Dynamic Co-embedding Model for Temporal Attributed Networks

Shaowei Tang, Zaiqiao Meng, and Shangsong Liang

***Abstract*—In this paper, we study the problem of embedding temporal attributed networks, with the goal of which is to learn dynamic low-dimensional representations over time for temporal attributed networks. Existing temporal network embedding methods only learn the representations for nodes, which are unable to capture the dynamic affinities between nodes and attributes. Moreover, existing co-embedding methods that learn the static embeddings of both nodes and attributes cannot be naturally utilized to obtain their dynamic embeddings for temporal attributed networks. To address these issues, we propose the Dynamic Co-embedding model for Temporal Attributed Networks based on the dynamic stochastic state space framework. Our model captures the dynamics of an temporal attributed network by modeling the abstract belief states representing the condition of the nodes and attributes of current time step, and predicting the transitions between temporal abstract states of two successive time steps. Our model is able to learn embeddings for both nodes and attributes based on their belief states at each time step of the temporal attributed network, while the state transition tendency for predicting the future network can be tracked and the affinities between nodes and attributes can be preserved. Experimental results on real-world networks demonstrate that our model achieves substantial performance gains in several static and dynamic graph mining applications compared with the state-of-the-art static and dynamic models.**

***Index Terms*—Temporal Attributed Network; Network Embedding; Variational Auto-encoder, Dynamic Co-embedding**

I. INTRODUCTION

NETWORK embedding techniques, aiming at learning low-dimensional representations for networks, have attracted a surge of attention in both researches and industries recently. Many approaches [1–7] for network embeddings mainly focus on encoding a network by node representations (also called node embeddings). A number of down-stream network analysis tasks, e.g., node classification [1, 3], link prediction [8] and community detection [2, 9], have been shown to benefit from the learned embeddings of the nodes obtained by these network embedding approaches.

However, previous works mainly focus on embedding nodes for static networks [1–4, 9], and ignore the fact that most of the real world networks evolve over time, i.e., with the edges between nodes and the node-to-attribute associations changing over time. Moreover, in many real world applications, such as dynamic user profiling [10] and dynamic link prediction [11], the learned embeddings are required to be able to track the evolution of the networks. For instance, the goal of the user profiling task is to learn the semantic representations of users and words to generate relevant keywords to represent a user's profile. Since the relevant keywords that describe users' status and interests would change over time, the learned embeddings of users and keywords should capture and distinguish such changes [10]. *Accordingly, in this paper, we study the problem of learning dynamic representations for* ***Temporal Attributed Networks*** *so that the evolving trends of the network topology and the node attributes can be effectively captured.*

In order to capture the dynamic changes for temporal networks, recently many temporal network embedding methods [6, 7, 11–13] have been proposed to learn dynamic low-dimensional vector representations of nodes. These methods normally optimize an objective function of preserving both the static structural and temporal information, and thereby the dynamic embeddings of nodes capturing the evolution of network can be obtained. With additional temporal information captured by these methods, the learned dynamic embeddings are able to boost the performance of many real-world dynamic network analysis and prediction tasks, such as dynamic link prediction [5–7, 14] and dynamic network visualization [6, 7, 12].

However, most of the existing temporal network embedding methods still suffer from a number of limitations: (1) These methods learn representations of nodes only, resulting in the fact that temporal affinities between nodes and attributes in the dynamic environment cannot be measured. However, capturing such affinities is of great importance to the success of many graph mining tasks, as the correlation between two different types of entities, e.g., user-word correlation in user profiling task [10], and node-attribute correlation in attribute inference task [15], needs to be quantitatively measured. (2) How to explicitly capture the evolutionary tendencies of the network from a given time step to the next time step is still unknown. Yet tracking such transition tendency is an effective way to predict the future states and trends from the historical states and records [16].

Accordingly, to alleviate the aforementioned limitations in

This work was partly supported by the National Natural Science Foundation of China under Grant 61906219.

Shaowei Tang is with the School of Computer Science and Engineering, Sun Yat-sen University, Guangzhou, China; and the Guangdong Key Laboratory of Big Data Analysis and Processing, Guangzhou, China. (email: tangshaowei1@gmail.com).

Zaiqiao Meng is with the School of Computing Science, University of Glasgow, Glasgow, UK. (e-mail: zaiqiao.meng@gmail.com).

Shangsong Liang is with the School of Computer Science and Engineering, Sun Yat-sen University, Guangzhou, China; the Guangdong Key Laboratory of Big Data Analysis and Processing, Guangzhou, China; and the Department of Machine Learning, Mohamed bin Zayed University of Artificial Intelligence, Abu Dhabi, UAE. (email: liangshangsong@gmail.com)

Shangsong Liang is the corresponding author of the paper. Shaowei Tang and Zaiqiao Meng contributed equally to the paper.

the existing methods, we propose a **D**ynamic **C**o-embedding model for **T**emporal **A**ttributed **N**etworks (abbreviated as DCTAN) that learns the dynamic representations of both nodes and attributes at each time step, such that the structure, nodes, attributes, and the temporal information of the dynamic attributed networks can be preserved. Specifically, our DCTAN utilizes two belief state encoders and inference networks to encode the temporal information into belief state codes and infer the distributions of the embeddings, two stochastic state transition and smoothing networks to dynamically embed both nodes and attributes of a temporal attributed network into the same semantic space, and a decoder network to reconstruct the target temporal attributed network based on the inferred embeddings. With the five well-designed specific neural networks, our DCTAN aims to preserve not only the proximities in nodes but also the affinities between nodes and attributes in the dynamic environment. Meanwhile, the state transition tendency can be tracked by training two state transition networks, which use the embeddings at the current time step to predict those at the next time step. Motivated by the successful application of VAE-based models like Dynamic Stochastic State Space Model [16] in learning representations for temporal data, our DCTAN extends VAE to effectively incorporate structural information (i.e., edges), graph information (i.e., affinities between nodes and attributes), and temporal information (i.e., tracking the evolving tendency) in a co-embedding way so as to learn high-quality representations for both nodes and attributes.

The learned dynamic representations of nodes and attributes by our DCTAN can service not only for traditional network analysis tasks (e.g. link reconstruction and attribute inference), but also for predictive tasks (e.g., predicting the links and node-attribute associations at the next time step). Furthermore, the obtained evolution tendency of dynamic network at two successive time steps by our DCTAN can help to predict the status (i.e., the links between nodes and attributes of nodes) of the target network at the next time step based on the learned embeddings of the current time step. In order to verify the effectiveness of our proposed methods, we conduct experiments on several network datasets. Our experimental results show that our models can achieve the best performance in link prediction and reconstruction tasks compared with state-of-the-art baseline methods, and yields superior performance in attribute inference and prediction tasks.

Our contributions in this paper can be summarized as follows:

(1) We propose the DCTAN, a novel Dynamic Co-embedding model for Temporal Attributed Networks, to learn the temporal representations of both nodes and attributes at each time step based on the dynamic stochastic state space model, such that dynamic affinities between nodes and attributes can be effectively captured.
(2) Based on the abstract belief states learned by the Long Short-Term Memory (LSTM) recurrent neural networks, we deploy two stochastic state smoothing networks to predict the transitions between belief states of two successive time steps so that the evolving tendency of the network can be captured.
(3) Through DCTAN as an example, we would provide insight and inspiration on how to extend VAE architecture to dynamically learn graph representations for evolving networks, especially the non-plain ones containing rich auxiliary information (e.g., node attributes).
(4) We conduct experiments on several real world dynamic network datasets in terms of two network analysis tasks (e.g., link reconstruction and attribute inference) and two prediction tasks (e.g., link and node-to-attribute association prediction). Experimental results show that our DCTAN outperforms the baseline methods in these tasks and validate that our DCTAN can learn high-quality representations at each time step for dynamic network and effectively capture the transition tendency of the embeddings and dynamic affinities between nodes and attributes over time.

The remainder of the paper is organized as follows. In § II we briefly review the related work. § III introduces the notations and formally defines the problem to be addressed. § IV details our proposed models. § V describes the experimental setup. § VI presents the experimental results and analyses. Finally, we conclude in § VII.

II. RELATED WORK

In what follows, we briefly discuss three lines of related work, plain network embedding, attributed network embedding, and temporal embedding.

A. Plain Network Embedding

Network embedding methods [17–19] for plain networks try to preserve topological information of network in the learned network representations. For example, DeepWalk [1] and Node2vec [2] learn node embeddings that capture the local structural information of the plain networks. These two embedding methods train their models using the Skip-Gram framework [20] by generating random walks to construct the sequences of nodes and then optimizing a likelihood of predicting the surrounding nodes in the sequences. The LINE [3] model further improves it by considering both the first-order and second-order proximities between nodes. Wang et al. [21] apply the deep neural networks to capture the highly non-linear structure and preserve the global and local structure of the network. In order to incorporate the community structure of network into result embeddings, Wang et al. [9] propose a model to preserve both of the microscopic and community structures. However, these methods ignore attributes of nodes that help to provide useful information for encoding nodes as vectors.

