}$	$56.24_{\pm 0.0252}$
	AA	$63.28_{\pm 0.0467}$	$62.24_{\pm 0.0544}$	$68.79_{\pm 0.0177}$	$67.12_{\pm 0.0261}$	$64.16_{\pm 0.0448}$	$62.67_{\pm 0.0651}$	$59.82_{\pm 0.0230}$	$59.32_{\pm 0.0256}$
	PA	$\underline{65.25_{\pm 0.0000}}$	$57.96_{\pm 0.0000}$	$\underline{70.58_{\pm 0.0000}}$	$64.51_{\pm 0.0000}$	$\underline{65.19_{\pm 0.0000}}$	$56.17_{\pm 0.0000}$	$\underline{65.48_{\pm 0.0000}}$	$61.02_{\pm 0.0000}$
	LP	$64.03_{\pm 0.0488}$	$\underline{63.84_{\pm 0.0518}}$	$68.98_{\pm 0.0966}$	$\underline{68.60_{\pm 0.0790}}$	$64.41_{\pm 0.0483}$	$\underline{63.43_{\pm 0.0377}}$	$60.75_{\pm 0.0203}$	$60.79_{\pm 0.0223}$
	PGNN	$60.71_{\pm 0.0360}$	$63.33_{\pm 0.0342}$	$61.24_{\pm 0.0259}$	$65.38_{\pm 0.0234}$	$59.46_{\pm 0.0571}$	$62.17_{\pm 0.0630}$	$59.31_{\pm 0.0344}$	$\underline{61.06_{\pm 0.0310}}$
	CAW	$57.05_{\pm 0.0437}$	$56.60_{\pm 0.0345}$	$57.42_{\pm 0.0642}$	$57.16_{\pm 0.0671}$	$56.79_{\pm 0.0709}$	$56.37_{\pm 0.0784}$	$55.24_{\pm 0.0367}$	$56.67_{\pm 0.0279}$
	CLIP	$\mathbf{71.01}_{\pm 0.0364}$	$\mathbf{73.72}_{\pm 0.0209}$	$\mathbf{72.21}_{\pm 0.0487}$	$\mathbf{76.33}_{\pm 0.0292}$	$\mathbf{75.66}_{\pm 0.0342}$	$\mathbf{76.33}_{\pm 0.0326}$	$\mathbf{74.30}_{\pm 0.0344}$	$\mathbf{73.99}_{\pm 0.0208}$
1-period	CN	$61.42_{\pm 0.0203}$	$60.55_{\pm 0.0279}$	$67.41_{\pm 0.0598}$	$67.53_{\pm 0.0906}$	$63.02_{\pm 0.0387}$	$61.78_{\pm 0.0671}$	$57.88_{\pm 0.0207}$	$57.12_{\pm 0.0274}$
	AA	$65.48_{\pm 0.0224}$	$64.46_{\pm 0.0295}$	$71.04_{\pm 0.0639}$	$69.52_{\pm 0.0899}$	$67.14_{\pm 0.0542}$	$65.78_{\pm 0.0703}$	$60.77_{\pm 0.0282}$	$60.31_{\pm 0.0266}$
	PA	$\underline{66.62_{\pm 0.0000}}$	$60.32_{\pm 0.0000}$	$\underline{72.11_{\pm 0.0000}}$	$67.26_{\pm 0.0000}$	$66.92_{\pm 0.0000}$	$58.98_{\pm 0.0000}$	$\underline{66.39_{\pm 0.0000}}$	$\underline{62.61_{\pm 0.0000}}$
	LP	$66.32_{\pm 0.0480}$	$\underline{66.44_{\pm 0.0588}}$	$71.12_{\pm 0.0591}$	$\underline{71.32_{\pm 0.0762}}$	$\underline{67.42_{\pm 0.0563}}$	$\underline{66.98_{\pm 0.0945}}$	$61.92_{\pm 0.0424}$	$62.10_{\pm 0.0490}$
	PGNN	$62.80_{\pm 0.0611}$	$65.82_{\pm 0.0583}$	$62.74_{\pm 0.0911}$	$67.59_{\pm 0.1002}$	$62.07_{\pm 0.0860}$	$65.46_{\pm 0.1010}$	$60.54_{\pm 0.0318}$	$62.42_{\pm 0.0468}$
	CAW	$58.32_{\pm 0.0186}$	$57.71_{\pm 0.0591}$	$58.73_{\pm 0.0104}$	$58.35_{\pm 0.0728}$	$58.73_{\pm 0.0387}$	$58.33_{\pm 0.0526}$	$56.44_{\pm 0.0245}$	$58.23_{\pm 0.0360}$
	CLIP	$\mathbf{71.59}_{\pm 0.0427}$	$\mathbf{74.44}_{\pm 0.0069}$	$\mathbf{72.74}_{\pm 0.0713}$	$\mathbf{77.06}_{\pm 0.0514}$	$\mathbf{76.61}_{\pm 0.0691}$	$\mathbf{77.39}_{\pm 0.0911}$	$\mathbf{74.97}_{\pm 0.0415}$	$\mathbf{74.66}_{\pm 0.0263}$
Continuous	CN	$64.23_{\pm 0.0359}$	$64.22_{\pm 0.0529}$	$70.20_{\pm 0.0293}$	$71.13_{\pm 0.0352}$	$66.41_{\pm 0.0608}$	$66.21_{\pm 0.0860}$	$59.37_{\pm 0.0218}$	$58.89_{\pm 0.0189}$
	AA	$69.31_{\pm 0.0395}$	$68.45_{\pm 0.0508}$	$\underline{74.40_{\pm 0.0334}}$	$73.43_{\pm 0.0350}$	$\underline{71.85_{\pm 0.0621}}$	$70.87_{\pm 0.0789}$	$62.70_{\pm 0.0181}$	$62.29_{\pm 0.0167}$
	PA	$67.65_{\pm 0.0000}$	$63.23_{\pm 0.0000}$	$73.26_{\pm 0.0000}$	$70.13_{\pm 0.0000}$	$68.46_{\pm 0.0000}$	$62.89_{\pm 0.0000}$	$\underline{67.19_{\pm 0.0000}}$	$64.61_{\pm 0.0000}$
	LP	$\underline{69.76_{\pm 0.0566}}$	$\underline{70.50_{\pm 0.0666}}$	$73.34_{\pm 0.0848}$	$\underline{74.82_{\pm 0.0767}}$	$71.57_{\pm 0.0650}$	$\underline{72.25_{\pm 0.0738}}$	$64.26_{\pm 0.0511}$	$64.61_{\pm 0.0490}$
	PGNN	$65.78_{\pm 0.0815}$	$69.61_{\pm 0.0704}$	$64.94_{\pm 0.0700}$	$70.86_{\pm 0.0735}$	$66.40_{\pm 0.0651}$	$70.71_{\pm 0.0839}$	$62.98_{\pm 0.0655}$	$\underline{64.99_{\pm 0.0570}}$
	CAW	$59.96_{\pm 0.0514}$	$58.63_{\pm 0.0578}$	$60.59_{\pm 0.0319}$	$59.46_{\pm 0.0405}$	$61.17_{\pm 0.0657}$	$59.93_{\pm 0.1158}$	$56.56_{\pm 0.0399}$	$57.60_{\pm 0.0463}$
	CLIP	$\mathbf{72.87}_{\pm 0.0342}$	$\mathbf{74.91}_{\pm 0.0347}$	$\mathbf{74.11}_{\pm 0.0531}$	$\mathbf{77.72}_{\pm 0.0509}$	$\mathbf{78.22}_{\pm 0.0961}$	$\mathbf{77.62}_{\pm 0.0775}$	$\mathbf{77.18}_{\pm 0.0105}$	$\mathbf{75.68}_{\pm 0.0096}$

