Knowledge-Based Systems 292 (2024) 111619

Contents lists available at ScienceDirect

Knowledge-Based Systems

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

Link prediction by continuous spatiotemporal representation via neural differential equations ☆

Liyi Huang [a,b], Bowen Pang [b,c], Qiming Yang [b,c], Xiangnan Feng [d], Wei Wei [a,b,c,e,*]

[a] Institute of Artificial Intelligence, Beihang University, Beijing, 100191, China
[b] Key Laboratory of Mathematics Informatics Behavioral Semantics, Ministry of Education, Beijing, 100191, China
[c] School of Mathematical Sciences, Beihang University, Beijing, 100191, China
[d] Complexity Science Hub Vienna, Vienna, 1080, Austria
[e] Zhongguancun Laboratory, Beijing, 100094, China

ARTICLE INFO

Keywords:
Graph neural networks
Neural differential equation
Temporal link prediction
Dynamic network

ABSTRACT

With the continuous advancement of data science and machine learning, temporal link prediction has emerged as a crucial aspect of dynamic network analysis, providing significant research and application potential across various domains. While deep learning techniques have achieved remarkable results in temporal link prediction, most existing studies have focused on discrete model frameworks. These frameworks face limitations in capturing deep structural features and effectively aggregating temporal information. To address these limitations, we draw inspiration from neural differential equations to propose a Continuous Temporal Graph Neural Differential Equation (CTGNDE) network model for temporal link prediction. Specifically, we design a spatial graph Ordinary Differential Equation (ODE) to capture the spatial correlations inherent in complex spatiotemporal information. Then we employ Neural Controlled Differential Equation (Neural CDE) to learn complex evolution patterns and effectively aggregate temporal information. Finally, we characterize completely continuous and more accurate hidden state trajectories by coupling spatial and temporal messages. Experiments conducted on 10 real-world network datasets validated the superior performance of the CTGNDE model over the state-of-the-art baselines.

1. Introduction

Graph is a natural means to model complex systems with interrelated components. Owing to its ability to capture and model diverse relationships in real-world networks, graph structure appears in diverse fields such as social networks [1], biological systems [2], and transportation networks [3], etc. In view of this, the problem of temporal link prediction arises within such complex networks. Temporal link prediction models are generally used to predict future connections at specific timestamps based on the network's existing topology and node attributes. The prediction of future connection patterns not only helps us better understand the evolution of network structures but also guides us in making better decisions. Consequently, link prediction models have numerous applications, including social network analysis [4], coauthor network analysis [5], drug discovery [6], and recommendation systems [7].

In recent years, with the efforts of researchers, various methods based on Graph Neural Network (GNN) models have been proposed to solve the problem of link prediction [8–10]. However, it is noteworthy that the majority of existing methods tend to concentrate solely on static graphs. In practice, real-world networks frequently exhibit inherent dynamics, as observed in domains such as social and transportation networks. In these scenarios, both network structures and node attributes undergo dynamic and often imperceptible changes over time. In a dynamic network, the interdependence between the time and space dimensions can substantially impact the connections between nodes. For instance, the attributes of a node at a specific timestamp may be influenced by the attributes of its neighboring nodes at the same timestamp, and the connections between nodes can evolve over time. Consequently, it is important to develop a comprehensive temporal network link prediction approach that can effectively capture spatiotemporal dependency, allowing for the accurate modeling of

☆ This work was supported by the National Natural Science Foundation of China (Grant Nos. 62276013, 62141605, 62050132), the Beijing Natural Science Foundation (Grant No. 1192012), and the Fundamental Research Funds for the Central Universities.

* Corresponding author.
E-mail addresses: liyihuang@buaa.edu.cn (L. Huang), pangbw@buaa.edu.cn (B. Pang), asdyqm@buaa.edu.cn (Q. Yang), fengxiangnan@gmail.com (X. Feng), weiw@buaa.edu.cn (W. Wei).

https://doi.org/10.1016/j.knosys.2024.111619
Received 4 September 2023; Received in revised form 7 February 2024; Accepted 6 March 2024
Available online 7 March 2024

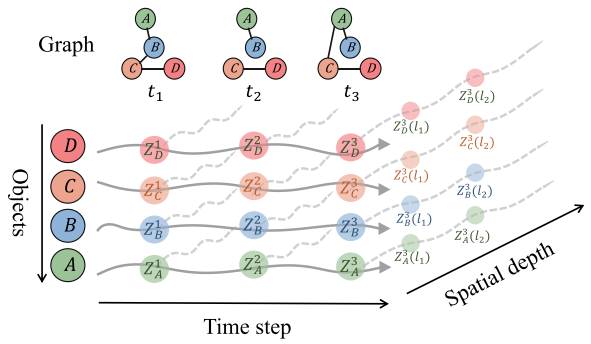

Fig. 1. A schematic diagram of our method, where Z denotes the node embedding. For node objects in a dynamic graph, their embeddings not only continuously evolve with timesteps but also continuously change with the depth of the spatial structure at each timestep.

both the structural properties and temporal characteristics of network evolution.

Numerous methodologies have been proposed to address the problem of temporal link prediction using deep learning techniques. These methodologies are designed to capture temporal and spatial patterns, with a notable emphasis on the amalgamation of a GNN with a Recurrent Neural Network (RNN) [11–13]. While these methods exhibit strong nonlinear modeling ability and can effectively handle high-dimensional, nonlinear, and complex dynamic graph data, they still face some challenges that hinder them from accurately predicting links. On the one hand, the prevalent snapshot-based approach in existing research dissects dynamic networks into multiple chronological snapshots. Consequently, the temporal features extracted often have a discretized nature. However, it is important to acknowledge that the evolution of spatiotemporal graphs fundamentally unfolds as a continuous dynamic process. Furthermore, a notable gap in most existing studies is the oversight of structural continuity within models, which is illustrated in Fig. 1. Discrete neural architectures may fall short in capturing finer-grained information and could potentially omit crucial details in the evolution of links. This oversight has been empirically shown to result in heightened numerical inaccuracies and a parallel escalation in model complexity [14,15]. On the other hand, most graph neural network-based approaches encounter the challenge of oversmoothing [16], which limits the network's ability to obtain deeper and richer spatial structural features and weakens the representation ability of GNN.

In this study, we propose Continuous Temporal Graph Neural Differential Equation Network (CTGNDE) to address the aforementioned problems. Inspired by the combination of differential equations and deep learning [14,15,17,18], we adopt neural differential equations to construct a continuous-depth model so as to capture the continuous evolution of dynamic graphs. Specifically, we formulate node representation on dynamic graphs as a coupling of temporal evolution information and spatial structural features. Our proposed model has two main components: spatial graph Ordinary Differential Equation (ODE) and temporal graph ODE. The spatial graph ODE component aims to capture the dynamic spatial correlations among nodes. This component focuses on the aggregation of spatial structural features and consists of two stages: feature propagation and feature transformation. Furthermore, we employ a continuous pattern and a shared global parameter matrix to aggregate the neighbor structural information of nodes, thus capturing deeper information. In addition, for the temporal graph ODE component, we incorporate richer temporal information by aggregating the current and historical moment information of each node through Neural Controlled Differential Equations (Neural CDEs) so as to capture long-range temporal correlations. Notably, our model is continuous in both the temporal and spatial levels of hidden states. Our contributions can be summarized as follows:

- We propose CTGNDE, a novel framework for completely continuous temporal graph neural networks. This framework effectively characterizes the hidden dynamics of nodes in both continuous space and continuous time, capturing and coupling the rich temporal and spatial messages from hidden spaces.
- We design a spatial graph ODE component that captures intricate spatial structural features while taking into account spatiotemporal correlations. Experimental results indicate the effectiveness of our approach in handling complex spatiotemporal dependencies and addressing the challenges of oversmoothing.
- We conduct extensive experiments on 10 real-world datasets and show that our proposed method outperforms other baseline models, thereby demonstrating the effectiveness of our approach.

2. Related works

2.1. Temporal link prediction

Temporal link prediction is an essential task in network analysis that has gained considerable attention in recent years. While early approaches, such as time series and probabilistic graphical models [19], were successful in some cases, they often relied on strong assumptions about the underlying distribution of the temporal evolution of the link structure and may have limited expressiveness. In recent years, deep learning-based methods have demonstrated promising performance by leveraging the power of neural networks to model complex temporal dependencies. In particular, graph neural networks, especially graph convolutional neural networks [20], have been successfully applied to temporal link prediction. For example, EvolveGCN [12] introduces an adaptive mechanism to update the Graph Convolutional Network (GCN) weights over time to handle the dynamic nature of the evolving networks. GCN-GRU [11] and VGRNN [13] combine GCNs with gated recurrent units to model the temporal information of the link sequences. Another popular approach is the use of temporal attention mechanisms in GNNs. For instance, TGAT [21] employs a self-attention mechanism to develop a time encoding technique that learns the temporal feature interactions. TARGAT [22] uses time-aware relational graph attention layers and temporal transformer layers to effectively handle the task of temporal knowledge graph embedding. Although these models have achieved considerable success in the domain of graph-based tasks, they may exhibit certain structural constraints, notably characterized by the discontinuous nature of information propagation. This phenomenon perhaps gives rise to the partial loss and incompleteness of vital information within the framework.