B. Attributed Network Embedding

Many attributed network embedding methods are proposed to learn representations via both structural information and auxiliary information, such as the attributes and the textual contents of nodes [22]. For instance, Huang et al. [23] propose a model to learn the joint embeddings of nodes by matrix factorization technique with the graph Laplacian to combine

the topological structure and attributes. Graph convolutional neural network models (GCNs) [4, 24, 25] are variants of traditional convolutional neural networks. These models effectively aggregate features from local nodes to learn the node embeddings in attributed networks. HYPER-SAGNN [26] is a new self-attention based graph neural network, which learns node embeddings to extract patterns among higher-order interaction for hypergraphs. Zhang et al. [27] propose a customized deep network model called ANRL, which learns node embeddings via both topological structure and attribute information. More recently, to further capture the affinities between nodes and attributes, Meng et al. [15, 28] propose a variational auto-encoder which learns the embeddings of both nodes and attributes and represent the learned embeddings by Gaussian distributions. Fang et al. [29] further extend it into the Hyperspherical space to obtain more effective representations of attributed networks. However, the above methods are restricted to deal with static networks, while most of real world networks are not static but continuously evolve with the changes of structure and auxiliary information.

C. Temporal Embedding

Temporal embedding that captures dynamic changes for real time-evolving data has attracted increasing attention recently in many fields [30–33]. For example, Kim et al. [34] and Hamilton et al. [33] model semantic changes of word by splitting the data into separate time bins and train the model by some word embedding algorithms, such as word2vec [20]. Bamler et al. [31] propose a dynamic Skip-Gram model to learn the probabilistic word embeddings. The DUWE [10] is a model that learns dynamic representations of users and words to retrieve top-K relevant and diversified keywords to profile users' dynamic expertise. Recently, Gregor et al. [16] propose a dynamic stochastic state space model called TD-VAE to learn representations containing explicit beliefs about states several steps into the future and track the transition tendency of states at different time steps. However, they cannot be directly applied to co-embed both nodes and attributes for a temporal attributed network which contains both the topological structure and attribute information while the network continuously evolves over time.

In order to deal with temporal networks, a number of temporal network embedding approaches have been proposed. Zhu et al. [5] propose a temporal latent space model to predict the temporal link probability of node pairs for the link prediction task in temporal networks. Zhou et al. [6] present a model to track evolution patterns of triads for plain networks by their triadic closure process. Li et al. [13] build an online model based on an offline model to maintain the freshness of embedding on nodes and attributes by leveraging matrix perturbation theory. Zuo et al. [7] propose a Hawkes process based Temporal Network Embedding (HTNE) model that takes the full historical neighbourhood formation process into account. Trivedi et al. [35] propose a model that learns non-linear evolving entity representations for entities and relations over time by a deep evolutionary knowledge network. Jin et al. [36] propose Recurrent Event Network (RE-Net) for modeling complex event sequences, which consists of a recurrent event encoder and a neighborhood aggregator. Pareja et al. [37] design a architecture, which uses RNN to evolve the parameters of GCN, for handling the problem of temporal network embedding. TGAT [38] is the temporal graph attention layer for efficiently aggregating temporal neighborhood features as well as for learning time-feature interactions. DyREP [39] posits representation learning as a latent mediation process bridging two observed processes namely– dynamics of the network (realized as topological evolution) and dynamics on the network (realized as activities between nodes). All of the temporal network embedding algorithms just aim to learn dynamic representations of nodes only, however, how to capture the dynamic affinities between nodes and attributes still remains challenging. In addition, different between previous models, our model has a explicit objective for tracking the evolving tendency of network in dynamic environment.

III. PRELIMINARIES

In this section, we first introduce the notations and definitions used in the paper (§ III-A), and then formulate the problem that we address in the paper (§ III-B). Next, we briefly describe the dynamic stochastic state space model, which exploits the variational auto-encoder (VAE) and the sequence state model to learn the stochastic latent states for dynamic observations (§ III-C).

A. Notations and Definitions

In this paper, we use *boldface uppercase alphabets* to denote 2-dimensional matrices or 3-dimensional tensors. E.g., a 2-dimensional adjacency matrix of the network can be denoted by $\mathbf{A}$, where $\mathbf{A}_i$ is a row vector of matrix $\mathbf{A}$ indicating the weights of neighbours of user i. We use *normal alphabets* to denote the scalars (e.g., the total number of nodes in the network is denoted by N), while sets are denoted by *calligraphy typeface alphabets* (e.g., the set of static networks for a temporary network is represented by $\mathcal{G}$). The main notations used across our paper are shown in Table I.

In what follows, we define two categories of networks: the static attributed networks and the temporal attributed networks, which are referred in the paper.

Definition 1: **Static Attributed Network.** Let $\mathcal{G}_t = \{\mathbf{A}_t, \mathbf{X}_t\}$ be a static attributed network, where $\mathbf{A}_t \in \mathbb{R}^{N\times N}$ is the adjacency matrix with N being the number of nodes, and $\mathbf{X}_t \in \mathbb{R}^{N\times M}$ is the node attribute matrix with M being the number of attributes.

A temporal attributed network can be represented as a set of sequential static attributed networks over different snapshots, which is defined as:

Definition 2: **Temporal Attributed Network.** A temporal attributed network $\mathcal{G}$ is represented as a sequence of attributed networks over time, i.e. $\mathcal{G}_{\leq T} = \{\mathcal{G}_0, \ldots, \mathcal{G}_T\}$, recording the dynamic changes of edges and node attributes by each $\mathcal{G}_t$ with the time steps evolving from 0 to T, where T is the number of time steps.

Figure 1 shows a toy example of the temporal attributed network with 34 nodes and 5 attributes. As time evolves,

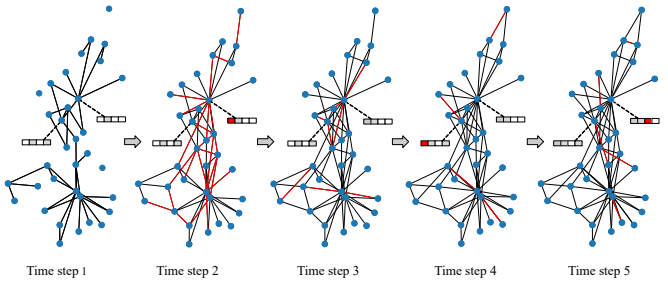


Fig. 1. **A toy example of a temporal attributed network with 34 nodes and 5 attributes evolving over five time steps. At each time step, links between two nodes and associations between nodes and attributes are added or removed randomly, compared with other time steps. To show the dynamic changes of the network over time clearly, newly created links and added attributes are marked by red lines and small red rectangles, respectively**.

edges between nodes and associations between nodes and attributes are constantly added and removed, where evolving tendency should be captured for predicting the changes in the future. However, the traditional network embedding methods can neither track such evolving tendency non-trivially, nor dynamically capture the affinities between nodes and attributes.

B. Problem Statement

The problem of *temporal attributed network co-embedding* we aim to address in this paper can be defined as follows:

Problem 1: **Temporal Attributed Network Co-embedding.**
Given a temporal attributed network $\mathcal{G}_{\leq T} = \{\mathcal{G}_0, \mathcal{G}_1, \ldots, \mathcal{G}_T\}$ with T time steps, the goal of a Temporal Attributed Network Co-embedding algorithm is to learn a mapping function Ξ:

$$\mathcal{G}_{\leq T} \xrightarrow{\Xi} \mathbf{Z}^{\mathcal{N}}_{\leq T}, \mathbf{Z}^{\mathcal{A}}_{\leq T}, \tag{1}$$

where $\mathbf{Z}^{\mathcal{N}}_{\leq T} = \{\mathbf{Z}^{\mathcal{N}}_0, \mathbf{Z}^{\mathcal{N}}_1, \ldots, \mathbf{Z}^{\mathcal{N}}_T\}$ and $\mathbf{Z}^{\mathcal{A}}_{\leq T} = \{\mathbf{Z}^{\mathcal{A}}_0, \mathbf{Z}^{\mathcal{A}}_1, \ldots, \mathbf{Z}^{\mathcal{A}}_T\}$ are the learned node latent embeddings and attribute latent embeddings at each time step, respectively. Specifically, the mapping function Ξ should satisfy the following: (1) The topological structure information of nodes and the associations between nodes and attributes can be preserved by the learned embeddings as much as possible. (2) The function Ξ is able to capture the temporal information of the dynamic network such that the observations of graph after time step T can be predicted by the learned embedding at the last time step T, i.e., the links and node attributes at time step $T+1$ can be predicted according to the embeddings of the previous time step T.

C. The Dynamic Stochastic State Space Model

We now briefly describe the dynamic stochastic state space model [16] that our proposed DCTAN builds based on. The dynamic stochastic state space model [16] is extended from the state space model [40] by using a modified evidence lower bound (ELBO) to learn latent stochastic states and stochastic transitions between states.