6. Results

In this section, we present the experimental results and conduct deep analysis to answer the four research questions proposed in Section 5.1. Specifically, we first give the overall performance of CLIP and the baselines for comparison in Section 6.1. Then, we test the generalization ability of CLIP in Section 6.2. After that, we analyze the impact of the non-linearities in CLIP in Section 6.3. Finally, we analyze the impact of the time period and substructure size on CLIP in Sections 6.4 and 6.5, respectively.

6.1. Overall performance

We test the performance of CLIP and the baselines on four real-world temporal networks with the snapshot timespan of various periods as well as the continuous graph, and present the results in terms of AP and AUC in Table 2. From Table 2, we can first observe that when the timespan decreases from 30 periods to 1 period and further to the continuous one, the performance of all methods consistently increases. This indicates that the more recent the neighbors are included in the node substructure, the more accurately the future inductive temporal link predictions are generated.

Then, for the baselines, we can observe that in the traditional methods, the structure-aware PA and the distance-based LP perform best in terms of AP and AUC, respectively, in most cases on four datasets. This indicates that both the node substructure and the distance between nodes play important roles in the inductive temporal link prediction task. In addition, it can be observed that the state-of-the-art method in traditional inductive temporal link prediction, i.e., CAW, fails to show satisfactory performance under the true inductive temporal link prediction setting. This may be attributed to the fact that the biased estimation of the distance between nodes makes CAW easily overfit the training graph structures, losing the ability to generalize the learned network evolution pattern from the training data to the test data in the true inductive temporal link prediction setting.

In addition, as for our proposed CLIP model, we can see that CLIP consistently achieves the state-of-the-art performance across four datasets, thereby validating its effectiveness in predicting future inductive links within real-world temporal networks. Moreover, we can see that the performance improvement of CLIP above the optimal baseline on four datasets generally decreases with the snapshot timespan decreasing from 30 periods to 1 period and further to the continuous one. For example, on the SuperUser dataset, the improvements on the 30-period, 7-period, 1-period and continuous cases in terms of AUC are 15.56%, 15.48%, 12.04% and 6.26%, respectively. This indicates that compared with the baselines, our proposed CLIP can effectively capture the historical correlations between nodes, lessening the demand for immediate edges on temporal networks.