2.2. Neural differential equation

The integration of dynamic systems and deep learning has emerged as a compelling research area in machine learning. Recently, neural differential equations have gained attention as a powerful tool for modeling continuous-time dynamic systems. The Neural Ordinary Differential Equation (Neural ODE) proposed by Chen et al. [14], characterizes the derivative of a hidden state with a neural network parameterized differential equation. The Neural ODE enables the construction of a continuity hierarchy and parameters, yielding promising results in various applications [23–25]. Despite its success, the Neural ODE framework has some limitations, such as sensitivity to initial conditions. To address this challenge, the Neural CDE [17] was proposed, which adjusts the trajectory of the solutions based on subsequent observations, enhancing their robustness to perturbations in the initial conditions. Furthermore, the Augmented Neural ODE (ANODE) model [26] adds extra dimensions to the hidden state of the network to model additional variables, and such an addition improves the accuracy and generalization of the model. Another variant, the Second Order Neural ODE [27], employs a second-order ODE to describe the evolution of the second derivative of the hidden state. This approach captures more complex temporal dynamics and enhances the interpretation of ANODE.

2.3. Graph neural ordinary differential equations

Graph neural networks have become a powerful framework for learning graph representations. With the proposal of Neural ODE, researchers have extended the GNN framework to operate on continuously evolving graphs. Graph neural ordinary Differential Equations (GDEs) [28] enable modeling of dynamic graph-structured data by formulating the evolution of the graph state as a continuous-time ODE. To further improve the expressiveness of GNNs, inspired by Pagerank- and diffusion-based methods, the Continuous Graph Neural Network (CGNN) [18] was proposed, which extends the existing discrete GNN and describes the continuous dynamics of node representation. Experimental results have indicated that this method is robust to oversmoothing. Furthermore, several continuous graph ODE models have been proposed for traffic flow prediction. For instance, the Multivariate Time series with dynamic Graph neural Ordinary Differential Equations (MTGODE) [15] designs two coupled ODEs for unified spatiotemporal message passing to learn the fully continuous latent spatiotemporal dynamics of the time series. In addition, the spatiotemporal Graph Ordinary Differential Equation Networks (STGODE) [29] were introduced to capture spatiotemporal dynamics through a tensor-based ODE.

3. Methodology

3.1. Problem formulation

In the temporal link prediction task, we are often given a dynamic graph $\mathcal{G}$ represented as a sequence of graph snapshots over time, $\mathcal{G} = \{G_1, G_2, \dots, G_T\}$, where $G_t = (\mathcal{V}_t, \mathcal{E}_t)$ denotes the graph at time $t \in \{1, 2, \dots, T\}$ with $\mathcal{V}_t$ and $\mathcal{E}_t \subseteq \mathcal{V}_t \times \mathcal{V}_t$ being the corresponding node and edge sets, respectively. To represent node attributes, we use a feature matrix $X_t \in \mathbb{R}^{|\mathcal{V}_t| \times m}$, where each node in $\mathcal{V}_t$ is encoded as a predefined m-dimensionality attribute vector. In addition, the graph G_t can be represented by the adjacency matrix $A_t \in \mathbb{R}^{|\mathcal{V}_t| \times |\mathcal{V}_t|}$. Due to the different structures and node attributes observed at each timestamp, an adjacency matrix sequence $A = \{A_1, A_2, \dots, A_T\}$ and a node attribute sequence $X = \{X_1, X_2, \dots, X_T\}$ are considered as inputs. The goal of our task is to learn a mapping function f from the historical T observations to predict the links in the graph at time $T + 1$, i.e., $\mathcal{E}_{T+1}$.

Note that $\mathcal{G}$ is an undirected graph, and we only consider changes in the edge sets over time, i.e., we assume that all graph snapshots have the same nodes, and $|\mathcal{V}_t| = N$. To simplify the input structure, in practice, we convert the two-dimensional vector sequences of length T into the three-dimensional tensors: $X \in \mathbb{R}^{T \times N \times m}$, $A \in \mathbb{R}^{T \times N \times N}$.

3.2. Overall framework

Fig. 2 shows the overview of our proposed CTGNDE network framework. The framework has four main components: input preprocessing, spatial graph ODE, temporal graph ODE, and prediction. With a series of graph snapshots, we first calculate the potential initial states of the nodes in the graph snapshots using an encoder. Subsequently, our spatial graph ODE captures rich continuous spatial structural features from the graph snapshots through a two-stage process consisting of feature propagation and feature transformation. Additional information regarding this process is presented in Section 3.3. The extracted node features obtained from the spatial graph ODE are then fed into the temporal graph ODE, which aggregates temporal information and uses an ODE solver to predict the potential future state of the nodes. This process is described in detail in Section 3.4. Finally, the learned node representation is utilized to predict the probability of future edge connections. Section 3.5 delves into the details of model training and algorithm utilized in our framework.

3.3. Spatial graph ODE

To effectively capture the complex structural information at each timestep, we usually utilize spectral convolution to extract graph structural data and aggregate nodes' neighbor information. The formula of a typical GCN is as follows [20]:

$$H_{(l+1)} = \sigma\left(\hat{D}^{-\frac{1}{2}}\hat{A}\hat{D}^{-\frac{1}{2}}H_{(l)}W_{(l)}\right), \tag{1}$$

where $H_{(l)} \in \mathbb{R}^{N \times m}$ denotes the node feature matrix at layer l; $\hat{A} \in \mathbb{R}^{N \times N}$, the normalized adjacency matrix; $\hat{D}$, the degree matrix; and $W_{(l)} \in \mathbb{R}^{m \times m}$, the weight matrix for layer l. However, GCN introduces computational complexity due to the utilization of nonlinear activation functions and weight matrices at each layer to learn node representations. This complexity increases the computational burden. Drawing inspiration from the success of the linear variant of GCN [15,30], given an adjacency matrix A and the initial feature X, a discrete dynamic formulation of the nodes in the K-th layer using the space-based convolution operation is defined as follows:

$$\begin{cases} H_{k+1} = \hat{A} \times_2 H_k + H_0, & k \in \{0, 1, \dots, K-1\}, \\ \bar{H}_K = \sigma(H_K \times_1 W \times_3 \Theta), \end{cases} \tag{2}$$

where $\bar{H}_K$ denotes the hidden representations of nodes at the K-th layer; $\hat{A} \in \mathbb{R}^{T \times N \times N}$, the normalized adjacency matrix; $\Theta \in \mathbb{R}^{m \times m}$, a learnable weight matrix; and $W \in \mathbb{R}^{T \times T}$, a trainable temporal transformation matrix. Furthermore, $H_0 = f(X) \in \mathbb{R}^{T \times N \times m}$ denotes the initial nodes' hidden embedded representation, which is obtained by mapping the input series to the latent space with a neural network f; σ, a nonlinear activation function such as ReLU; and $\times_i$, the summation over the i-th dimension of the tensor products by using the Einstein summation convention [29]. i.e., $(\hat{A} \times_2 H_k)_{ilm} = \sum_j \hat{A}_{ijl} \cdot (H_k)_{ijm}$, $(H_K \times_1 W \times_3 \Theta)_{ljm} = \sum_k (\sum_i (H_K)_{ijk} \cdot W_{il})_{ljk} \cdot \Theta_{km}$.

To effectively capture complex temporal and spatial dependencies, we utilize the spatiotemporal tensors as inputs. It is important to note the difference between the standard GCN and our proposed formula. In each layer of GCN, the update of nodes is transformed through a nonlinear activation function. Different from the standard GCN, our proposed formula divides the updating of node representation into two stages: feature propagation and feature transformation. During feature propagation, nodes directly update across multiple layers through neighboring nodes, whereas in the subsequent feature transformation stage, nonlinear activation operation is applied only once to the propagated information. This not only eliminates the redundant nonlinearity between continuous layers but also compresses the weight matrix, resulting in a simple yet effective model. By recursively applying Eq. (2), we can effectively couple the temporal and structural information on the dynamic graph through tensor multiplication.

The aforementioned feature propagation stage formula can be expressed as follows:

$$H_K = \sum_{i=0}^{K-1} (\hat{A}^i \times_2 H_0). \tag{3}$$

To extend Eq. (3) to a continuous form, we replace the discrete variable K with a continuous variable t. In this way, Eq. (3) can be viewed as a Riemann sum. The Riemann integral formulation is given as follows:

$$H_t = \int_0^t \hat{A}^s \times_2 H_0 \mathrm{d}s. \tag{4}$$

Intuitively, we can obtain an ODE by taking the derivative,

$$\frac{\mathrm{d}H_t}{\mathrm{d}t} = \hat{A}^t \times_2 H_0. \tag{5}$$

However, computing $\hat{A}^t$ for $t \in \mathbb{R}$ can be challenging. To simplify the calculation, we consider the second-order ODE:

$$\frac{\mathrm{d}^2 H_t}{\mathrm{d}t^2} = \ln \hat{A} \times_2 \hat{A}^t \times_2 H_0 = \ln \hat{A} \times_2 \frac{\mathrm{d}H_t}{\mathrm{d}t}. \tag{6}$$