Suppose we have an observation sequence $\mathbf{O}_{\leq T} = (\mathbf{O}_0, \ldots, \mathbf{O}_T)$, and let $\mathbf{S}_{\leq T} = (\mathbf{S}_0, \ldots, \mathbf{S}_T)$ be the corresponding latent state sequence of these observations. We

TABLE I
MAIN NOTATIONS USED ACROSS THE WHOLE PAPER.

Symbol	Description
$\mathcal{G}$	a temporal attributed network
$\mathcal{G}_t$	snapshot of temporal network at t
N	number of nodes
M	number of attributes
T	number of time steps
$\mathbf{A}_t \in \mathbb{R}^{N\times N}$	adjacency matrix of network at t
$\mathbf{X}_t \in \mathbb{R}^{N\times M}$	attribute matrix of nodes at t
D	dimension of latent embeddings
$\mathbf{Z}^{\mathcal{N}}_t \in \mathbb{R}^{N\times D}$	representation matrix for nodes at t
$\mathbf{Z}^{\mathcal{A}}_t \in \mathbb{R}^{M\times D}$	representation matrix for attributes at t
B	size of belief code
$\mathbf{B}^{\mathcal{N}}_t \in \mathbb{R}^{N\times B}$	belief codes for nodes at t
$\mathbf{B}^{\mathcal{A}}_t \in \mathbb{R}^{M\times B}$	belief codes for attributes at t

can obtain an ELBO, by introducing a variational posterior $q(\mathbf{S}\,|\,\mathbf{O})$ over the previous states given the observations:

$$\log p(\mathbf{O}_{\leq T}) \geq \mathop{\mathbb{E}}_{\mathbf{S}_{\leq T}\sim q(\mathbf{S}_{\leq T}|\mathbf{O}_{\leq T})}\Big[\sum_{t=1}^{T} \log p\left(\mathbf{O}_t|\mathbf{S}_t\right) + \log p\left(\mathbf{S}_t|\mathbf{S}_{t-1}\right) - \log q\left(\mathbf{S}_t|\mathbf{S}_{t-1}, \phi(\mathbf{O}_{\leq \mathbf{T}})\right)\Big], \tag{2}$$

where ϕ is a function of $(\mathbf{O}_0, \ldots, \mathbf{O}_t)$ for filtering posteriors or the entire sequence $\mathbf{O}_{\leq T}$ for smoothing posteriors. Here the joint state and observation likelihood can be written as $p\left(\mathbf{O}_{\leq t}, \mathbf{S}_{\leq t}\right) = \prod_{t'=1}^{t} p\left(\mathbf{S}_{t'} \mid \mathbf{S}_{t'-1}\right) p\left(\mathbf{O}_{t'} \mid \mathbf{S}_{t'}\right)$, and the variational posterior $q(\mathbf{S}_{\leq t} \mid \mathbf{O}_{\leq t})$ has a mean-field factorization form $q(\mathbf{S}_{\leq t}\,|\,\mathbf{O}_{\leq t}) = \prod_{t'=1}^{t} q(\mathbf{S}_{t'}\,|\,\mathbf{S}_{t'-1}, \phi(\mathbf{O}_{\leq t}))$. By further introducing an online belief state sequence $\mathbf{B}_{\leq \mathbf{T}} = \{\mathbf{B}_0, \ldots, \mathbf{B}_T\}$ with $\mathbf{B}_t$ being defined as $\mathbf{B}_t = g(\mathbf{B}_{t-1}, \mathbf{O}_t)$, where g is a recurrent neural network, and decomposing the data likelihood as: $\log p(\mathbf{O}_{\leq \mathbf{t}}) = \sum_{t'=1}^{t} \log p\left(\mathbf{O}_{t'}|\mathbf{O}_{<t'}\right)$, we obtain the following belief-based ELBO for state-space models:

$$\log p\left(\mathbf{O}_t|\mathbf{O}_{<t}\right) \geq \mathop{\mathbb{E}}_{\substack{\mathbf{S}_t\sim\mathbf{B}_t \\ \mathbf{S}_{t-1}\sim\mathbf{S}_t,\mathbf{B}_t,\mathbf{B}_{t-1}}}\big[\log p\left(\mathbf{O}_t|\mathbf{S}_t\right) + \log p\left(\mathbf{S}_{t-1}|\mathbf{B}_{t-1}\right) + \log p\left(\mathbf{S}_t|\mathbf{S}_{t-1}\right) - \log p\left(\mathbf{S}_t|\mathbf{B}_t\right) - \log q\left(\mathbf{S}_{t-1}|\mathbf{S}_t, \mathbf{B}_{t-1}, \mathbf{B}_t\right)\big]. \tag{3}$$

Instead of smoothing the state information for two consecutive times $t-1$ and t, one can also learn a **jumpy** state-to-state model by choosing the distribution of times (e.g. uniformly), which leads to the Temporal Difference Variational Auto-Encoder (TD-VAE) model [16]. In TD-VAE model, the belief state $\mathbf{B}_t$ acts a role of smoothing the stochastic states between two time steps, and thereby the temporal abstraction from temporally separated time points can be learned without back-propagating through the entire time interval of the sequence observation. However, we need not to consider the jumpy states in our temporal attributed network embedding problem, but rather consider the state of the temporal network smoothly transiting between two consecutive time steps.

Figure 2 shows the graphical representation of the dynamic

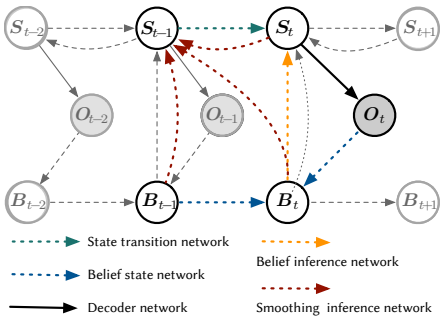


Fig. 2. **Graphical representation of the dynamic stochastic state space model. Note that arrows with different styles represent different data flows through a different neural networks**.

stochastic state space model. The dynamic stochastic state space model first produces a belief state $\mathbf{B}_t$ (blue arrows) from observation $\mathbf{O}_t$ and its previous belief state $\mathbf{B}_{t-1}$ online, using a recurrent neural network g. Then it makes a guess of the current state ($\mathbf{S}_t$) of the observation by taking its belief state $\mathbf{B}_t$ (yellow arrows) as input to infer $p_{\mathbf{B}}(\mathbf{S}_t \mid \mathbf{B}_t)$. And combing $\mathbf{S}_t$ with the current belief state $\mathbf{B}_t$ and a prior belief state $\mathbf{B}_{t-1}$, it forms a previous post guess of the state of prior observation $\mathbf{S}_{t-1}$, by computing a smoothing distribution $q(\mathbf{S}_{t-1} \mid \mathbf{S}_t, \mathbf{B}_t, \mathbf{B}_{t-1})$ and drawing a corresponding sample $\mathbf{S}_{t-1}$ (red arrows). Given the state at $t-1$, the model predicts the state at t, which reflects how the states evolve (green arrows). Meanwhile, the decoder enforces the samples of state distribution $\mathbf{S}_t$ to be able to reconstruct the current observation $\mathbf{O}_t$.

The dynamic stochastic state space models, e.g., TD-VAE [16], are hard to deal with the problem of temporal network representation learning due to the following reasons: 1) They cannot reconstruct or predict the links between nodes naturally due to the lack of a well-designed decoder for data represented in graph domain. 2) The used ELBO of them are not applicable to the variational inference on graph-structured data consisting of edges or additional auxiliary information 3) In the problem of co-embedding temporal attributed network, it is still tricky to apply the dynamic stochastic state space models to learn a latent representation space, where the topological structure of network and affinities between nodes and attributes should be preserved, even if using a feasible decoder for them, as there is not an efficient way of them to simultaneously learn the latent variables for two different categories of entities, i.e., nodes and attributes.

IV. METHODOLOGY

To address the temporal attributed network co-embedding problem, we propose a novel **D**ynamic **C**o-embedding model for **T**emporal **A**ttributed **N**etwork (DCTAN) that effectively captures the dynamics of a network by building an abstract state representing the condition of the network of current time step, and predicting the transitions between temporal abstract states of two successive time steps. An overview of our model are described in Figure 3. In the following, we illustrate our DCTAN model by first formulating the variational lower bound based on the VAE framework [41], then detailing all the neural network models for constructing a trainable data flow and finally describing the optimization process.

A. The Variational Lower Bound of DCTAN

For a temporal attributed network $\mathcal{G}_{\leq T}$, we have two types of observation sequences, i.e. the adjacency matrices $\mathbf{A}_{\leq T} = \{\mathbf{A}_0, \mathbf{A}_1, \ldots, \mathbf{A}_T\}$ and the node attributed matrices $\mathbf{X}_{\leq T} = \{\mathbf{X}_0, \mathbf{X}_1, \ldots, \mathbf{X}_T\}$ for each time step. We propose to learn the representations for both nodes and attributes, so that the affinities between nodes and attributes can be effectively captured and measured.