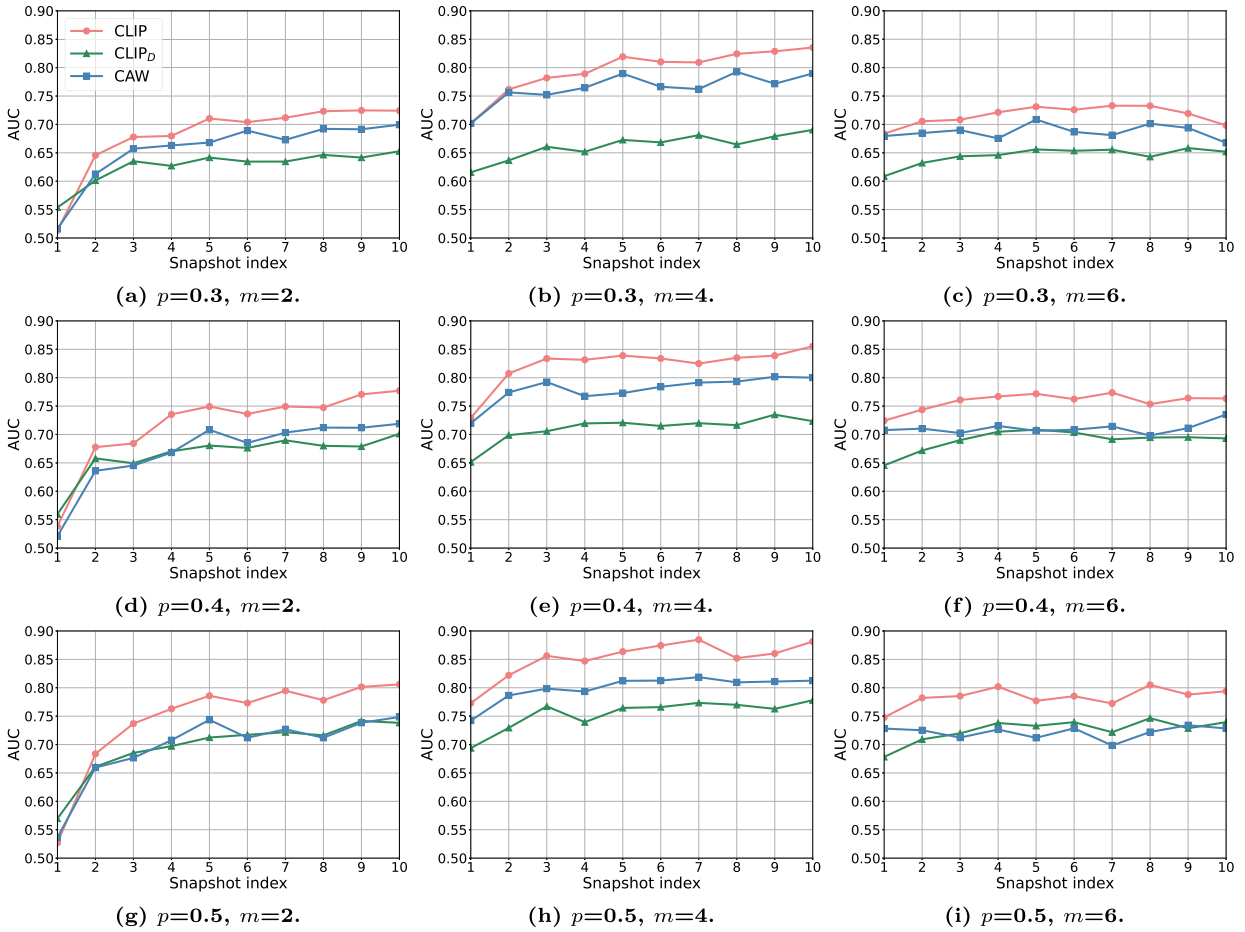


Fig. 4. Results in terms of AUC on the synthetic test graph data with various parameters p and m in the synthetic graph data generation.

6.2. Generalization ability of CLIP

To test the generalization of CLIP, we vary the parameters p and m in the synthetic test graph data generation in $\{0.3, 0.4, 0.5\}$ and $\{2, 4, 6\}$, respectively, for producing the synthetic test graph data with various distributions. The performance of CLIP and the baseline CAW as well as the variant CLIP$_D$ in terms of AUC on different snapshots of each generated synthetic test graph is plotted in Fig. 4.

As shown in Fig. 4, we can first see that the true distance aware method CLIP$_D$ shows better performance than CAW in most cases. This is due to the fact that CAW fails to accurately estimate the true distance between nodes while CLIP$_D$ explicitly models the true distance, which indicates that the true distance estimation plays a critical role in predicting future inductive links on temporal networks. Moreover, it is observed that our proposed CLIP can outperform the state-of-the-art baseline CAW and the variant CLIP$_D$ in various cases on nine generated synthetic test graphs, which is because CLIP simultaneously takes the substructure feature modeling in CAW and the explicit true distance estimation in CLIP$_D$ into the modeling process.

In addition, it can be observed that when fixing m, with the parameter p increasing, the performance of all three models increases. This could be explained by the fact that with p increasing, the node substructures become abundant, which simultaneously contributes to the node substructure feature modeling and the true node distance estimation, leading to accurate predictions of future inductive links. Furthermore, we can observe that when p is fixed, with the parameter m increasing, the performance of the three models first improves with m increasing from 2 to 4, and then generally decreases when m changes from 4 to 6. This may be attributed to the fact that both $m = 2$ and $m = 6$ deviate from the value $m = 4$ in the synthetic training graph data, which indicates that the parameter m has an apparent impact on the generalization ability of methods for true inductive temporal link prediction. Moreover, it can be observed that in addition to the better performance of CLIP than of CAW and CLIP$_D$ on the synthetic test graph sharing the same parameters $p = 0.4, m = 4$ as the synthetic training graph, CLIP also consistently outperforms CAW and CLIP$_D$ on the other generated synthetic test graph data with various p and m. This indicates that CLIP shows strong generalization ability, i.e., the CLIP model trained from a specific data distribution can well generalize to application scenarios with different data distributions, see Fig. 5.