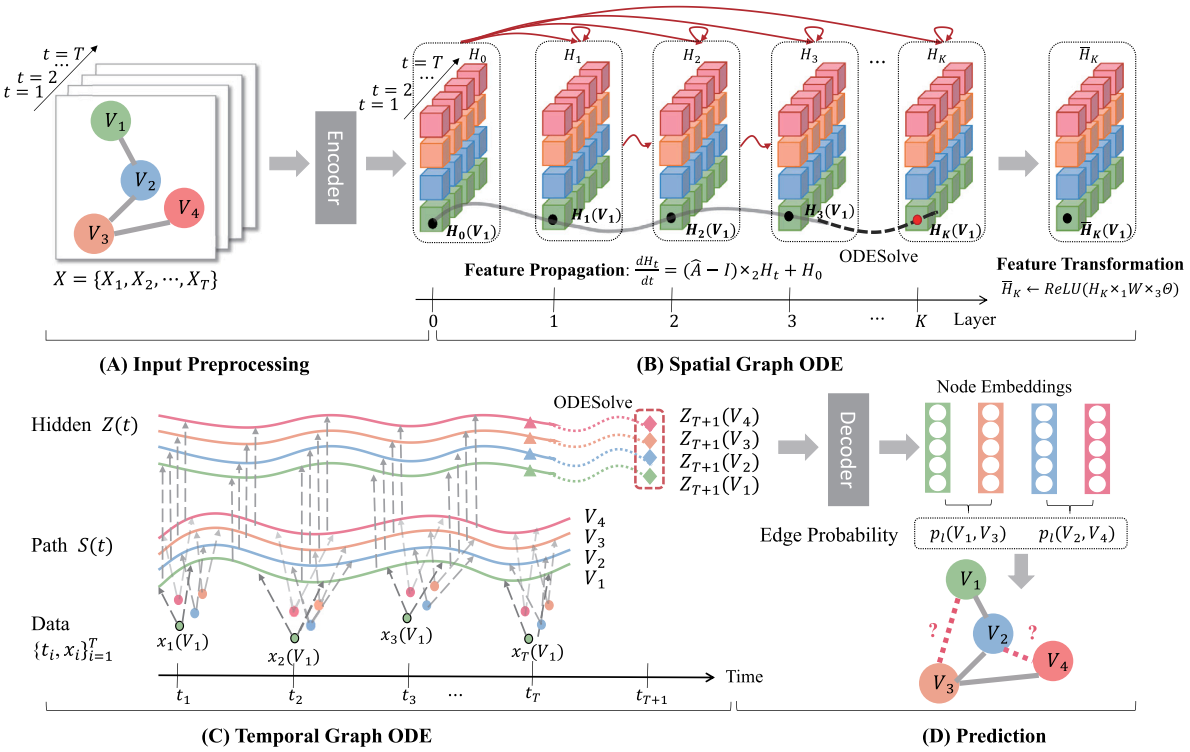

Fig. 2. Overall framework of CTGNDE: first, we encode a series of historical observations X through the encoder to obtain the potential initial state of the nodes in the graph. Subsequently, we employ the spatial graph ODE to model the continuous spatial structural features. The red line denotes the feature transmission process and the gray curve denotes the continuous hidden state of the nodes through the ODE solver. Subsequently, we construct a continuous path S through interpolation and then employ the temporal graph ODE to model the aggregation of continuous temporal information. Finally, leveraging the learned node representation, we make predictions regarding future link connections.

By integrating both sides of Eq. (6), we obtain:

$$\frac{\mathrm{d}H_t}{\mathrm{d}t} = \ln \hat{A} \times_2 H_t + const. \tag{7}$$

As for Eq. (4), we can evaluate it at $t = 1$ as follows:

$$H_1 = \int_0^1 \hat{A}^s \times_2 H_0 \mathrm{d}s = \frac{\hat{A} - I}{\ln \hat{A}} \times_2 H_0. \tag{8}$$

To solve for $const$, we combine Eqs. (5) and (7) by letting $t = 1$:

$$\left.\frac{\mathrm{d}H_t}{\mathrm{d}t}\right|_{t=1} = \hat{A} \times_2 H_0 = \ln \hat{A} \times_2 H_1 + const. \tag{9}$$

Thus, we have,

$$const = H_0. \tag{10}$$

The final ODE of Eq. (5) can be reformulated as,

$$\frac{\mathrm{d}H_t}{\mathrm{d}t} = \ln \hat{A} \times_2 H_t + H_0. \tag{11}$$

In practice, computing the matrix logarithm of $\hat{A}$ can be computationally expensive. Consequently, a first-order Taylor expansion is often used as an approximation. In this case, we can express the continuous-depth version of the feature propagation stage in Eq. (2) as an ODE, which is given by,

$$\frac{\mathrm{d}H_t}{\mathrm{d}t} = \left(\hat{A} - I\right) \times_2 H_t + H_0. \tag{12}$$

Up to this point, given the initial embedding representation H_0 and continuous variable t, the spatial structural representation updated in Eq. (2) can be expressed as a spatial graph ODE in the following form:

$$\begin{cases} H_t = \mathrm{ODESolve}\left(\frac{\mathrm{d}H_t}{\mathrm{d}t}, H_0, t\right), \\ \bar{H}_t = \sigma(H_t \times_1 W \times_3 \Theta), \end{cases} \tag{13}$$

where the function ODESolve is a black-box differential equation solver such as Euler or Runge–Kutta method employed to solve the ODE.

3.4. Temporal graph ODE

After capturing the spatial dependencies between nodes in the network through Eq. (13), it is crucial to incorporate rich temporal information. Among the existing research endeavors, RNN models have been extensively used to address time series-related challenges. A standard RNN cell can be denoted as follows:

$$h_t = f_\theta\left(h_{t-1}, x_t\right), \tag{14}$$

where x_t denotes the input vector; h_t, the hidden state vector; and f, a function incorporating the model parameter θ. It is noteworthy that the hidden state of the current timestep is not only dependent on the input at that moment but also incorporates information from the hidden state of the previous timestep.

Despite the time-dependent nature of RNN, it relies on discrete timesteps to model time-series data sequentially, necessitating individual computations for forward propagation and backpropagation. Consequently, to enhance the flexibility, accuracy, and computational efficiency of modeling sequence data, it becomes imperative to explore alternative approaches that leverage continuous timesteps for modeling.

First, we draw a comparison between RNNs and differential equations. In particular, the state update mechanism of an RNN can be seen as analogous to the state evolution in a differential equation. The recursive connections in an RNN resemble the integral operation in a differential equation, where the current state depends on the previous state. Thus, we can draw a comparison between Eq. (14) and the following explicit Euler discrete form [17]:

$$h(t) = h(0) + \int_0^t \left(f_\theta\left(h(s), x(s)\right) - h(s)\right) \mathrm{d}s. \tag{15}$$

In this case, the integrand in Eq. (15) can be regarded as a function of $h(s)$ and $x(s)$. In accordance with a theorem presented by Kidger et al. [17], it follows that "Any equation of the form $z_t = z_0 +$

$\int_{t_0}^{t} h_\theta(z_s, X_s)\mathrm{d}s$ may be represented exactly by a Neural CDE of the form $z_t = z_0 + \int_{t_0}^{t} f_\theta(z_s)\mathrm{d}X_s$. However the converse statement is not true". Consequently, RNNs can be perceived as discrete Neural CDEs [23], and we consider capturing hidden states of continuous timesteps based on Neural CDE.

Let $f_\theta : \mathbb{R}^{N\times m} \to \mathbb{R}^{N\times m\times d_S}$ be a neural network, where d_S denotes the dimension of the continuous path S. The Neural CDE [17] for the time aggregation process can be rewritten as follows:

$$Z(T) = Z(0) + \int_0^T (f_\theta(Z(t))) \cdot \mathrm{d}S(t), \tag{16}$$

where $Z(t) \in \mathbb{R}^{N\times m}$ denotes the hidden trajectory of nodes (over time $t \in [0, T]$) and is controlled by S. Here $(f_\theta(Z(t))) \cdot \mathrm{d}S(t)$ denotes a tensor–vector multiplication, and the integral $\mathrm{d}S(t)$ is a Riemann–Stieltjes integral. $S : [0, T] \to \mathbb{R}^{d_S}$ denotes a continuous function of bounded variation, which is a continuous path created from the spatial information $\bar{H}_t$ obtained using Eq. (13). Specifically, for $\bar{H}_t \in \mathbb{R}^{T\times N\times m}$, we split it into the observed value $x_i \in \mathbb{R}^{N\times m}$ at each timestamp $t_i \in \mathbb{R}^T$, i.e., $\bar{H}_t = \{x_i\}_{i=t_0}^{t_T} = \left((t_0, x_0), (t_1, x_1), \ldots, (t_T, x_T)\right)$, which is a discrete data sequence. Subsequently, we interpolate this discrete sequence using natural cubic splines to construct a continuous path S [17], i.e.,

$$S(t) = \sum_{i=t_0}^{t_T} S_i(t) \cdot x_i, \tag{17}$$

where $S_i(t)$ is a cubic spline interpolation function of the time variable t.

Thus, we construct a continuous observation path S, which approximates the observed discretization process $\{x_i\}_{i=t_0}^{t_T}$. Using Eq. (16), we model in the way of continuous time and finally extract the continuous hidden state containing rich temporal information. Compared with standard RNNs, this continuous hidden state better captures the dynamic evolution and long-term dependence of data, making the modeling of sequence data more accurate and stable. Furthermore, due to the use of Riemann–Stieltjes integration and interpolation in Eq. (16), the hidden state at the previous timestamp will change dynamically with the input data at the current timestamp [17], but the hidden state obtained by RNNs will not change with the input data, indicating that our model is more flexible. In addition, the forward propagation process in RNN can be regarded as the initial value problem of Eq. (16) to solve the differential equation. The temporal graph ODE is expressed as follows:

$$\begin{cases} \dfrac{\mathrm{d}Z(t)}{\mathrm{d}t} = (f_\theta(Z(t))) \cdot \dfrac{\mathrm{d}S(t)}{\mathrm{d}t}, \\ Z(t) = \mathrm{ODESolve}\left(\dfrac{\mathrm{d}Z(t)}{\mathrm{d}t}, Z(0), t\right), \end{cases} \tag{18}$$

where $Z(0)$ is created from $\bar{H}$ in Eq. (13) at timestamp t_0. For simplicity, we define f as a Multilayer Perceptron(MLP).

3.5. Model training

By coupling the spatial graph ODE of Eq. (13) with the temporal graph ODE of Eq. (18), we obtain the fully continuous underlying spatiotemporal dynamics. Finally, we obtain the learned node representation y by designing an MLP as the output layer.