To extend the dynamic stochastic state space model to our dynamic co-embedding scenario, we begin with maximizing the log-likelihood of the observation sequence of $\log p(\mathcal{G}_{\leq T})$, which can be decomposed as:

$$\log(\mathcal{G}_{\leq T}) = \sum_{t=1}^{T} \log p(\mathcal{G}_t|\mathcal{G}_{<t}), \tag{4}$$

where $\log p(\mathcal{G}_t|\mathcal{G}_{<t})$ is obtained by:

$$\begin{aligned}
\log p(\mathcal{G}_t|\mathcal{G}_{<t}) &= \log \iint p\left(\mathcal{G}_t, \mathbf{S}_t, \mathbf{S}_{t-1}|\mathcal{G}_{<t}\right) d\mathbf{S}_t d\mathbf{S}_{t-1} \\
&= \log \iint p\left(\mathcal{G}_t, \mathbf{S}_t, \mathbf{S}_{t-1}|\mathcal{G}_{<t}\right) \frac{q\left(\mathbf{S}_t, \mathbf{S}_{t-1}|\mathcal{G}_{\leq t}\right)}{q\left(\mathbf{S}_t, \mathbf{S}_{t-1}|\mathcal{G}_{\leq t}\right)} d\mathbf{S}_t d\mathbf{S}_{t-1} \\
&\geq \mathop{\mathbb{E}}_{\mathbf{S}_t, \mathbf{S}_{t-1} \sim q\left(\mathbf{S}_t, \mathbf{S}_{t-1}|\mathcal{G}_{\leq t}\right)} [\log p\left(\mathcal{G}_t, \mathbf{S}_t, \mathbf{S}_{t-1}|\mathcal{G}_{<t}\right) \\
&\qquad - \log q\left(\mathbf{S}_t, \mathbf{S}_{t-1}|\mathcal{G}_{\leq t}\right)].
\end{aligned} \tag{5}$$

Here $q\left(\mathbf{S}_t, \mathbf{S}_{t-1}|\mathcal{G}_{\leq t}\right)$ is the variational distribution of the latent states, and $p\left(\mathcal{G}_t, \mathbf{S}_t, \mathbf{S}_{t-1}|\mathcal{G}_{<t}\right)$ is the the joint distribution of the latent states and the observations. The inequality of Equation (5) is obtained according to the Jensen's inequality.

According to the Markov assumption, the conditional likelihood $p(\mathcal{G}_t|\mathcal{G}_{<t})$ is naturally integrated by only the latent states of two consecutive time steps, i.e. $\mathbf{S}_t$ and $\mathbf{S}_{t-1}$, and the joint distribution $p\left(\mathcal{G}_t, \mathbf{S}_t, \mathbf{S}_{t-1}|\mathcal{G}_{<t}\right)$ can be factorized as:

$$p\left(\mathcal{G}_t, \mathbf{S}_t, \mathbf{S}_{t-1}|\mathcal{G}_{<t}\right) = p\left(\mathcal{G}_t|\mathbf{S}_t\right) p\left(\mathbf{S}_t|\mathbf{S}_{t-1}\right) p\left(\mathbf{S}_{t-1}|\mathcal{G}_{<t}\right) \tag{6}$$

Hence, Equation (5) can be rewritten as:

$$\begin{aligned}
p(\mathcal{G}_t|\mathcal{G}_{<t}) \geq & \mathop{\mathbb{E}}_{\mathbf{S}_t, \mathbf{S}_{t-1} \sim q(\chi)} [\log p\left(\mathcal{G}_t|\mathbf{S}_t\right) + \log p\left(\mathbf{S}_t|\mathbf{S}_{t-1}\right) \\
& + \log p\left(\mathbf{S}_{t-1}|\mathcal{G}_{<t}\right) - \log q\left(\mathbf{S}_t, \mathbf{S}_{t-1}|\mathcal{G}_{\leq t}\right)] \\
\stackrel{\text{def}}{=} & \mathcal{L},
\end{aligned} \tag{7}$$

where $\mathcal{L}$ is the ELBO for the log conditional likelihood $p(\mathcal{G}_t|\mathcal{G}_{<t})$. For the sake of simplicity, we use $q(\chi)$ to represent $q(\mathbf{S}_t, \mathbf{S}_{t-1}|\mathcal{G}_{\leq t})$. Since the input observations $\mathcal{G}_{\leq t} = \{\mathcal{G}_0, \ldots, \mathcal{G}_t\}$ are given sequentially, it is convenient that we model the temporal input as conditional generation of data given a context sequence. Thus, we can train two recurrent neural networks (e.g. LSTM) that extract the codes of the two types of sequence observations, i.e. $\mathbf{B}_t^{\mathcal{N}}$ and $\mathbf{B}_t^{\mathcal{A}}$, according to:

$$\mathbf{B}_t^{\mathcal{N}} = f^{\mathcal{N}}(\mathbf{B}_{t-1}^{\mathcal{N}}, \mathbf{X}_t, \mathbf{A}_t) \tag{8}$$

$$\mathbf{B}_t^{\mathcal{A}} = f^{\mathcal{A}}(\mathbf{B}_{t-1}^{\mathcal{A}}, \mathbf{X}_t^T), \tag{9}$$

where $f^{\mathcal{N}}(\cdot)$ and $f^{\mathcal{A}}(\cdot)$ are the internal state functions of two recurrent neural networks for nodes and attributes, respectively. $\mathbf{B}_t^{\mathcal{N}}$ and $\mathbf{B}_t^{\mathcal{A}}$ are called the **belief states codes** for nodes and attributes at time step t, respectively. All the information about the states of temporal attributed network can be effectively captured using these internal states of recurrent neural networks, and therefore these belief states are sufficient to compute the conditional future distribution, due to the Markov assumption underlying the state-space model. The latent states $\mathbf{S}_t$ in our DCTAN are represented as two latent factors, i.e., $\mathbf{S}_t = \{\mathbf{Z}_t^{\mathcal{N}}, \mathbf{Z}_t^{\mathcal{A}}\}$, indicating the embeddings of nodes and attributes at time step t, respectively. Then the variational distribution can be factorized, according to the mean-filed assumption, as:

$$\begin{aligned} q(\mathbf{S}_t,\mathbf{S}_{t-1}|\mathcal{G}_{\leq t}) &= q(\mathbf{Z}_t^{\mathcal{N}},\mathbf{Z}_t^{\mathcal{N}},\mathbf{Z}_{t-1}^{\mathcal{N}},\mathbf{Z}_{t-1}^{\mathcal{A}}\,|\,\mathbf{B}_t^{\mathcal{N}},\mathbf{B}_t^{\mathcal{A}},\mathbf{B}_{t-1}^{\mathcal{N}},\mathbf{B}_{t-1}^{\mathcal{A}}) \\ &= q(\mathbf{Z}_t^{\mathcal{N}},\mathbf{Z}_{t-1}^{\mathcal{N}}\,|\,\mathbf{B}_t^{\mathcal{N}},\mathbf{B}_{t-1}^{\mathcal{N}})q(\mathbf{Z}_t^{\mathcal{A}},\mathbf{Z}_{t-1}^{\mathcal{A}}|\mathbf{B}_t^{\mathcal{A}},\mathbf{B}_{t-1}^{\mathcal{A}}). \end{aligned} \tag{10}$$

To obtain the smooth state tendency of the latent embeddings of nodes and attributes over time, we further decompose these variational distributions over the two consecutive time steps by representing the smoothing posteriors over $\mathbf{Z}_t^{\mathcal{N}}$ and $\mathbf{Z}_t^{\mathcal{A}}$ as:

$$\begin{aligned} &q(\mathbf{Z}_t^{\mathcal{N}},\mathbf{Z}_{t-1}^{\mathcal{N}}|\mathbf{B}_t^{\mathcal{N}},\mathbf{B}_{t-1}^{\mathcal{N}}) \\ &= q_{B,t}^{\mathcal{N}}(\mathbf{Z}_t^{\mathcal{N}}|\mathbf{B}_t^{\mathcal{N}})q_{t-1|t}^{\mathcal{N}}(\mathbf{Z}_{t-1}^{\mathcal{N}}|\mathbf{Z}_t^{\mathcal{N}},\mathbf{B}_t^{\mathcal{N}},\mathbf{B}_{t-1}^{\mathcal{N}}), \end{aligned} \tag{11}$$

$$\begin{aligned} &q(\mathbf{Z}_t^{\mathcal{A}},\mathbf{Z}_{t-1}^{\mathcal{A}}|\mathbf{B}_t^{\mathcal{A}},\mathbf{B}_{t-1}^{\mathcal{A}}) \\ &= q_{B,t}^{\mathcal{A}}(\mathbf{Z}_t^{\mathcal{A}}|\mathbf{B}_t^{\mathcal{A}})q_{t-1|t}^{\mathcal{A}}(\mathbf{Z}_{t-1}^{\mathcal{A}}|\mathbf{Z}_t^{\mathcal{A}},\mathbf{B}_t^{\mathcal{A}},\mathbf{B}_{t-1}^{\mathcal{A}}), \end{aligned} \tag{12}$$

where we call $q_{B,t}^{\mathcal{N}}(\cdot)$ and $q_{B,t}^{\mathcal{A}}(\cdot)$ as the **belief inference networks** that infer the latent state distributions over time step t conditioned by the belief states; and $q_{t-1|t}^{\mathcal{N}}(\cdot)$ and $q_{t-1|t}^{\mathcal{A}}(\cdot)$ as the **smoothing inference networks** that infer distributions over latent states of time step $t-1$ smoothly from the latent states of time step t and the belief states over time step $t-1$ and t.