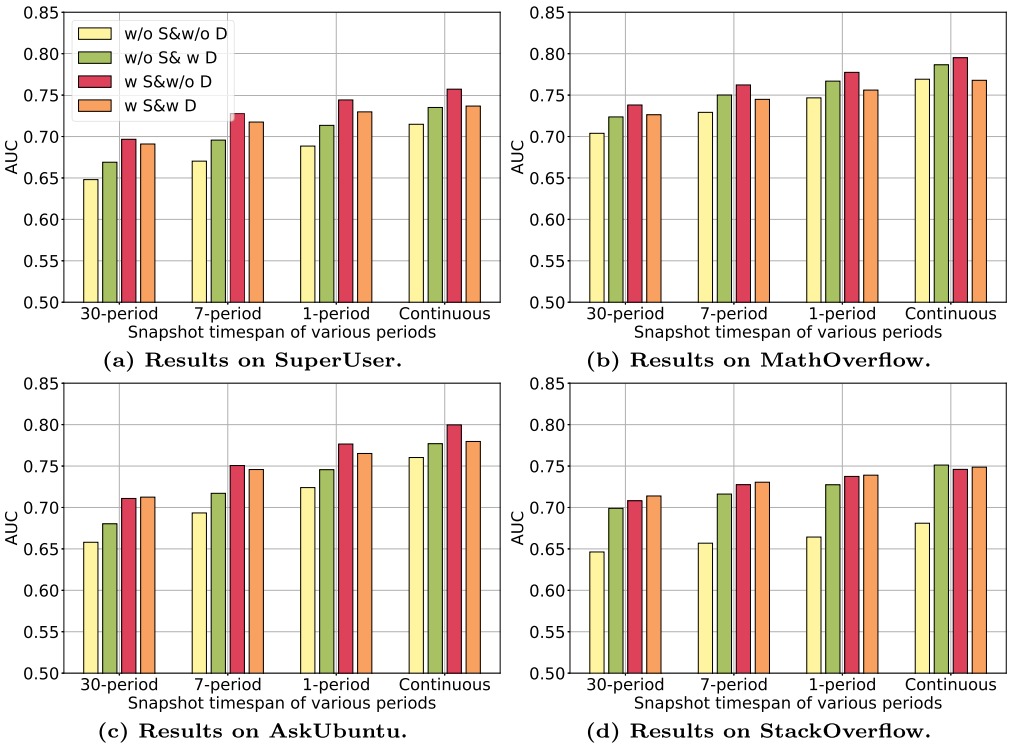


Fig. 5. Ablation study on the non-linearities in CLIP.

6.3. Analysis of non-linearities

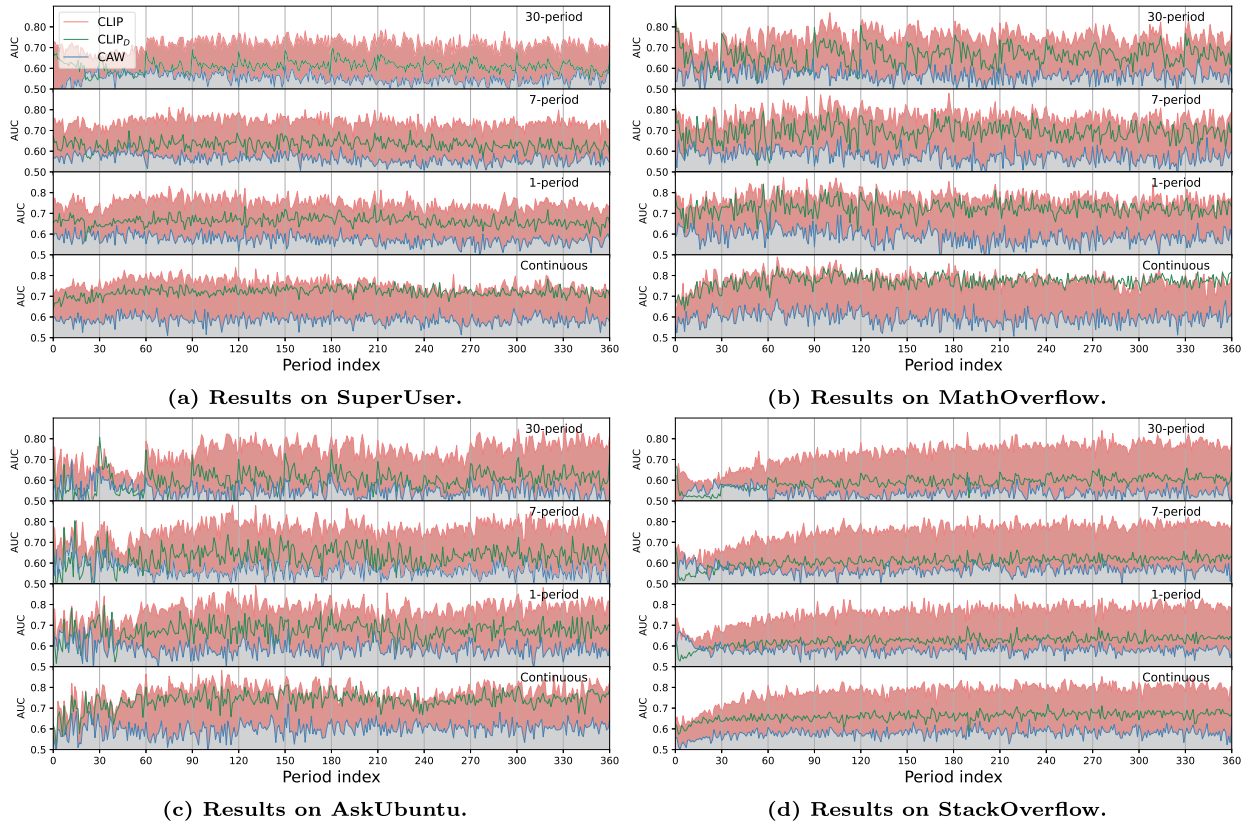
To examine the impact of the non-linearities in CLIP on prediction performance, we introduce four variants of CLIP by controlling the linearity or non-linearity in the structure component (corresponding to Eqs. (14) and (15)) and the distance component (corresponding to Eqs. (5), (6) and (7)). The variants include: (1) w/o S&w/o D, which adopt linearity in both the structure and distance components; (2) w/o S&w D, which adopt linearity in the structure component and non-linearity in the distance component, respectively; (3) w S&w/o D, which adopts non-linearity in the structure component and linearity in the distance component, respectively; and (4) w S&w D, which adopts non-linearity in both the structure and distance components. The results of four variants in terms of AUC on four datasets with snapshot timespans of various periods are shown in Fig. 5.