To complete the task of link prediction, we need to use the historical information of nodes u and v to determine whether there is an edge (u, v) at time $T + 1$. By capturing the node representation y_{T+1} at time $T + 1$ through CTGNDE, we can easily obtain y^u_{T+1} and y^v_{T+1} and then concatenate the two node representations to obtain the link probability. We regard link prediction as a binary classification problem and select cross entropy loss as the loss function of model training:

$$\mathcal{L} = -\sum_{l\in E_{T+1}} \left(y_l \log(p_l) + (1 - y_l)\log(1 - p_l)\right), \tag{19}$$

Algorithm 1 The Continuous Temporal Graph Neural Differential Equation Network Algorithm

Input: Node feature sequences $X = \{X_1, X_2, \ldots, X_T\}$, adjacency matrix sequences $A = \{A_1, A_2, \ldots, A_T\}$, input length T, model depth K, predicted link set E_{T+1}.
Output: Prediction result.

$H_0 \leftarrow \mathrm{Encoder}(X)$;
$\hat{A} \leftarrow \mathrm{adj\_norm}(A)$;
$H_K \leftarrow \mathrm{ODESolver}(H_0, \frac{\mathrm{d}H_t}{\mathrm{d}t}, K)$; ▷ The ODE function defined in Eq. (12)
$\bar{H}_K \leftarrow \mathrm{Trans}(H_K)$; ▷ The feature transformation defined in Eq. (13)
$\{x_i\}_{i=t_0}^{t_T}, Z_{t_0} \leftarrow \bar{H}_K$;
$S(t) \leftarrow \mathrm{Interp}(x_i)$; ▷ Construct a continuous path
$Z_{T+1} \leftarrow \mathrm{ODESolver}(Z_{t_0}, \frac{\mathrm{d}Z_t}{\mathrm{d}t}, T + 1)$; ▷ The ODE function defined in Eq. (18)
$y_{T+1} \leftarrow \mathrm{Decoder}(Z_{T+1})$; ▷ Update node representation
for $l \in E_{T+1}$ **do**
    Update the edge representation y^l_{T+1} from y_{T+1};
    Calculate the link probability p_l based on y^l_{T+1};
end for

Table 1
Statistics of the datasets used in the experiments.

Datasets	#Nodes	#Edges	#Avg.Degree	#TimeSpan(days)
Ia-Enron	151	50 572	669.8	1137
Ia-Contact	274	28 244	206.2	4
Fb-Forum	899	33 720	75.0	164
Email-Eu	986	332 334	674.1	803
Email-DNC	1891	39 264	41.5	982
UCI	1899	59 835	63.0	196
SO	3262	13 077	8.0	36
BC-Alpha	3783	24 186	12.8	1901
BC-OTC	5881	35 592	12.1	1904
Wikipedia	9227	157 474	34.1	30

where E_{T+1} is the set of links to be predicted, which includes the existing target links $\mathcal{E}_{T+1}$ and random samples of the same number of nonexistent links. p_l is the probability of the existence of link l predicted by the model, and $y_l \in \{0, 1\}$ is the real label of the link indicating whether the link exists or not. The pseudocodes of CTGNDE are illustrated in Algorithm 1.

4. Experiment

In this section, we conduct empirical experiments using a diverse range of 10 distinct datasets to assess the efficacy of our proposed model, CTGNDE, in the context of the link prediction task.

4.1. Datasets

We conducted the experiments using the following 10 real-world network datasets. The specific characteristics and details of these datasets are presented in Table 1.

- **Ia-Enron-Employees**[1] (Ia-Enron): An email communication network among Enron employees, with nodes representing individual employees and edges denoting email exchanges between them. We aim to predict the monthly email traffic between the employees.
- **Ia-Contact**[2]: A social network comprises nodes representing unique individuals, with edges indicating the connections or

[1] https://networkrepository.com/ia-enron-employees.php
[2] https://networkrepository.com/ia-contact.php

relationships between them. We aim to predict changes in interpersonal relationships.

- **Fb-Forum**[3]: A network dataset derived from Facebook forums. Nodes represent users, whereas edges represent their interactions. We use it to predict changes in user interaction over a 2-day period.
- **Email-Eu-Core**[4] (Email-Eu): An email communication network within a European research institution. Nodes represent institution members, and edges indicate email exchanges between them. We use this dataset to predict fortnightly mail traffic between institution members.
- **Email-DNC**[5]:A network of email communications in the 2016 Democratic National Committee email leak. Nodes represent persons in the datasets and edges denote email exchanges between them. We aim to make predictions about email delivery on this dataset.
- **UC Irvine messages**[6] (UCI): A social network of online posts by students from the University of California, Irvine. Nodes represent students, and edges indicate social connections between them. We utilize this dataset to predict the interactions between students.
- **SO**[7]:A collaboration network where nodes represent Stack Overflow users and the edge represents comments from one user to another. We use it to predict interactions between users.
- **Bitcoin Alpha**[8] (BC-Alpha): A who-trusts-whom network among bitcoin users trading on the Alpha platform. Nodes represent users, and edges denote trust relationships, quantified by user ratings on a scale ranging from −10 (total distrust) to +10 (total trust). We use this dataset to determine whether one user will rate another at the next time.
- **Bitcoin OTC**[9] (BC-OTC): Another who-trusts-whom network among bitcoin users trading on the OTC platform, similar to the BC-Alpha dataset.
- **Wikipedia**[10]: A dataset of wiki page edits spanning a month. Each node represents either a user or an edited page, and edges denote interactions between users and the pages they edit. We utilize this dataset to predict the daily user interactions with the pages.

4.2. Baselines

We conducted a comparative analysis between our proposed method and several state-of-the-art graph-based neural network models, including the following:

- **GCN** [20]: A convolutional neural network designed for processing graph-structured data. GCN operates on static graphs and utilizes node features to propagate and aggregate information by learning the neighboring relationships between nodes.
- **GraphSage** [31]: A method specifically designed for representation learning in large-scale graph-structured data. GraphSage employs neighbor sampling and feature aggregation techniques to learn expressive node representations.
- **EvolveGCN** [12]: A dynamic graph neural network capable of effectively handling evolving graph data. EvolveGCN addresses changes in graph structure by incrementally updating the parameters of the graph convolutional layers over time. It offers two variants: EvolveGCN-H, which uses GRU for dynamic modeling, and EvolveGCN-O, which uses LSTM for the same purpose.
- **CGNN** [18]: A continuous neural network designed to handle graph-structured data, wherein it uses ODEs to establish continuous and dynamic representations of nodes. This approach enables the network to effectively capture temporal dependencies and evolving patterns within the graph, facilitating comprehensive and continuous learning of node representations.
- **NCDE** [17]: A framework that empowers continuous-time modeling while leveraging neural networks for dynamic evolution. Neural CDE exhibits flexibility for capturing nonlinear temporal patterns and handling irregular time intervals and missing data.
- **DGCN** [32]: An innovative approach for learning dynamic graph representations based on the GCNs. This method incorporates LSTM to dynamically update the weight parameters of the GCN, effectively capturing the comprehensive global structural insights across all temporal stages of the dynamic graph.
- **SRG** [33]: A novel approach to deep generative modeling. This method combines the concept of super-resolution with conditional normalizing flows, capturing intricate temporal patterns within the evolving graph data to enhance the granularity of link prediction.
- **CTGNDE variants**: **CTGNDE-discrete** integrates the discrete graph propagation process outlined in Eq. (2) with the discrete temporal information aggregation process described in Eq. (14) for node representation learning. We compare it with **CTGNDE-GCN**, which combines GCN and Eq. (14) for node representation learning.

4.3. Experimental settings

In our experimental setup, we partitioned each dataset into training, validation, and test sets in a ratio of 70%, 15%, and 15%, respectively. Furthermore, during training and testing, we randomly selected an equal number of nonexistent links as negative samples from the dataset. The model was trained on the training set, and the best performing model on the validation set was selected for evaluation on the test set.

To assess the performance of different models, we used two representative evaluation metrics: the area under the receiver operating characteristic curve (AUC) and the average precision (AP). Each dataset was trained for 100 epochs, and we conducted 5 independent experiments, randomly dividing the dataset each time, to mitigate bias and obtain average values with standard derivation.

During the experiments, we set the batch size to 8, used the Adam optimizer to optimize the loss function, and assigned a learning rate of 0.002.

We used Python 3.8.10, PyTorch 1.12.1, PyG 2.2.0, and CUDA 12.0 as the computing environment. We executed the CTGNDE model and other comparative methods on our workstation, which featured two Intel(R) Xeon(R) Platinum 8255C 2.50 GHz CPUs and four NVIDIA RTX 2080 Ti GPUs.

4.4. Experimental results and analysis

1. Comparison Results. In this section, we conduct experiments using our proposed approach and baseline approaches on 10 datasets. We use different evaluation metrics to compare the performance of each model. Table 2 presents the AP scores obtained using different methods on each dynamic network, whereas the corresponding AUC scores are presented in Table 3. The performance of our model is derived when the spatial aggregation depth K is set to 6, and based on these results, we make the following observations.

On the one hand, CTGNDE consistently outperforms all the baseline methods across all 10 datasets. Compared with the baseline methods, the CTGNDE model demonstrates an average increase in AP scores of approximately 2.8% and AUC values of around 2.7%. In particular, in the BC-Alpha and BC-OTC datasets, our model surpasses the AP and AUC scores of all the other models by more than nearly 5%. In general,

[3] https://networkrepository.com/fb-forum.php
[4] http://snap.stanford.edu/data/email-Eu-core.html
[5] https://networkrepository.com/email-dnc.php
[6] http://konect.cc/networks/opsahl-ucsocial/
[7] https://archive.org/details/stackexchange
[8] http://snap.stanford.edu/data/soc-sign-bitcoin-alpha.html
[9] http://snap.stanford.edu/data/soc-sign-bitcoin-otc.html
[10] http://snap.stanford.edu/jodie/wikipedia.csv

Table 2
AP (%) comparison of our method and baselines. The best-performing methods are bold faced, and the runner-ups are underlined. ⋆ SRG ran out of memory on SO, BC-Alpha, BC-OTC and Wikipedia.