Then, substituting Equation (8), (11) and (12) into the ELBO (Equation (7)), the first term of the expectation can be rewritten as:

$$\begin{aligned} p_D\left(\mathcal{G}_t|\mathbf{S}_t\right) =& p_D\left(\mathcal{G}_t|\mathbf{Z}_t^{\mathcal{N}},\mathbf{Z}_t^{\mathcal{A}}\right) \\ =& p_D\left(\mathbf{A}_t,\mathbf{X}_t|\mathbf{Z}_t^{\mathcal{N}},\mathbf{Z}_t^{\mathcal{A}}\right) \\ =& p_D^A\left(\mathbf{A}_t|\mathbf{Z}_t^{\mathcal{N}}\right)p_D^X\left(\mathbf{X}_t|\mathbf{Z}_t^{\mathcal{N}},\mathbf{Z}_t^{\mathcal{A}}\right), \end{aligned} \tag{13}$$

which is called as **decoder network**, while the other two positive terms of Equation (7) can be rewritten as:

$$p\left(\mathbf{S}_t|\mathbf{S}_{t-1}\right) = p_T^{\mathcal{A}}(\mathbf{Z}_t^{\mathcal{N}} \mid \mathbf{Z}_{t-1}^{\mathcal{N}})p_T^{\mathcal{N}}(\mathbf{Z}_t^{\mathcal{A}} \mid \mathbf{Z}_{t-1}^{\mathcal{A}}), \tag{14}$$

$$\begin{aligned} &p\left(\mathbf{S}_{t-1}|\mathcal{G}_{t-1}\right) = \\ &p_{B,t-1}^{\mathcal{N}}\left(\mathbf{Z}_{t-1}^{\mathcal{N}}|\mathbf{B}_{t-1}^{\mathcal{N}}\right)p_{B,t-1}^{\mathcal{A}}\left(\mathbf{Z}_{t-1}^{\mathcal{A}}|\mathbf{B}_{t-1}^{\mathcal{A}})\right), \end{aligned} \tag{15}$$

where $p_T^{\mathcal{N}}(\cdot)$ and $p_T^{\mathcal{A}}(\cdot)$ are the **state transition networks** that predict the future latent state based on its previous latent state; $p_{B,t-1}^{\mathcal{N}}(\cdot)$ and $p_{B,t-1}^{\mathcal{A}}(\cdot)$ are approximated as the belief inference networks $q_{B,t-1}^{\mathcal{N}}(\cdot)$ and $q_{B,t-1}^{\mathcal{A}}(\cdot)$ respectively. Combining Equation (14) and (15) with Equation (11) and (12), the last three terms of the expectation of Equation (7) can be reformulated as a form of four Kullback-Leibler (KL) divergence terms:

$$\begin{aligned} &\mathop{\mathbb{E}}_{\mathbf{S}_t,\mathbf{S}_{t-1}\sim q(\chi)}[\log p\left(\mathbf{S}_t|\mathbf{S}_{t-1}\right)+p\left(\mathbf{S}_{t-1}|\mathcal{G}_{\leq t}\right)-\log q\left(\mathbf{S}_t,\mathbf{S}_{t-1}|\mathcal{G}_{\leq t}\right)] \\ &\quad= -D_{KL}\left(q_{B,t-1}^{\mathcal{N}}\|q_{t-1|t}^{\mathcal{N}}\right)-D_{KL}\left(q_{B,t-1}^{\mathcal{A}}\|q_{t-1|t}^{\mathcal{A}}\right) \\ &\qquad- D_{KL}\left(p_T^N\|q_{B,t}^N\right)-D_{KL}\left(p_T^A\|q_{B,t}^A\right) \\ &\quad\stackrel{\text{def}}{=}\mathcal{L}_{KL}. \end{aligned} \tag{16}$$

Hence, the total loss for our DCTAN model is:

$$\begin{aligned} \mathcal{L} =& \mathop{\mathbb{E}}_{\mathbf{Z}_t^{\mathcal{N}}\sim q_{B,t}^{\mathcal{N}}\mathbf{Z}_t^{\mathcal{A}}\sim q_{B,t}^{\mathcal{A}}}\left[\log p_D^A\left(\mathbf{A}_t|\mathbf{Z}_t^{\mathcal{N}}\right)+\log p_D^X\left(\mathbf{X}_t|\mathbf{Z}_t^{\mathcal{N}},\mathbf{Z}_t^{\mathcal{A}}\right)\right] \\ &+\mathcal{L}_{KL}. \end{aligned} \tag{17}$$

We will detail all these neural networks applied in our DCTAN model in the next subsection, with, again, the overview of DCTAN being shown in Figure 3, where different types of arrows represent the data flows of the different neural networks.

B. Neural Network Models

As it is in Figure 3, our model consists of five specific neural network models, i.e., the **belief state network**, the **belief inference network**, the **smoothing inference network**, the **state transition network** and the **decoder network**. We now present the detailed architecture of these neural networks.

The belief state network. To encode the observations $\mathbf{A}_{\leq t}$ and $\mathbf{X}_{\leq t}$ for the target temporal attributed network, we apply two Long Short-Term Memory (LSTM) recurrent neural networks to learn the belief state codes for nodes and attributes at each time steps. Specifically, the belief state codes of nodes and attributes at time step t (i.e. $\mathbf{B}_t^{\mathcal{N}}$ and $\mathbf{B}_t^{\mathcal{A}}$) are encoded by LSTM according to their corresponding input features (i.e., their adjacency matrices of nodes $\{\mathbf{A}_1,\ldots,\mathbf{A}_t\}$ and the attribute information matrices $\{\mathbf{X}_1,\ldots,\mathbf{X}_t\}$ at different time steps). For example, the attribute belief state codes $\mathbf{B}_t^{\mathcal{A}}$ can be represented as the following:

$$\mathbf{I}_t = \text{sigmoid}(\mathbf{X}_t^{\mathbf{T}}\mathbf{W}_{ii} + \mathbf{b}_{ii} + \mathbf{H}_{t-1}\mathbf{W}_{hi} + \mathbf{b}_{hi}) \tag{18}$$

$$\mathbf{F}_t = \text{sigmoid}(\mathbf{X}_t^{\mathbf{T}}\mathbf{W}_{if} + \mathbf{b}_{if} + \mathbf{H}_{t-1}\mathbf{W}_{hf} + \mathbf{b}_{hf}) \tag{19}$$

$$\mathbf{C}_t = \tanh(\mathbf{X}_t^{\mathbf{T}}\mathbf{W}_{ig} + \mathbf{b}_{ig} + \mathbf{H}_{t-1}\mathbf{W}_{hg} + \mathbf{b}_{hg}) \tag{20}$$

$$\mathbf{B}_t^{\mathcal{A}} = \text{sigmoid}(\mathbf{X}_t^{\mathbf{T}}\mathbf{W}_{io} + \mathbf{b}_{io} + \mathbf{H}_{t-1}\mathbf{W}_{ho} + \mathbf{b}_{ho}) \tag{21}$$

$$\mathbf{C}_t = \mathbf{F}_t\mathbf{C}_{t-1} + \mathbf{I}_t\mathbf{C}_t \tag{22}$$

$$\mathbf{H}_t = \mathbf{B}_t^{\mathcal{A}} * \tanh(\mathbf{C}_t) \tag{23}$$

where $\mathbf{H}_t$ is the hidden states at time step t, $\mathbf{C}_t$ is the cell states at time step t, $\mathbf{X}_t^{\mathbf{T}}$ is the input (the transpose of attribute information matrix) at time step t, $*$ is the Hadamard product, and $\mathbf{I}_t, \mathbf{F}_t, \mathbf{C}_t$ are the input, forget and cell gates, respectively. Since the belief state network is generally a recurrent neural network, $\mathbf{B}_t^{\mathcal{A}}$ is both the output belief code at current time step and the contributed factors for belief code of the next time step. The belief state for nodes $\mathbf{B}_t^{\mathcal{N}}$ can be extracted with a similar LSTM network, with the only difference that it takes the concatenation of $\mathbf{A}_t$ and $\mathbf{X}_t$ as input at time step t, i.e., $\mathbf{A}_t \oplus \mathbf{X}_t$, where $\oplus$ is the simple concatenation operation, and preprocesses the node features by multi-layer