As shown in Fig. 5, we can observe that the variant w/o S&w/o D generally performs worst among the four variants in different cases on four datasets. Then, comparing w/o S&w D and w S& w/o D to w/o S&w/o D, we can observe that introducing the non-linearity in either the structure component or the distance component can both apparently contribute to the performance improvements. This observation highlights deep learning's capability to model the inherent nonlinearity present in training data for inductive temporal link prediction. In addition, focusing on the variant w S&w D, we can see that it generally underperforms w S&w/o D. We analyze the unsatisfactory performance of w S&w D is due to the fact that, on the basis of including the non-linearity in the structure component, the error resulted by the introduced non-linearity in the distance component in Eq. (11) decreases the performance. The experimental results align well with our analysis in Section 4.3.2.

6.4. Impact of time period

To assess CLIP's ability to estimate the true distance between nodes and evaluate its performance on real-world temporal networks across different time periods, we compare CLIP's performance with that of CAW and the variant CLIP_D. Specifically, the performance of CLIP and CAW as well as CLIP_D on the whole 360 time periods on four datasets is tested, and the results in terms of AUC are shown in Fig. 6. Here, we plot the performance curves of CLIP and CAW in red and blue, respectively, and fill the same colors under their respective curves, where the overlapping part (i.e., the area below the performance curve constituted by the lower performance of CLIP and CAW at each period) is filled in gray. Then, we can intuitively compare the performance difference between CLIP and CAW by examining the filled color above the gray region, where red indicates that CLIP outperforms CAW while blue signifies the opposite case. In addition, we plot the performance of CLIP_D using a single green line in each case on four datasets.

As shown in Fig. 6, we can first observe that with the time period increasing, the performance of all three models fluctuates, and CLIP generally outperforms CAW and CLIP_D in various cases on all four datasets, indicating the effectiveness of CLIP on different stages of the temporal networks. In addition, by meticulously comparing the performance of CLIP and CAW as well as CLIP_D, we observe that the performance gap between CLIP and CAW exhibits a remarkably similar changing trend to the performance of CLIP_D



(a) Results on SuperUser. (b) Results on MathOverflow.

(c) Results on AskUbuntu. (d) Results on StackOverflow.

Fig. 6. Results in terms of AUC on various time periods of four real-world temporal networks.

across all the snapshot-based and continuous temporal networks. That is, the gap widens with the performance of CLIP$_D$ improving, and narrows when CLIP$_D$'s performance declines. This indicates that the baseline CAW fails to accurately model the true distance between nodes for making future inductive temporal link predictions while CLIP well estimates the true distance, which conforms to the theoretical analysis in Section 4.

6.5. *Impact of substructure size*

To analyze the impact of the node substructure size on the performance of CLIP, we range the hop number and hop length in $\{4, 8, 16, 32\}$ and $\{1, 2, 3, 4\}$, respectively. We test the performance of CLIP with various node substructures on four real-world datasets. The results in terms of AUC are shown in Fig. 7.

As shown in Fig. 7, we can first observe the phenomenon that the performance changing is highly similar on four datasets. Moreover, by fixing the hop length and increasing the hop number, the performance of CLIP generally improves. This is due to the fact that including more neighbors at each hop can better reflect the node substructure features and estimate the true distance between nodes. In addition, the performance increasing rate generally decreases when the hop number increases. For instance, the increasing rate from hop number 4 to 8 with a 3-hop substructure on the 30-period MathOverflow is 10.06%, while those from hop number 8 to 16 and from 16 to 32 are 6.94% and 3.46%, respectively. This is due to the fact that while including more neighbors can contribute to modeling the node substructure features and the true distance between nodes, noisy neighbors may also be introduced as bias, disturbing accurate predictions of future inductive temporal links.

In addition, we can see that on the snapshot-based graphs of each dataset, including the snapshot timespan ranging from 30 periods to 1 period, when the hop number is fixed, increasing the hop length from 1 to 2 generally improves the performance of CLIP, while the performance of CLIP continuously decreases when the hop length increases from 2 to 4. We attribute this to the large hop length of the node substructure, which will introduce much noise disturbing the true node distance estimation and smooth the substructure features of nodes [2]. However, differently on the continuous temporal network of each dataset, the performance of CLIP continuously improves with the hop length increasing from 1 to 3, achieving peak performance at a hop length of 3, and then begins to decrease. The possible reason is that compared with the periodically increasing nodes and edges in the snapshot-based graphs, the constantly coming nodes and edges in the continuous graphs make the node substructure with large hop lengths more likely to include useful rather than noisy neighbors for accurate inductive temporal link predictions.