Methods	Ia-Enron	Ia-Contact	Fb-Forum	Email-Eu	Email-DNC	UCI	SO	BC-Alpha	BC-OTC	Wikipedia
GCN	79.23 ± 0.75	86.48 ± 1.63	75.57 ± 0.37	85.16 ± 0.71	93.72 ± 0.32	76.52 ± 0.43	78.08 ± 0.78	85.60 ± 0.69	86.90 ± 0.45	91.21 ± 0.17
GraphSage	79.29 ± 1.41	86.91 ± 0.59	80.63 ± 0.41	88.05 ± 1.56	93.47 ± 0.13	81.54 ± 0.48	78.31 ± 0.55	87.73 ± 0.66	88.82 ± 0.22	90.69 ± 0.31
EvolveGCN-H	76.11 ± 1.03	85.46 ± 1.33	70.64 ± 0.61	83.54 ± 0.44	91.92 ± 0.74	73.22 ± 0.96	75.93 ± 1.62	86.31 ± 1.10	86.69 ± 0.59	87.95 ± 1.05
EvolveGCN-O	75.43 ± 0.93	84.45 ± 1.80	71.40 ± 0.72	83.65 ± 0.98	92.30 ± 0.55	73.28 ± 0.98	76.13 ± 1.48	84.96 ± 0.92	86.55 ± 0.72	89.00 ± 0.59
CGNN	46.21 ± 1.83	85.11 ± 0.35	77.11 ± 0.27	87.66 ± 0.63	89.91 ± 0.25	75.57 ± 0.25	64.21 ± 0.08	81.68 ± 0.18	82.63 ± 0.26	88.95 ± 0.16
NCDE	60.47 ± 4.02	91.75 ± 1.83	80.97 ± 0.72	82.94 ± 0.51	89.44 ± 2.30	74.94 ± 1.43	77.73 ± 2.09	73.59 ± 3.04	73.13 ± 1.69	90.27 ± 1.21
DGCN	77.12 ± 0.44	95.37 ± 0.28	81.65 ± 0.17	83.43 ± 0.24	94.16 ± 0.19	87.07 ± 0.12	84.85 ± 0.29	88.71 ± 0.14	88.00 ± 0.21	86.96 ± 0.23
SRG⋆	83.66 ± 0.75	95.37 ± 0.22	90.97 ± 0.74	88.17 ± 3.63	89.21 ± 1.30	70.70 ± 1.84	–	–	–	-
CTGNDE-discrete	70.50 ± 1.09	96.95 ± 0.10	83.67 ± 0.54	88.36 ± 0.84	93.92 ± 0.46	74.94 ± 0.22	81.17 ± 0.80	69.41 ± 0.74	79.38 ± 0.37	91.15 ± 0.05
CTGNDE-GCN	74.44 ± 2.80	96.59 ± 0.80	84.51 ± 0.54	84.90 ± 1.36	93.69 ± 1.04	76.34 ± 1.39	81.03 ± 0.87	75.23 ± 4.44	81.91 ± 3.68	92.94 ± 0.47
CTGNDE	**84.93 ± 1.77**	**97.08 ± 0.18**	**91.30 ± 0.53**	**89.86 ± 0.82**	**96.65 ± 0.27**	**90.92 ± 0.64**	**88.07 ± 0.38**	**93.77 ± 0.75**	**93.52 ± 0.53**	**94.71 ± 0.22**

Table 3
AUC (%) comparison of our method and baselines. The best-performing methods are bold faced, and the runner-ups are underlined. ⋆ SRG ran out of memory on SO, BC-Alpha, BC-OTC and Wikipedia.

Methods	Ia-Enron	Ia-Contact	Fb-Forum	Email-Eu	Email-DNC	UCI	SO	BC-Alpha	BC-OTC	Wikipedia
GCN	78.35 ± 0.42	86.75 ± 1.62	75.32 ± 0.21	86.08 ± 0.48	93.12 ± 0.49	76.38 ± 0.27	77.65 ± 0.96	85.28 ± 1.07	86.15 ± 0.91	91.72 ± 0.20
GraphSage	80.94 ± 0.59	87.98 ± 0.36	79.66 ± 0.19	89.00 ± 0.76	93.22 ± 0.19	80.98 ± 0.28	78.16 ± 0.59	87.57 ± 0.53	88.64 ± 0.47	92.62 ± 0.40
EvolveGCN-H	74.04 ± 0.40	85.31 ± 1.26	71.10 ± 0.65	82.74 ± 0.42	91.32 ± 1.11	74.20 ± 0.88	77.02 ± 1.55	87.90 ± 1.07	87.40 ± 0.83	88.29 ± 1.02
EvolveGCN-O	73.13 ± 0.85	83.60 ± 2.11	71.48 ± 0.45	82.66 ± 0.55	91.67 ± 0.59	73.11 ± 1.15	76.48 ± 2.88	84.71 ± 1.32	85.63 ± 1.13	89.81 ± 0.48
CGNN	41.53 ± 3.79	82.92 ± 0.37	74.04 ± 0.15	86.37 ± 0.33	88.90 ± 0.16	72.33 ± 0.30	62.24 ± 0.21	81.83 ± 0.07	82.70 ± 0.18	88.11 ± 0.12
NCDE	60.23 ± 3.20	94.22 ± 0.70	80.90 ± 0.42	84.38 ± 0.90	90.51 ± 1.70	72.86 ± 1.47	73.64 ± 1.85	70.45 ± 3.50	72.94 ± 1.24	90.78 ± 1.45
DGCN	78.78 ± 0.32	96.10 ± 0.22	81.58 ± 0.15	86.80 ± 0.17	94.09 ± 0.22	87.03 ± 0.11	84.54 ± 0.18	88.91 ± 0.14	88.17 ± 0.20	87.89 ± 0.13
SRG⋆	81.94 ± 0.91	95.89 ± 0.63	90.07 ± 0.44	87.32 ± 2.94	87.10 ± 1.85	68.37 ± 2.43	–	–	–	-
CTGNDE-discrete	70.04 ± 1.32	97.37 ± 0.08	83.59 ± 0.43	88.43 ± 0.36	92.81 ± 0.52	73.14 ± 0.32	75.94 ± 0.75	63.17 ± 0.62	74.18 ± 0.41	90.95 ± 0.07
CTGNDE-GCN	75.59 ± 2.15	97.27 ± 0.62	84.98 ± 0.58	85.96 ± 1.04	93.76 ± 0.94	74.84 ± 1.59	75.88 ± 0.51	69.86 ± 6.43	78.41 ± 4.58	94.12 ± 0.36
CTGNDE	**85.93 ± 1.42**	**97.60 ± 0.20**	**91.11 ± 0.72**	**91.49 ± 0.94**	**96.36 ± 0.36**	**90.43 ± 0.53**	**85.46 ± 0.41**	**93.08 ± 0.53**	**92.59 ± 0.61**	**95.59 ± 0.20**

these results visually highlight the efficacy of CTGNDE in tasks related to dynamic link prediction. Furthermore, they provide robust evidence supporting the notion that neural differential equations provide a superior choice for modeling continuously evolving spatiotemporal features. In addition, it is worth mentioning that while SRG exhibited competitive performance on the initial four datasets, the reliance of its design on transformations between adjacency matrices led to a critical drawback: the requirement for significantly larger storage space. This limitation restricts the applicability of the SRG solely to small-scale data scenarios.

On the other hand, our model outperforms the discrete variants of CTGNDE (CTGNDE-discrete and CTGNDE-GCN) on all datasets. This suggests that the two graph ODEs we have continuously designed are more effective in capturing dynamic changes in node representations, leading to improved prediction accuracy. Consequently, these results further substantiate the advantages of the continuous dynamic graph network structure of CTGNDE, which is based on neural differential equations.

2. Statistical Test. To comprehensively assess and contrast the efficacy of CTGNDE against baseline models, we conduct statistical analyses to ascertain if CTGNDE exhibits superior performance. Specifically, we employ the Friedman test [34] to determine whether significant differences exist in the performance across a set of models, assuming the null hypothesis that all models demonstrate equivalent performance. Furthermore, if the null hypothesis of the Friedman test is rejected, the Bonferroni–Dunn test [34] is further applied to determine whether a significant difference exists between the performances of CTGNDE and other models within the comparative set.

According to the comparison results presented in Tables 2 and 3, we select results from six datasets, Ia-Enron, Ia-Contact, Fb-Forum, Email-Eu, Email-DNC, and UCI, to provide support for our study. Moreover, statistical tests are conducted in the four best-performing models: CTGNDE, SRG, DGCN, and GraphSage. These tests are based on the rankings of the models, which are determined by their performance on each dataset as measured by a specific metric. For example, when using the AP score as the evaluation metric on the Ia-Contact dataset, the rankings of CTGNDE, SRG, DGCN, and GraphSage are 1, 2, 3, and 4, respectively. Based on the above description, with the significance

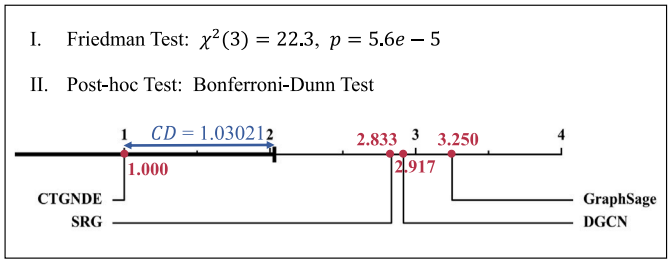

Fig. 3. Results of the statistical test. Label I shows the result of the Friedman test, including χ^2 statistics and the corresponding p-value. Meanwhile, label II illustrates the critical difference diagram derived from the Bonferroni–Dunn test, where the critical difference(CD) is marked in blue, and the average rank of each model is highlighted in red.

level α set to 0.05, the results of the Friedman and Bonferroni–Dunn tests are presented in Fig. 3.