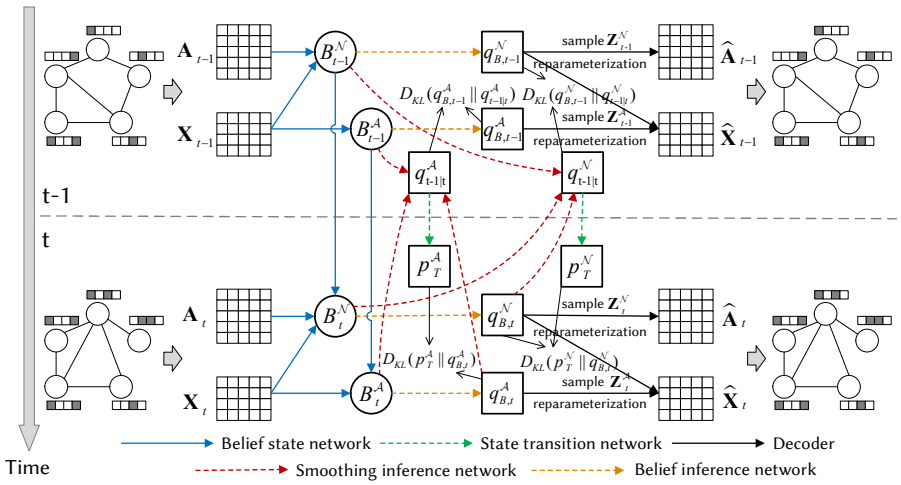


Fig. 3. The overview of our proposed DCTAN at given time steps $t-1$ and t, from the top to the bottom. The belief state network employs two Long Short-Term Memory (LSTM) recurrent neural networks with $\mathbf{B}_t^{\mathcal{N}} = f^{\mathcal{N}}(\mathbf{B}_{t-1}^{\mathcal{N}}, \mathbf{X}_t, \mathbf{A}_t)$ and $\mathbf{B}_t^{\mathcal{N}} = f^{\mathcal{A}}(\mathbf{B}_{t-1}^{\mathcal{A}}, \mathbf{X}_t^T)$, respectively. $q_B^{\mathcal{N}}$ and $q_B^{\mathcal{A}}$ are the Gaussian embeddings of nodes and attributes based on belief state codes respectively, i.e, $q_B^{\mathcal{N}}(\mathbf{Z}_t^{\mathcal{N}}|\mathbf{B}_t^{\mathcal{N}})$ and $q_B^{\mathcal{A}}(\mathbf{Z}_t^{\mathcal{A}}|\mathbf{B}_t^{\mathcal{A}})$. We use $p_{t-1|t}^{\mathcal{N}}$ and $p_{t-1|t}^{\mathcal{A}}$ for the inferred smoothing Gaussian distributions at time step $t-1$, $p_T^{\mathcal{A}}$ and $p_T^{\mathcal{N}}$ for the predicted Gaussian distributions of latent variables, i.e., $p_T^{\mathcal{N}}(\mathbf{Z}_t^{\mathcal{N}}|\mathbf{Z}_{t-1}^{\mathcal{N}})$ and $p_T^{\mathcal{A}}(\mathbf{Z}_t^{\mathcal{A}}|\mathbf{Z}_{t-1}^{\mathcal{A}})$. We sample determinist embeddings from these Gaussian distributions as inputs of decoder network to reconstruct the observations of the target network.

GCNs (graph convolutional networks) to preserve the high-order neighborhood information. In our experiments, we only use 2-layer GCNs, as adding more layers leads to small performance improvement as mentioned in related literature [4, 15]. For brevity, we omit the full network formulation for $\mathbf{B}_t^{\mathcal{N}}$.

The belief inference network. The purpose of belief inference network is to infer high-quality and informative embeddings of nodes and attributes at each time step based on the above extracted belief state codes. The learned latent state representations are represented by Gaussian distributions with their means being the embedding of nodes and attributes and their variances measuring the uncertainly of their embeddings. To infer the Gaussian embeddings of nodes and attributes at a given time step t from the corresponding belief state codes, we employ a two layer fully connected neural networks composed of non-linear mapping functions to map the codes (i.e. $\mathbf{B}_t^{\mathcal{A}}$ and $\mathbf{B}_t^{\mathcal{N}}$) to the means and variations of two Gaussian distributions, respectively. Specifically, the means and variances of Gaussian embeddings for nodes and attributes are inferred according to the corresponding belief state codes by:

$$\begin{aligned}\left[\boldsymbol{\mu}_t^{\mathcal{N}}, (\boldsymbol{\sigma}_t^{\mathcal{N}})^2\right] &= q_{B,t}^{\mathcal{N}}(\mathbf{Z}_t^{\mathcal{N}}|\mathbf{B}_t^{\mathcal{N}}),\\ \left[\boldsymbol{\mu}_t^{\mathcal{A}}, (\boldsymbol{\sigma}_t^{\mathcal{A}})^2\right] &= q_{B,t}^{\mathcal{A}}(\mathbf{Z}_t^{\mathcal{A}}|\mathbf{B}_t^{\mathcal{A}}),\end{aligned} \tag{24}$$

where $\mu_t^{\mathcal{N}}$, $\mu_t^{\mathcal{A}}$, $(\boldsymbol{\sigma}_t^{\mathcal{N}})^2$ and $(\boldsymbol{\sigma}_t^{\mathcal{A}})^2$ are the means and variances of embeddings for nodes and attributes respectively; $q_{B,t}^{\mathcal{N}}(\cdot)$ and $q_{B,t}^{\mathcal{N}}(\cdot)$ are two two-layer-fully-connected networks that have the same structure but different parameters from each other, the structure of which is defined as:

$$\begin{aligned}\mathbf{H}^{(1)} &= \tanh\left(\mathbf{B}\mathbf{W}^{(1)} + \mathbf{b}^{(1)}\right)\\ \mathbf{H}^{(2)} &= \text{sigmoid}\left(\mathbf{B}\mathbf{W}^{(2)} + \mathbf{b}^{(2)}\right)\\ \left[\boldsymbol{\mu}, \boldsymbol{\sigma}^2\right] &= \left(\mathbf{H}^{(1)} \odot \mathbf{H}^{(2)}\right)\mathbf{W}^{(3)} + \mathbf{b}^{(3)}.\end{aligned} \tag{25}$$

Here $\odot$ is the element-wise product, and $\mathbf{W}^{(1)}$, $\mathbf{W}^{(2)}$, $\mathbf{W}^{(3)}$, $\mathbf{b}^{(1)}$, $\mathbf{b}^{(2)}$ & $\mathbf{b}^{(3)}$ are trainable parameters.

The smoothing inference network. The smoothing inference network aims at enhancing our model by smoothly inferring more informative embeddings considering the future belief states of network. Given two successive training time steps $t-1$ and t, the smoothing inference network learns a smoothing function such that the embeddings at time step $t-1$ can be inferred from the embeddings at time step t, which make sure that the embeddings at time step $t-1$ are also informative about the state of the world at time t. Since the embeddings of nodes and attributes inferred by the belief inference networks are Gaussian distributions, we use the smoothing inference network to infer two similar Gaussian distributions of time step $t-1$ according to the embedding of nodes and attributes at t (i.e. $\mathbf{Z}_t^{\mathcal{N}}$ and $\mathbf{Z}_t^{\mathcal{A}}$) and the belief states of time steps $t-1$ and t (i.e., $\mathbf{B}_{t-1}^{\mathcal{N}}$, $\mathbf{B}_{t-1}^{\mathcal{A}}$, $\mathbf{B}_t^{\mathcal{N}}$ and $\mathbf{B}_t^{\mathcal{A}}$):

$$\begin{aligned}\left[\boldsymbol{\mu}_{t-1}^{\mathcal{N}}, (\boldsymbol{\sigma}_{t-1}^{\mathcal{N}})^2\right] &= q_{t-1|t}^{\mathcal{N}}(\mathbf{Z}_{t-1}^{\mathcal{N}}|\mathbf{Z}_t^{\mathcal{N}}, \mathbf{B}_{t-1}^{\mathcal{N}}, \mathbf{B}_t^{\mathcal{N}}),\\ \left[\boldsymbol{\mu}_{t-1}^{\mathcal{A}}, (\boldsymbol{\sigma}_{t-1}^{\mathcal{A}})^2\right] &= q_{t-1|t}^{\mathcal{A}}(\mathbf{Z}_{t-1}^{\mathcal{A}}|\mathbf{Z}_t^{\mathcal{A}}, \mathbf{B}_{t-1}^{\mathcal{A}}, \mathbf{B}_t^{\mathcal{A}}).\end{aligned} \tag{26}$$