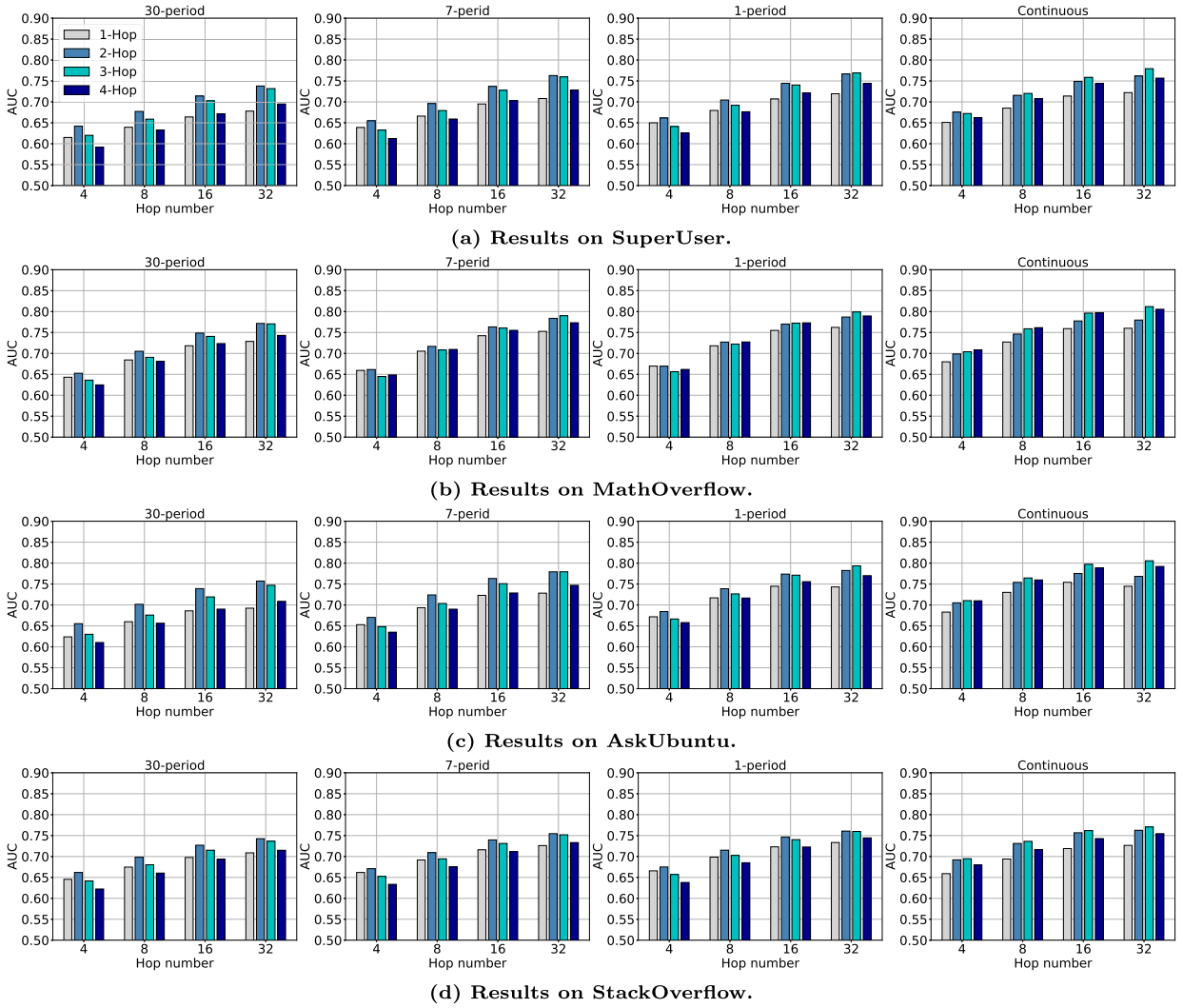


Fig. 7. Performance of CLIP in terms of AUC with various node substructures on four real-world temporal networks.

7. Conclusion

In this paper, we propose a Causal LInk Prediction (CLIP) framework for inductive temporal link prediction. Specifically, CLIP proposes to tackle the structure bias problem in the existing anonymous distance encoding method by utilizing the backdoor adjustment through the do-calculus, where the result is further approximated by decoupling the distance encoding vector, so as to estimate the true distance between nodes. After that, the node substructure features are considered as the intervention on the basis of the true node distance for adapting CLIP to realistic scenarios. In addition, we propose to train CLIP on the synthetic training graph data generated from the powerlaw cluster networks, which enables CLIP to capture the network evolution pattern across various networks and eventually achieve true inductive temporal link prediction. Experiments on four real-world temporal networks demonstrate the effectiveness of CLIP, and the generalization ability of CLIP is also validated by the performance evaluation on the synthetic test graph data with various distributions.

As to future work, we would like to investigate the adoption of non-linear functions in the backdoor adjustment approximation, as mentioned in Section 4.3.2. In addition, we are also interested in incorporating the deep graph structure learning [20,21] into CLIP to refine the topological structure for improving the link prediction performance.