The results show a p-value of $5.6e-5$ obtained from the Friedman test, which is substantially smaller than the predetermined significance level α. This discrepancy prompts the rejection of the null hypothesis, indicating that the four models exhibit distinct performance levels. Upon further comparison using the Bonferroni–Dunn test, it is observed that the average rankings of CTGNDE, SRG, DGCN, and GraphSage are 1, 2.833, 2.917, and 3.25, respectively, indicating that CTGNDE outperforms the other models. Furthermore, the critical difference (CD) obtained from the Bonferroni–Dunn test is 1.03021. Notably, the difference between the average ranking of CTGNDE and other models exceeds this CD. This implies that significant differences exist between CTGNDE and the other models, thereby establishing the superiority of CTGNDE over the baseline models.

3. Complexity Analysis. The computational complexity of our proposed CTGNDE is mainly determined by the construction of the spatial graph ODE component and temporal graph ODE component. We denote N, m, K, and S as the number of nodes, feature dimensions, spatial aggregation depth, and continuous path dimensions, respectively. For each timestamp, the time complexity of the feature propagation stage

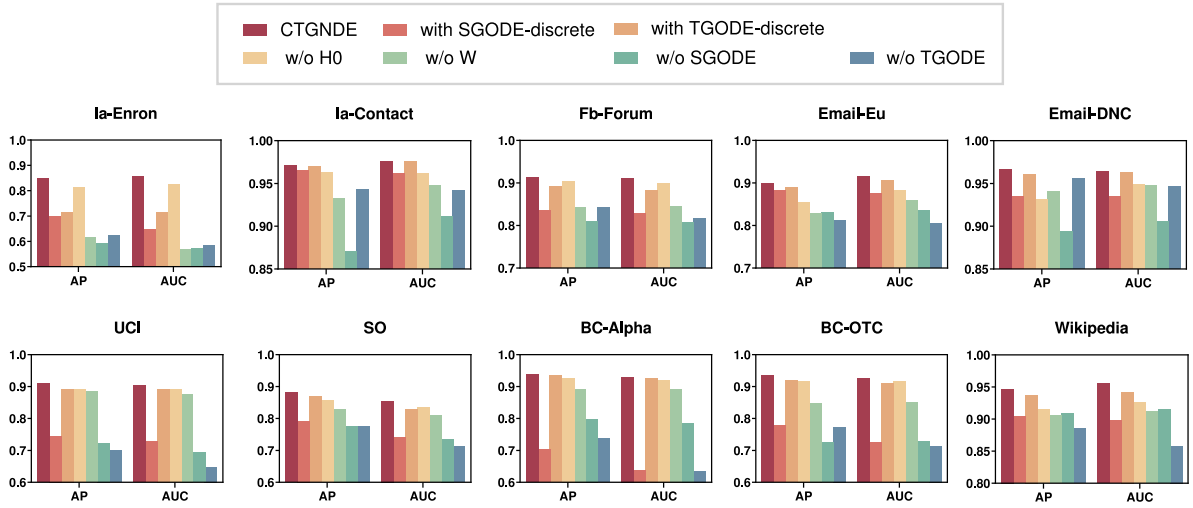

Fig. 4. Results of ablation experiments conducted on all 10 datasets.

in the spatial graph ODE, as presented in Eq. (12), is $\mathcal{O}(KN^2 + KNm)$, and the computational complexity of the feature nonlinear activation is $\mathcal{O}(Nm + Nm^2) = \mathcal{O}(Nm^2)$. Consequently, the time complexity of the proposed spatial graph ODE component is $\mathcal{O}(KN^2 + KNm + Nm^2)$. Similarly, considering the temporal aggregation expressed in Eq. (18), the computational complexity of the temporal graph ODE component is $\mathcal{O}(NmS + Nm) = \mathcal{O}(NmS)$. Based on our empirical observations, m, K, and S are generally smaller than N, and we assume that $m \geq S$. Thus, the overall computational complexity of CTGNDE is $\mathcal{O}(KN^2 + Nm^2)$. Compared with SRG and DGCN, where SRG exhibits a time complexity exceeding $\mathcal{O}(96N^3)$ [33] and the overall time complexity of DGCN is $\mathcal{O}(|E|c_{avg}\log c_{avg})$ [32], our proposed method CTGNDE remains competitive in terms of computational costs.

4.5. Ablation study

To further validate the effectiveness of the various modules within the CTGNDE model, we designed the following six variants of CTGNDE.

- **CTGNDE without H_0** (w/o H_0) removes the initial embedding state during the spatial graph ODE procedure as indicated in Eq. (12).
- **CTGNDE without W** (w/o W) only considers the aggregation of neighbor structural information in the spatial graph ODE module and neglects the alteration of temporal features outlined in Eq. (13).
- **CTGNDE without spatial graph ODE** (w/o SGODE) eliminates the spatial graph ODE module, limiting the model to exclusively perform temporal information aggregation based on Eq. (18).
- **CTGNDE without temporal graph ODE** (w/o TGODE) excludes the temporal graph ODE module, and this variant only relies on Eq. (13) for link prediction.
- **CTGNDE with discrete spatial graph ODE** (with SGODE-discrete), instead of Eq. (13), this variant uses Eq. (2) to discretize the spatial graph ODE process.
- **CTGNDE with discrete temporal graph ODE** (with TGODE-discrete) replaces the continuous temporal graph ODE procedure with the discrete RNN of Eq. (14).

We conducted comprehensive ablation studies on all datasets, and the results are shown in Fig. 4. It clearly indicates that CTGNDE outperforms its variants in terms of both AP and AUC. The results not only unequivocally affirm the validity of each component within our designed model but also clearly indicate the outstanding effectiveness of CTGNDE across diverse datasets, thereby emphasizing the enhanced capability of spatial and temporal graph ODEs in capturing intricate spatiotemporal information within dynamic networks. In addition, the experimental analysis involving different variants of CTGNDE reaffirms the strong interdependence of temporal and spatial characteristics, as disregarding any module (e.g., w/o W) diminishes the predictive capacity of CTGNDE. Moreover, compared with our proposed CTGNDE, the discrete variant model exhibits inferior performance, which is attributable to the continuous model structure we devised based on neural differential equations, enabling it to capture continuous hidden state trajectories and thereby improving prediction accuracy.

4.6. Continuous vs. Discrete

To further validate the effectiveness of the continuous structure of the CTGNDE and assess the robustness of the model through experiments, we defined two variants of CTGNDE as described in Section 4.2: CTGNDE-discrete and CTGNDE-GCN. Subsequently, we compared the performance of CTGNDE and its variants at different depths of the model structure (i.e., different values of t in Eq. (13)), and the results are depicted in Fig. 5.

The observation results indicate that our proposed CTGNDE consistently outperforms its variants on all datasets across different depths of the model structure. This clearly indicates that the continuous model we designed exhibits higher prediction accuracy than the discrete model. In addition, although CTGNDE-discrete performs worse than CTGNDE-GCN on most datasets, the stability of CTGNDE-GCN decreases as the depth of the model structure increases. Notably, on the BC-Alpha and BC-OTC datasets, the performance of CTGNDE-GCN substantially drops, whereas the performances of CTGNDE and CTGNDE-discrete remain stable. This stability is achieved by avoiding parameter layering, thereby eliminating redundant trainable parameters and enhancing the stability of the model. These results also indicate the robustness of our CTGNDE model in mitigating the problem of over-smoothing.

Furthermore, we compared the parametric efficiency of CTGNDE with its discrete variant, as represented by the dashed line in Fig. 5. Overall, CTGNDE has fewer and constant parameters than CTGNDE-GCN, resulting in lower computational overhead. Although CTGNDE has slightly more parameters than CTGNDE-discrete, the improved AUC score confirms the negligible increase in parameter count. In summary, compared with discrete methods, the continuous approach of CTGNDE provides higher prediction accuracy while exhibiting better parametric efficiency.

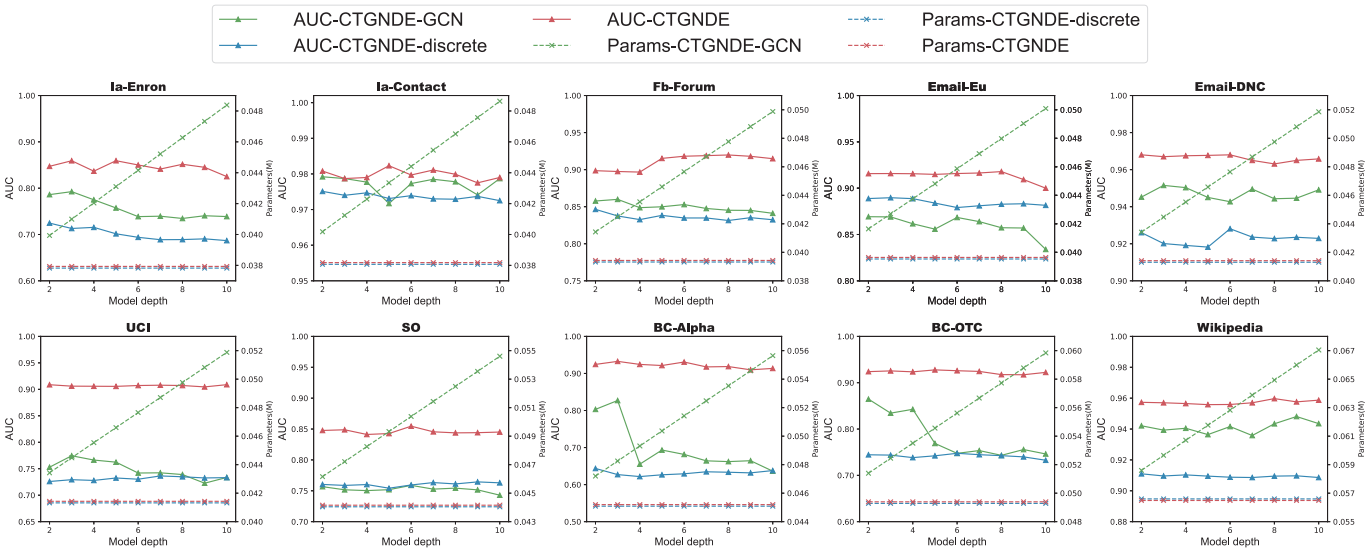

Fig. 5. AUC comparison and model parameters on all 10 datasets using different model depths.