Here $\mathbf{Z}_t^{\mathcal{N}}$ and $\mathbf{Z}_t^{\mathcal{A}}$ are the deterministic embeddings samples of nodes and attributes at time step t from the corresponding Gaussian embeddings by using the reparameterization trick [41]; $\boldsymbol{\mu}_{t-1}^{\mathcal{N}}, \boldsymbol{\mu}_{t-1}^{\mathcal{A}}, (\boldsymbol{\sigma}_{t-1}^{\mathcal{N}})^2$ and $(\boldsymbol{\sigma}_{t-1}^{\mathcal{A}})^2$ are the means and variances of the smoothing Gaussian distributions over latent embeddings to be infered at time step $t-1$. $q_{t-1|t}^{\mathcal{N}}(\cdot)$ and $q_{t-1|t}^{\mathcal{N}}(\cdot)$ are two two-layer-fully-connected networks that have the same structure but different parameters from each other,

the structure of which is defined as:

$$\begin{aligned}\mathbf{H}^{(1)} &= \tanh\left((\mathbf{Z}_t \oplus \mathbf{B}_{t-1} \oplus \mathbf{B}_t)\mathbf{W}^{(1)} + \mathbf{b}^{(1)}\right),\\ \mathbf{H}^{(2)} &= \text{sigmoid}\left((\mathbf{Z}_t \oplus \mathbf{B}_{t-1} \oplus \mathbf{B}_t)\mathbf{W}^{(2)} + \mathbf{b}^{(2)}\right),\\ \left[\boldsymbol{\mu}_{t-1}, \boldsymbol{\sigma}_{t-1}^2\right] &= \left(\mathbf{H}^{(1)} \odot \mathbf{H}^{(2)}\right)\mathbf{W}^{(3)} + \mathbf{b}^{(3)}. \end{aligned} \tag{27}$$

After inferring the above smoothing distributions at time step $t-1$, we are able to calculate the KL-divergence terms $D_{KL}\left(q^{\mathcal{N}}_{B,t-1} \| q^{\mathcal{N}}_{t-1|t}\right)$ and $D_{KL}\left(q^{\mathcal{A}}_{B,t-1} \| q^{\mathcal{A}}_{t-1|t}\right)$ in Equation (17). The representation quality therefore can be enhanced based on the smoothing inference and minimizing the KL-divergence between the inferred distributions and the smoothing distributions.

The state transition network. An important purpose of our proposed DCTAN is to track the state transition tendency of the latent state representations of temporal attributed network over time. We achieve this by training two state transition networks that use the embeddings of current time step to predict the embeddings of the next time step. Similar to the smoothing inference networks, we predict the means and variances of the embedding and minimizing the KL-divergence between these predicted embeddings and the inferred embedding from the belief inference networks. To capture the transition tendency between two state representations at successive time steps $t-1$ and t, we train two two-layer-fully-connected networks which predict the states of network at the next time step based on those at current time step:

$$\begin{aligned}\left[\boldsymbol{\mu}_t^{\mathcal{N}}, (\boldsymbol{\sigma}_t^{\mathcal{N}})^2\right] &= p_T^{\mathcal{N}}(\mathbf{Z}_t^{\mathcal{N}} | \mathbf{Z}_{t-1}^{\mathcal{N}}),\\ \left[\boldsymbol{\mu}_t^{\mathcal{A}}, (\boldsymbol{\sigma}_t^{\mathcal{A}})^2\right] &= p_T^{\mathcal{A}}(\mathbf{Z}_t^{\mathcal{A}} | \mathbf{Z}_{t-1}^{\mathcal{A}}),\end{aligned} \tag{28}$$

where $\mathbf{Z}_{t-1}^{\mathcal{N}}$ and $\mathbf{Z}_{t-1}^{\mathcal{A}}$ are the sampled determinist embeddings of nodes and attributes from the corresponding inferred Gaussian embeddings of them at time step $t-1$, respectively; $\boldsymbol{\mu}_t^{\mathcal{N}}$, $\boldsymbol{\mu}_t^{\mathcal{A}}$, $(\boldsymbol{\sigma}_t^{\mathcal{N}})^2$ and $(\boldsymbol{\sigma}_t^{\mathcal{A}})^2$ are the means and variances of the predicted Gaussian embeddings at time step t for nodes and attributes, respectively; $p_T^{\mathcal{N}}(\cdot)$ and $p_T^{\mathcal{A}}(\cdot)$ are two two-layer-fully-connected networks defined as:

$$\begin{aligned}\mathbf{H}^{(1)} &= \tanh\left(\mathbf{Z}_{t-1}\mathbf{W}^{(1)} + \mathbf{b}^{(1)}\right),\\ \mathbf{H}^{(2)} &= \text{sigmoid}\left(\mathbf{Z}_{t-1}\mathbf{W}^{(2)} + \mathbf{b}^{(2)}\right),\\ \left[\boldsymbol{\mu}_t, \boldsymbol{\sigma}_t^2\right] &= \left(\mathbf{H}^{(1)} \odot \mathbf{H}^{(2)}\right)\mathbf{W}^{(3)} + \mathbf{b}^{(3)}. \end{aligned} \tag{29}$$

Having obtained the predicted Gaussian embeddings, we are able to calculate the KL-divergence terms $-D_{KL}\left(p_T^N \| q_{B,t}^N\right)$ and $-D_{KL}\left(p_T^A \| q_{B,t}^A\right)$ in loss function Equation 16. Maximizing the above two KL-divergence terms during training forces the predicted Gaussian distributions to be closer to the belief based Gaussian embeddings, the state transition tendency thereby can be learned and captured.

The decoder network. The decoder aims to reconstruct the target attributed network based on the inferred embeddings of nodes and attributes. Since the learned embeddings of each node and attribute are represented by Gaussian distributions, we first sample determinist embeddings from such distributions by using the reparameterization trick [41]. To efficiently reconstruct the observations of the target network at a given time step t, i.e., the adjacency matrix of nodes $\mathbf{A}_t{}'$ and attribute information matrix $\mathbf{X}_t{}'$, we employ two inner-product decoders with nonlinear activation function to predict the probabilities of each link and node-attribute association based on the sampled embeddings at time step t, i.e., $\mathbf{Z}_t^{\mathcal{N}}$ and $\mathbf{Z}_t^{\mathcal{A}}$:

$$\begin{aligned}\mathbf{A}_t{}' &= p_D^A\left(\mathbf{A}_t | \mathbf{Z}_t^{\mathcal{N}}, (\mathbf{Z}_t^{\mathcal{N}})^{\mathbf{T}}\right),\\ \mathbf{X}_t{}' &= p_D^X\left(\mathbf{X}_t | \mathbf{Z}_t^{\mathcal{A}}, \mathbf{Z}_t^{\mathcal{N}}\right),\end{aligned} \tag{30}$$

where $(\mathbf{X}_t{}')_{ij} \in \mathbf{X}_t{}'$ and $(\mathbf{A}_t{}')_{ij} \in \mathbf{A}_t{}'$ are the predicted probabilities of the i-th node having the j-th attribute and an edge existing between the i-th node and the j-th node at time step t, respectively; $p_D^A(\cdot)$ and $p_D^X(\cdot)$ are two decoders that perform the similar inner-product operations with nonlinear function but have different inputs. The inner-product decoder $p_D^A(\cdot)$, which decodes the adjacency probability matrix of nodes by predicting all links from the corresponding sampled node embeddings, is defined as:

$$p_D^A\left(\mathbf{Z}_t^{\mathcal{N}}, (\mathbf{Z}_t^{\mathcal{N}})^{\mathbf{T}}\right) = \text{sigmoid}\left(\mathbf{Z}_t^{\mathcal{N}}(\mathbf{Z}_t^{\mathcal{N}})^{\mathbf{T}}\right), \tag{31}$$

where $\mathbf{Z}_t^{\mathcal{N}}$ is the embedding matrix for all nodes at time step t. We use a similar decoder $p_D^X(\cdot)$ to reconstruct the node attribute matrix by predicting all node-attribute associations from the determinist embeddings of attributes and nodes (i.e., $\mathbf{Z}_t^{\mathcal{A}}$ and $\mathbf{Z}_t^{\mathcal{N}}$), the inner-product operation of which is defined as:

$$p_D^X\left(\mathbf{Z}_t^{\mathcal{A}}, \mathbf{Z}_t^{\mathcal{N}}\right) = \text{sigmoid}\left(\mathbf{Z}_t^{\mathcal{N}}(\mathbf{Z}_t^{\mathcal{A}})^{\mathbf{T}}\right), \tag{32}$$

To train the above probabilistic decoders for reconstructing the correct observations of the target network, we can maximize the two expectation terms in Equation 17, which are regarded as expected negative reconstruction error loss.