CRediT authorship contribution statement

Zhiqiang Pan: Writing – original draft, Validation, Methodology, Formal analysis, Conceptualization. **Fei Cai:** Writing – review & editing, Project administration, Investigation. **Wanyu Chen:** Writing – review & editing, Resources. **Taihua Shao:** Writing – review & editing, Investigation. **Yupu Guo:** Investigation, Formal analysis. **Honghui Chen:** Resources, Methodology, Funding acquisition.

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.

Acknowledgements

This research was funded by the National Natural Science Foundation of China under Nos. 62372458 and 62302511, the Basic Strengthening Program (Young Elite Scientists Sponsorship Program) under No. 2023-JCJQ-QT-048 and the Postgraduate Scientific Research Innovation Project of Hunan Province under No. CX20230083.

References

[1] L.A. Adamic, E. Adar, Friends and neighbors on the web, Soc. Netw. 25 (2003) 211–230, https://doi.org/10.1016/S0378-8733(03)00009-1.
[2] F.M. Bianchi, D. Grattarola, L. Livi, C. Alippi, Graph neural networks with convolutional ARMA filters, IEEE Trans. Pattern Anal. Mach. Intell. 44 (2022) 3496–3507, https://doi.org/10.1109/TPAMI.2021.3054830.
[3] Z. Bu, Y. Wang, H. Li, J. Jiang, Z. Wu, J. Cao, Link prediction in temporal networks: integrating survival analysis and game theory, Inf. Sci. 498 (2019) 41–61, https://doi.org/10.1016/j.ins.2019.05.050.
[4] T.Q. Chen, Y. Rubanova, J. Bettencourt, D. Duvenaud, Neural ordinary differential equations, in: NeurIPS, 2018, pp. 6572–6583.
[5] W. Cong, S. Zhang, J. Kang, B. Yuan, H. Wu, X. Zhou, H. Tong, M. Mahdavi, Do we really need complicated model architectures for temporal networks?, in: The Eleventh International Conference on Learning Representations, ICLR 2023, Kigali, Rwanda, May 1–5, 2023, 2023.
[6] C. Gao, Y. Zheng, W. Wang, F. Feng, X. He, Y. Li, Causal inference in recommender systems: a survey and future directions, arXiv preprint, arXiv:2208.12397, 2022.
[7] E. Hajiramezanali, A. Hasanzadeh, K.R. Narayanan, N. Duffield, M. Zhou, X. Qian, Variational graph recurrent neural networks, in: NeurIPS, 2019, pp. 10700–10710.
[8] W.L. Hamilton, Z. Ying, J. Leskovec, Inductive representation learning on large graphs, in: NeurIPS, 2017, pp. 1024–1034.
[9] P. Holme, B.J. Kim, Growing scale-free networks with tunable clustering, Phys. Rev. E 65 (2002) 026107.
[10] C. Hou, H. Zhang, S. He, K. Tang, Glodyne: global topology preserving dynamic network embedding, IEEE Trans. Knowl. Data Eng. 34 (2022) 4826–4837, https://doi.org/10.1109/TKDE.2020.3046511.
[11] H. Huang, J. Tang, L. Liu, J. Luo, X. Fu, Triadic closure pattern analysis and prediction in social networks, IEEE Trans. Knowl. Data Eng. 27 (2015) 3374–3389, https://doi.org/10.1109/TKDE.2015.2453956.
[12] M. Jin, Y. Li, S. Pan, Neural temporal walks: motif-aware representation learning on continuous-time dynamic graphs, in: NeurIPS, 2022.
[13] S.M. Kazemi, R. Goel, K. Jain, I. Kobyzev, A. Sethi, P. Forsyth, P. Poupart, Representation learning for dynamic graphs: a survey, J. Mach. Learn. Res. 21 (2020) 70:1–70:73.
[14] T.N. Kipf, M. Welling, Semi-supervised classification with graph convolutional networks, in: ICLR, 2017.
[15] S. Kumar, X. Zhang, J. Leskovec, Predicting dynamic embedding trajectory in temporal interaction networks, in: KDD, ACM, 2019, pp. 1269–1278.
[16] P. Li, Y. Wang, H. Wang, J. Leskovec, Distance encoding: design provably more powerful neural networks for graph representation learning, in: NeurIPS, 2020.
[17] Y. Li, D. Tarlow, M. Brockschmidt, R.S. Zemel, Gated graph sequence neural networks, in: ICLR, 2016.
[18] Y. Li, Y. Wu, M. Sun, B. Yang, Y. Wang, Learning continuous dynamic network representation with transformer-based temporal graph neural network, Inf. Sci. 649 (2023) 119596, https://doi.org/10.1016/j.ins.2023.119596.
[19] B. Liu, M. Ding, S. Shaham, W. Rahayu, F. Farokhi, Z. Lin, When machine learning meets privacy: a survey and outlook, ACM Comput. Surv. 54 (2022) 31:1–31:36, https://doi.org/10.1145/3436755.
[20] N. Liu, X. Wang, L. Wu, Y. Chen, X. Guo, C. Shi, Compact graph structure learning via mutual information compression, in: WWW, ACM, 2022, pp. 1601–1610.
[21] Y. Liu, Y. Zheng, D. Zhang, H. Chen, H. Peng, S. Pan, Towards unsupervised deep graph structure learning, in: WWW, ACM, 2022, pp. 1392–1403.
[22] Z. Liu, C. Huang, Y. Yu, J. Dong, Motif-preserving dynamic attributed network embedding, in: WWW, ACM / IW3C2, 2021, pp. 1629–1638.
[23] F. Lorrain, H.C. White, Structural equivalence of individuals in social networks, J. Math. Sociol. 1 (1971) 49–80.
[24] L. Lü, C.H. Jin, T. Zhou, Similarity index based on local paths for link prediction of complex networks, Phys. Rev. E 80 (2009) 046122.
[25] M.E. Newman, Clustering and preferential attachment in growing networks, Phys. Rev. E 64 (2001) 025102.
[26] Z. Pan, F. Cai, X. Liu, H. Chen, Distance-aware learning for inductive link prediction on temporal networks, IEEE Trans. Neural Netw. Learn. Syst. (2023), https://doi.org/10.1109/TNNLS.2023.3328924.
[27] J. Pearl, Causality, Cambridge University Press, 2009.
[28] Q. Ren, Y. Chen, Y. Mo, Q. Wu, J. Yan, DICE: domain-attack invariant causal learning for improved data privacy protection and adversarial robustness, in: KDD, ACM, 2022, pp. 1483–1492.
[29] E. Rossi, B. Chamberlain, F. Frasca, D. Eynard, F. Monti, M.M. Bronstein, Temporal graph networks for deep learning on dynamic graphs, arXiv preprint, arXiv:2006.10637, 2020.
[30] D.E. Rumelhart, G.E. Hinton, R.J. Williams, Learning representations by back-propagating errors, Nature 323 (1986) 533–536.
[31] A. Sankar, Y. Wu, L. Gou, W. Zhang, H. Yang, Dysat: deep neural representation learning on dynamic graphs via self-attention networks, in: WSDM, ACM, 2020, pp. 519–527.
[32] U. Singer, I. Guy, K. Radinsky, Node embedding over temporal graphs, in: Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI), 2019.
[33] Q. Tian, K. Kuang, K. Jiang, F. Liu, Z. Wang, F. Wu, Confoundergan: protecting image data privacy with causal confounder, in: NeurIPS, 2022.
[34] R. Trivedi, M. Farajtabar, P. Biswal, H. Zha, Dyrep: learning representations over dynamic graphs, in: ICLR, 2019.
[35] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A.N. Gomez, L. Kaiser, I. Polosukhin, Attention is all you need, in: NeurIPS, 2017, pp. 5998–6008.
[36] P. Velickovic, G. Cucurull, A. Casanova, A. Romero, P. Liò, Y. Bengio, Graph attention networks, in: ICLR, 2018.
[37] J. Wang, K. Ding, L. Hong, H. Liu, J. Caverlee, Next-item recommendation with sequential hypergraphs, in: SIGIR, ACM, 2020, pp. 1101–1110.
[38] J. Wang, G. Song, Y. Wu, L. Wang, Streaming graph neural networks via continual learning, in: Proceedings of the International Conference on Information and Knowledge Management (CIKM), 2020.