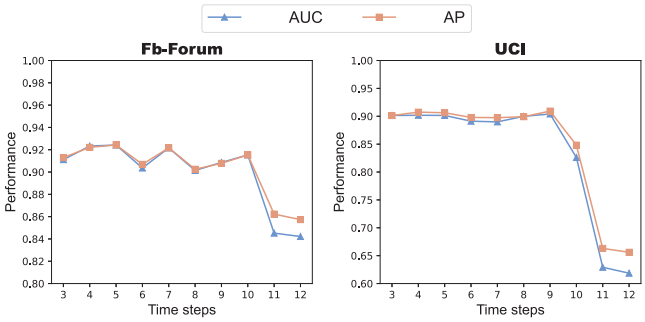

Fig. 6. Performance comparison across various timestep sizes on the Fb-Forum and UCI datasets.

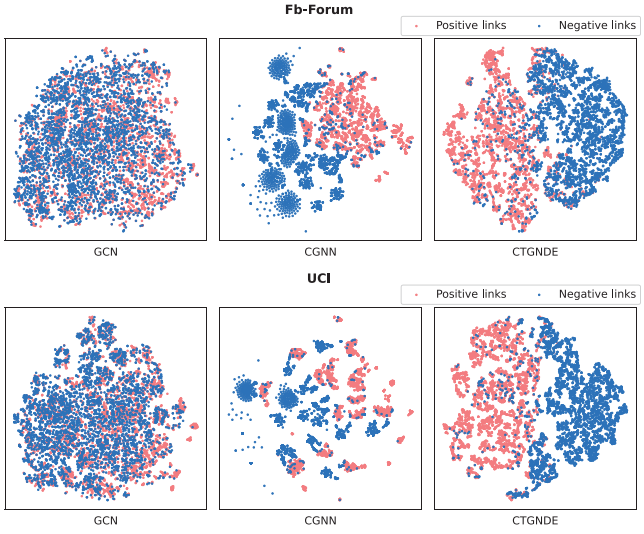

Fig. 7. t-SNE visualization on the Fb-Forum and UCI datasets.

4.7. Parameter analysis

As the choice of historical observation length T significantly influences the extracted features for link prediction, we investigated the sensitivity of hyperparameter T within the context of the CTGNDE model. This empirical analysis was conducted on both the Fb-Forum and UCI datasets, and the results are graphically represented in Fig. 6.

Based on the observations presented in Fig. 6, it becomes evident that both the AUC and AP scores exhibit a noticeable stability with minor fluctuations within a specific range of increasing timesteps. This empirical evidence suggests that a moderate augmentation of the historical observation timestep can enhance the performance of the model. It can also be observed that even if there are fluctuations in the performance of the model with historical timesteps between 3 and 9, the difference in performance compared with the optimal performance model is minimal throughout the process. In the case of social network datasets such as Fb-Forum and UCI, this observation might be because they focus more on short-term social changes and are less influenced by long-term trends. Consequently, a relatively small value of the historical observation length, such as 3, is sufficient to achieve results comparable to those of the optimal performance model.

In a comprehensive assessment, optimal performance on the Fb-Forum dataset is achieved with a historical observation length of 7, whereas for the UCI dataset, the peak performance is manifested at a historical observation length of 9. Furthermore, it is crucial to emphasize that a noteworthy decline in performance is observed beyond a timestep of 10 for the Fb-Forum dataset and 9 for the UCI dataset. This phenomenon may be attributed to an excessive extraction of historical timestamp data, which introduces elevated noise into the informational context, consequently impeding further enhancements in performance. Further investigation is imperative to gain a more comprehensive insight into this identified phenomenon.

4.8. Visualization analysis

To obtain a better illustration of the performance of our proposed CTGNDE, we conducted a visualization analysis using the learned node pair representations. These representations were extracted from the trained CTGNDE and used as features for each edge. Subsequently, we projected these edge features into a two-dimensional space using the t-distributed Stochastic Neighbor Embedding (t-SNE) technique [35]. In our analysis, we compared CTGNDE with GCN and CGNN using the Fb-Forum and UCI datasets. Fig. 7 presents the visualization results for 15% test edges, where positive links are represented in pink and negative links in blue.

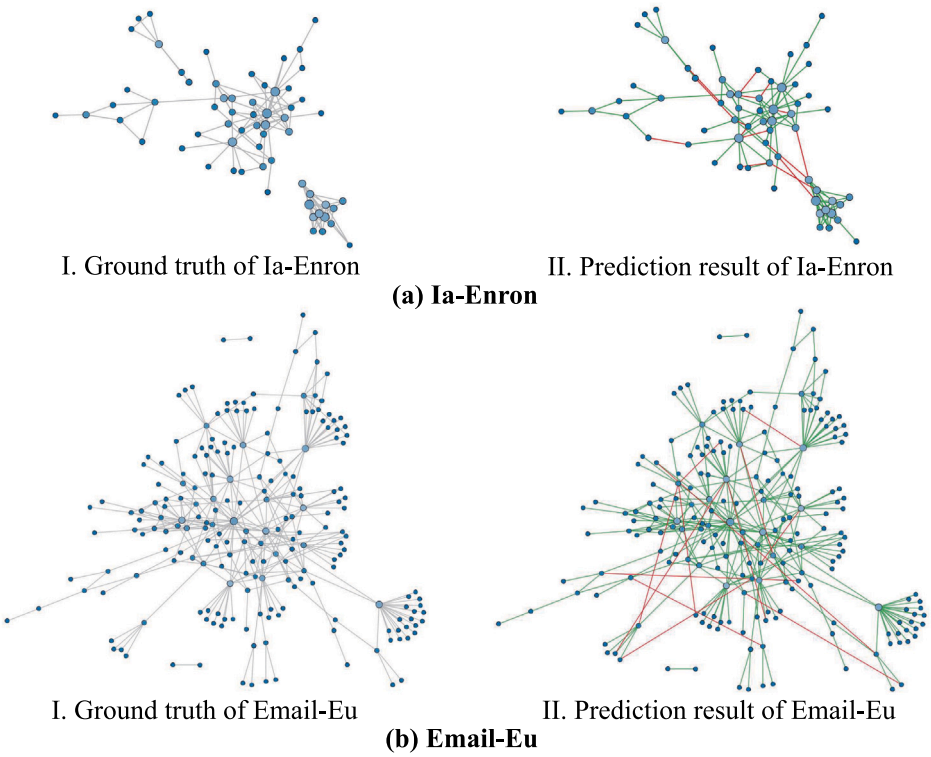

Fig. 8. Case studies on different dynamic networks. Subfigures (a) and (b) correspond to the network graphs of Ia-Enron and Email-Eu, respectively. For each subfigure, label I shows the ground truth network, whereas label II displays the predicted network. Furthermore, correctly predicted edges are highlighted in green, whereas inaccurately predicted edges are marked in red.

Notably, regardless of the dataset Fb-Forum or UCI, it is evident from the visualization results that the edge features learned by CTGNDE can effectively determine whether a link is connected. Furthermore, to evaluate the classification performance, we employed contour coefficients to measure the intracluster distances and intercluster distances. For the Fb-Forum dataset, the contour coefficients of CTGNDE, CGNN, and GCN were calculated as 0.4220, 0.3670, and 0.3769, respectively. Similarly, for the UCI dataset, the contour coefficients of CTGNDE, CGNN, and GCN were computed as 0.4067, 0.3519, and 0.3756, respectively. These results provide additional evidence supporting the superior predictive ability of CTGNDE in representing edge features.

4.9. Case study

To further investigate how the CTGNDE model benefits the link prediction task, we demonstrate two detailed case studies on the Ia-Enron and Email-Eu datasets, as illustrated in Fig. 8. As can be seen from the figure, the real email communication network shows the presence of multiple local tree structures, which is similar to the communication patterns between department heads and members within the research institution. Compared with both the ground-truth and predicted network, our model demonstrates effective accuracy in predicting email exchanges based on historical data. Particularly noteworthy are cases where predictions deviate from actual results; in such cases, the two connected nodes tend to exhibit a substantial distance from each other on the network, indicating a greater spatial separation. This phenomenon is analogous to the research institution connecting two noncommunicating interdepartmental members for a transaction, thus providing results that are consistent with realistic scenarios and are easily interpretable.

To more clearly illustrate the efficacy of CTGNDE in predicting links with precision, we further used the heat maps of adjacency matrices

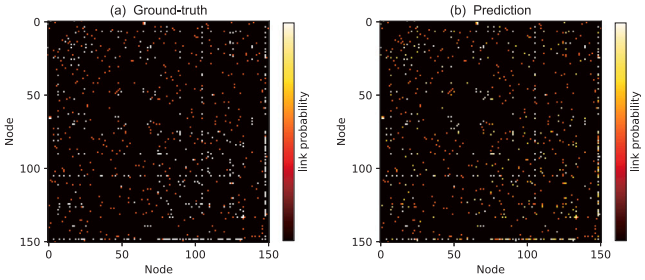

Fig. 9. Subfigures (a) and (b) correspond to the adjacency matrices of the ground truth and the prediction result in Ia-Enron, respectively.

on the Ia-Enron dataset, as depicted in Fig. 9. As our approach relies on negative sampling for link prediction, to determine which samples we choose to predict, we set −1 as the element value to represent the unpredicted links in the adjacency matrix, visualized in black on the heat map. In addition, the color intensity reflects the probability of link existence, where closer to dark red means lower probability and closer to white indicates higher probability. Comparing Figs. 9(a) and 9(b), the proposed CTGNDE consistently demonstrates improved accuracy in predicting link probabilities, regardless of positive or negative samples. This capability provides valuable insights into spam interception. This case highlights the effectiveness of our model in capturing spatiotemporal information, contributing to its robust prediction performance.

5. Conclusion

In this work, our primary focus lies in addressing the task of link prediction in dynamic networks. While numerous existing methods

have achieved impressive results in link prediction by leveraging deep learning techniques, their reliance on discrete neural network architectures hinders their prediction capabilities and computational efficiency. Recognizing these limitations, we introduced a novel temporal link prediction model named CTGNDE, which leverages neural differential equations. CTGNDE encompasses two differential equations, allowing for the capture of both graph structural information and temporal aggregation features. By adopting a continuous approach, the model effectively captures the dynamic trajectories of hidden states. Our empirical evaluation conducted on 10 publicly available datasets indicates the superior performance of our proposed method compared with all baseline approaches. We also confirmed the statistical significance of the performance of CTGNDE over the baseline model. Furthermore, our results provide additional evidence supporting the enhanced prediction accuracy and lower computational overhead offered by CTGNDE contrary to discrete methods. In addition, we further validated the effectiveness of CTGNDE through case studies, affirming its potential as a valuable tool in link prediction tasks for dynamic networks.

In our future research endeavors, we will focus on achieving more accurate node feature representations for the task of dynamic graph link prediction. Our future efforts will be directed to two primary areas. First, we aspire to augment the capabilities of the CTGNDE model to accommodate nodes that evolve over time within dynamic graphs. Second, we will explore the expansion of our model to handle weighted dynamic link prediction. Through these works, we aim to derive superior-quality low-dimensional representations of nodes, thereby enhancing the overall effectiveness of our link prediction methodologies. As noted in Section 4.7 the performance of the model experiences a significant decline when the historical observation timestep exceeds 10. This phenomenon also requires to further scholarly investigation.

CRediT authorship contribution statement

Liyi Huang: Writing – original draft, Visualization, Software, Resources, Methodology, Investigation, Formal analysis, Data curation, Conceptualization. **Bowen Pang:** Writing – review & editing, Validation, Software, Resources. **Qiming Yang:** Validation, Software, Resources, Data curation. **Xiangnan Feng:** Writing – review & editing, Validation, Resources, Data curation. **Wei Wei:** Writing – review & editing, Supervision, Resources, Investigation, 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.

References

[1] L.A. Adamic, E. Adar, Friends and neighbors on the web, Soc. Netw. 25 (3) (2003) 211–230.

[2] M. Gosak, R. Markovič, J. Dolenšek, M.S. Rupnik, M. Marhl, A. Stožer, M. Perc, Network science of biological systems at different scales: A review, Phys. Life Rev. 24 (2018) 118–135.

[3] M.G. Bell, Y. Iida, et al., Transportation Network Analysis, Wiley Online Library, 1997.

[4] S.A. Myers, A. Sharma, P. Gupta, J. Lin, Information network or social network? The structure of the Twitter follow graph, in: Proceedings of the 23rd International Conference on World Wide Web, 2014, pp. 493–498.

[5] Y. Sun, R. Barber, M. Gupta, C.C. Aggarwal, J. Han, Co-author relationship prediction in heterogeneous bibliographic networks, in: 2011 International Conference on Advances in Social Network Analysis and Mining, IEEE, 2011, pp. 121–128.

[6] S. Daminelli, J.M. Thomas, C. Durán, C.V. Cannistraci, Common neighbours and the local-community-paradigm for topological link prediction in bipartite networks, New J. Phys. 17 (11) (2015) 113037.

[7] S. Wu, Y. Tang, Y. Zhu, L. Wang, X. Xie, T. Tan, Session-based recommendation with graph neural networks, in: Proceedings of the AAAI Conference on Artificial Intelligence, Vol. 33, No. 01, 2019, pp. 346–353.

[8] M. Zhang, Y. Chen, Weisfeiler-lehman neural machine for link prediction, in: Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, 2017, pp. 575–583.

[9] M. Zhang, Y. Chen, Link prediction based on graph neural networks, Adv. Neural Inf. Process. Syst. 31 (2018).

[10] L. Cai, J. Li, J. Wang, S. Ji, Line graph neural networks for link prediction, IEEE Trans. Pattern Anal. Mach. Intell. 44 (9) (2021) 5103–5113.

[11] Y. Seo, M. Defferrard, P. Vandergheynst, X. Bresson, Structured sequence modeling with graph convolutional recurrent networks, in: Neural Information Processing: 25th International Conference, ICONIP 2018, Siem Reap, Cambodia, December 13-16, 2018, Proceedings, Part I 25, Springer, 2018, pp. 362–373.

[12] A. Pareja, G. Domeniconi, J. Chen, T. Ma, T. Suzumura, H. Kanezashi, T. Kaler, T. Schardl, C. Leiserson, Evolvegcn: Evolving graph convolutional networks for dynamic graphs, in: Proceedings of the AAAI Conference on Artificial Intelligence, Vol. 34, No. 04, 2020, pp. 5363–5370.

[13] E. Hajiramezanali, A. Hasanzadeh, K. Narayanan, N. Duffield, M. Zhou, X. Qian, Variational graph recurrent neural networks, Adv. Neural Inf. Process. Syst. 32 (2019).

[14] R.T. Chen, Y. Rubanova, J. Bettencourt, D.K. Duvenaud, Neural ordinary differential equations, Adv. Neural Inf. Process. Syst. 31 (2018).

[15] M. Jin, Y. Zheng, Y.-F. Li, S. Chen, B. Yang, S. Pan, Multivariate time series forecasting with dynamic graph neural ODEs, IEEE Trans. Knowl. Data Eng. (2022).

[16] Q. Li, Z. Han, X.-M. Wu, Deeper insights into graph convolutional networks for semi-supervised learning, in: Proceedings of the AAAI Conference on Artificial Intelligence, Vol. 32, No. 1, 2018.

[17] P. Kidger, J. Morrill, J. Foster, T. Lyons, Neural controlled differential equations for irregular time series, Adv. Neural Inf. Process. Syst. 33 (2020) 6696–6707.

[18] L.-P. Xhonneux, M. Qu, J. Tang, Continuous graph neural networks, in: International Conference on Machine Learning, PMLR, 2020, pp. 10432–10441.

[19] A. Divakaran, A. Mohan, Temporal link prediction: A survey, New Gener. Comput. 38 (2020) 213–258.

[20] T.N. Kipf, M. Welling, Semi-supervised classification with graph convolutional networks, 2016, arXiv preprint arXiv:1609.02907.

[21] D. Xu, C. Ruan, E. Korpeoglu, S. Kumar, K. Achan, Inductive representation learning on temporal graphs, 2020, arXiv preprint arXiv:2002.07962.

[22] Z. Xie, R. Zhu, J. Liu, G. Zhou, J.X. Huang, TARGAT: A time-aware relational graph attention model for temporal knowledge graph embedding, IEEE/ACM Trans. Audio Speech Lang. Process. (2023).

[23] P. Kidger, On neural differential equations, 2022, arXiv preprint arXiv:2202.02435.

[24] Y. Rubanova, R.T. Chen, D.K. Duvenaud, Latent ordinary differential equations for irregularly-sampled time series, Adv. Neural Inf. Process. Syst. 32 (2019).

[25] Z. Han, Z. Ding, Y. Ma, Y. Gu, V. Tresp, Learning neural ordinary equations for forecasting future links on temporal knowledge graphs, in: Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing, 2021, pp. 8352–8364.

[26] E. Dupont, A. Doucet, Y.W. Teh, Augmented neural odes, Adv. Neural Inf. Process. Syst. 32 (2019).

[27] A. Norcliffe, C. Bodnar, B. Day, N. Simidjievski, P. Liò, On second order behaviour in augmented neural odes, Adv. Neural Inf. Process. Syst. 33 (2020) 5911–5921.

[28] M. Poli, S. Massaroli, J. Park, A. Yamashita, H. Asama, J. Park, Graph neural ordinary differential equations, 2019, arXiv preprint arXiv:1911.07532.

[29] Z. Fang, Q. Long, G. Song, K. Xie, Spatial-temporal graph ode networks for traffic flow forecasting, in: Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining, 2021, pp. 364–373.

[30] F. Wu, A. Souza, T. Zhang, C. Fifty, T. Yu, K. Weinberger, Simplifying graph convolutional networks, in: International Conference on Machine Learning, PMLR, 2019, pp. 6861–6871.

[31] W. Hamilton, Z. Ying, J. Leskovec, Inductive representation learning on large graphs, Adv. Neural Inf. Process. Syst. 30 (2017).

[32] C. Gao, J. Zhu, F. Zhang, Z. Wang, X. Li, A novel representation learning for dynamic graphs based on graph convolutional networks, IEEE Trans. Cybern. (2022).

[33] Y. Yin, Y. Wu, X. Yang, W. Zhang, X. Yuan, Super resolution graph with conditional normalizing flows for temporal link prediction, IEEE Trans. Knowl. Data Eng. (2023).

[34] J. Demšar, Statistical comparisons of classifiers over multiple data sets, J. Mach. Learn. Res. 7 (2006) 1–30.

[35] L. Van der Maaten, G. Hinton, Visualizing data using t-sne, J. Mach. Learn. Res. 9 (11) (2008).