[39] W. Wang, F. Feng, X. He, X. Wang, T. Chua, Deconfounded recommendation for alleviating bias amplification, in: KDD, ACM, 2021, pp. 1717–1725.
[40] X. Wang, H. Yang, M. Zhang, Neural common neighbor with completion for link prediction, CoRR, arXiv:2302.00890 [abs], https://doi.org/10.48550/arXiv.2302.00890, 2023.
[41] Y. Wang, Y.-Y. Chang, Y. Liu, J. Leskovec, P. Li, Inductive representation learning in temporal networks via causal anonymous walks, in: 9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3–7, 2021, 2021.
[42] Y. Wang, P. Li, C. Bai, J. Leskovec, TEDIC: neural modeling of behavioral patterns in dynamic social interaction networks, in: WWW, ACM / IW3C2, 2021, pp. 693–705.
[43] Z. Wu, S. Pan, F. Chen, G. Long, C. Zhang, P.S. Yu, A comprehensive survey on graph neural networks, IEEE Trans. Neural Netw. Learn. Syst. 32 (2021) 4–24, https://doi.org/10.1109/TNNLS.2020.2978386.
[44] D. Xu, C. Ruan, E. Körpeoglu, S. Kumar, K. Achan, Inductive representation learning on temporal graphs, in: ICLR, 2020.
[45] C. Yang, C. Wang, Y. Lu, X. Gong, C. Shi, W. Wang, X. Zhang, Few-shot link prediction in dynamic networks, in: WSDM, ACM, 2022, pp. 1245–1255.
[46] C. Yang, J. Li, K. Lu, B. Hooi, L. Zhou, Continuous-time graph directed information maximization for temporal network representation, Inf. Sci. 644 (2023) 119240, https://doi.org/10.1016/j.ins.2023.119240.
[47] X. Yang, F. Xiao, An improved gravity model to identify influential nodes in complex networks based on k-shell method, Knowl.-Based Syst. 227 (2021) 107198, https://doi.org/10.1016/j.knosys.2021.107198.
[48] Y. Yin, Y. Wu, X. Yang, W. Zhang, X. Yuan, SE-GRU: structure embedded gated recurrent unit neural networks for temporal link prediction, IEEE Trans. Netw. Sci. Eng. 9 (2022) 2495–2509, https://doi.org/10.1109/TNSE.2022.3164659.
[49] T. Zhao, G. Liu, D. Wang, W. Yu, M. Jiang, Learning from counterfactual links for link prediction, in: ICML, PMLR, 2022, pp. 26911–26926.
[50] L. Zhou, Y. Yang, X. Ren, F. Wu, Y. Zhuang, Dynamic network embedding by modeling triadic closure process, in: AAAI, AAAI Press, 2018, pp. 571–578.