C. Optimization

Since all priors of latent states distributions and the variational posteriors are assumed to be Gaussian distributions, the KL-divergence terms in Equation (16) have analytical forms. However, optimizing the expectation terms of Equation (17) are intractable commonly. We address this issue by using the Stochastic Gradient Variational Bayes (SGVB) estimator and the reparameterization trick [41], so that we can directly optimize ELBO by sampling deterministic and differentiable embedding samples from the inferred variational distributions:

$$\begin{aligned}-\mathcal{L} =& \frac{1}{L}\sum_{l=1}^{L}\left(\log p_D^A(\mathbf{A}_t | \mathbf{Z}_t^{\mathcal{N}^{(l)}})\right)\\ &+ \frac{1}{L}\sum_{l=1}^{L}\left(\log p_D^X(\mathbf{X}_t | \mathbf{Z}_t^{\mathcal{A}^{(l)}}, \mathbf{Z}_t^{\mathcal{N}^{(l)}})\right) + \mathcal{L}_{KL},\end{aligned} \tag{33}$$

where

$$\mathbf{Z}_t^{\mathcal{N}(l)} = (\boldsymbol{\sigma}_t^{\mathcal{N}})^{(l)} \odot \epsilon + (\boldsymbol{\mu}_t^{\mathcal{N}})^{(l)}, \tag{34}$$

$$\mathbf{Z}_t^{\mathcal{A}(l)} = (\boldsymbol{\sigma}_t^{\mathcal{A}})^{(l)} \odot \epsilon + (\boldsymbol{\mu}_t^{\mathcal{A}})^{(l)}, \tag{35}$$

$\epsilon \sim \mathcal{N}(0, \mathrm{I})$ is a auxiliary variable, and L is the size of sampling. To optimize Equation (17), we back-propagate total loss through the belief state network, inference network, state prediction network and decoder network whose trainable parameters are updated by the gradient descent.

Here we provide the time complexity analysis for our proposed model DCTAN. For per epoch of the training procedures, the total time complexity of our model is $O(T|\tilde{\mathbf{A}}^+|(D+B+H_{BI}+H_{ST}+H_{SI}))$, where T is the total time steps; $|\tilde{\mathbf{A}}^+|$ indicates the number of nonzero entires in the Laplacian matrix, i.e., the symmetric normalized adjacent matrix $\tilde{\mathbf{A}}$, which is generated by ignoring the timestamp of the temporal edges of a dynamic network; D is the dimension of the latent embeddings; B denotes the dimension of the belief state codes; H_{BI}, H_{ST}, and H_{SI} denote the dimension of the hidden layer of the belief inference networks, the state transition networks, and the smoothing inference networks, respectively.

D. Discussion between DCTAN and TD-VAE

To some extent, DCTAN is relatively linked to TD-VAE, as both depends on the dynamic stochastic state space model to track the evolving tendency in a dynamic environment. However, there are a number of significant improvements of DCTAN over TD-VAE when tackling the graph embedding problem in dynamic settings: Compared with TD-VAE [16] that does not consider the structure information of a graph, DCTAN is proposed to additionally take into account the evolving topology of a dynamic network via employing Graph Neural Network encoders, e.g., Graph Convolutional Network [4]. In the light of prolific auxiliary node attributes in many real-world scenarios, DCTAN ulteriorly extends the dynamic stochastic state space model in a co-embedding way which is efficient in capturing the affinities between two categories of entities, e.g., user-word correlation in user profiling [10]. Additionally, in the experiments we take plain TD-VAE as a baseline and find that DCTAN outperforms it in most cases, which validates that the superior of our model over TD-VAE.

V. EXPERIMENTAL SETUP

In what follows, we first introduce the research questions that we aim to answer in this paper (Section V-A). Then we describe the datasets (Section V-B). Finally, we describe our baselines and settings (Section V-C).

A. Research Questions

We evaluate our proposed DCTAN model in this paper by answering the following research questions: (**RQ** 1) How does our DCTAN perform in static graph mining tasks (e.g. link reconstruction) by utilizing the learned representations compared with other network embedding methods? (**RQ** 2) Can the way of capturing temporal information of a network benefit the performance of our method on graph mining tasks (e.g. link prediction)? Is the dynamically modelling way used in our DCTAN better than other dynamic modelling methods? (**RQ** 3) How does DCTAN perform in the task of attribute inference, where capturing the affinities between nodes and attributes is crucial? (**RQ** 4) Can our DCTAN capture the affinities between nodes and attributes for temporal attributed networks to predict the future attributes of nodes? How does our DCTAN perform by utilizing the learned representations in the task of predicting the future attributes of nodes? (**RQ** 5) How do the learned embeddings of nodes and attributes of a temporal network change over time and can we intuitively evaluate this evolution? (**RQ** 6) Could the proposed DCTAN be evaluated through detailed ablation study to help better understand it?

B. Datasets

We conduct experiments on the following datasets: three plain temporal networks, three temporal attributed networks and one synthetic network from static network, the statistics of which is provided in Table II:

- **Email-d, Email** [42]: These two networks are generated using email data from a large European research institution, where members and e-mails with time labels are nodes and temporal edges, respectively. Email contains all the communication information about all the departments at the institution, and Email-D is a sub-network dataset of Email corresponding to the communication of only one of these departments.
- **College** [43]: It is an online social network at the University of California based on private sent messages, where nodes are staff and temporal edges are messages with time labels indicating the message delivery time.
- **School** [43]: This is a social network of a primary school, where nodes represent children or teachers, and edges correspond to contacts between the children and teachers in that school. Attributes of each node are the class numbers of children and teachers.
- **Cora** [44]: The Cora dataset is static citation networks, where nodes are publications and edges represents citation links. Attributes of each node are bag-of-words representations of the corresponding publications.
- **DBLP-4000, DBLP-10000**: These two datasets are crawled from the DBLP public bibliography data [1]. We construct the dynamic coauthor networks extracted from publications in the top 172 computer science conferences, where the nodes represent authors, the edges are publications with their time steps being the publishing year, and the attributes are the keyterms extracted from the paper titles. The scales of the two network datasets are different.

In order to generate snapshots of temporal network for different time steps, for each of the above mentioned datasets, we first evenly split its total time span into a pre-fixed number of time steps, and determine which time step each temporal edge belongs to according to its original timestamp. As shown in Table II, we generate 14 snapshots for Email, Email-d, DBLP-4000 and DBLP-10000, and 20 snapshots for School and College. Cora is a static attributed network and we will detail how to generate its dynamic version for

[1]Available from: http://dblp.uni-trier.de/xml/.

TABLE II
STATISTICS OF DATASETS

Datasets	#Nodes	#Edges	#time steps	#Attributes
School	238	13,941	20	11
College	1,899	59,835	20	-
Email-D	319	61,046	14	-
Email	986	332,334	14	-
Cora	2,708	5,429	-	1,433
DBLP-4000	4,000	11,2452	14	2000
DBLP-10000	10,000	17,3124	14	5000

attribute inference/prediction tasks in Section VI-C. We use the three real-world plain networks, i.e., Email [45], Email-d [45], and College [46], as they have been widely used in former works [45, 46]. We find that there are scarce publicly available dynamic attributed networks upto now, and therefore we construct two dynamic attributed networks in different scales (called DBLP-4000 and DBLP-10000, respectively) from DBLP. Moreover, we create a synthetic dynamic graph based on a real-world static attributed network, Cora, which has been used in previous work [15].

C. Baselines and Settings

We compare our DCTAN model with the following baseline network embedding methods (both the static and dynamic ones):

- **DeepWalk** [1]: This network embedding method samples node sequences by random walks on network, and then inputs these sequences as sentences to the Skip-gram model to learn representations for nodes of network. We set $\gamma = 80$ and $w = 10$.
- **Node2vec** [2]: The Node2vec model is extended from DeepWalk by employing a better random walk method to capture the neighbourhood information of nodes, which can strike a balance between local and global properties of a network with biased parameters p and q. In our experiments, we set the hyper-parameters as $\gamma = 80$, $w = 10$, $p = 1$ and $q = 0.25$.
- **TNE** [11]: TNE is a dynamic network embedding method that learns embeddings for nodes in all the time steps based on matrix factorization. We set the hyper-parameters as $\lambda = 0.01$, $\zeta = \sqrt{node\_num}$ and $\delta = \frac{2\zeta}{emb\_dim}$, where $node\_num$ and emb_dim denote the number of nodes and the dimension of node embeddings, respectively.
- **CTDNE** [12]: CTDNE is a continuous-time dynamic network embedding method for learning latent graph representations, which considers the temporal information of network structure evolution. We set the hyper-parameters as $\gamma = 80$, $w = 10$, $p = 1$ and $q = 0.25$.
- **CAN** [15]: CAN is a extended variational auto-encoder model, which co-embeds the attributes and nodes for static attributed networks.
- **GraphSage** [47]: It is a general inductive framework that leverages node feature information (e.g., text attributes) to efficiently generate node embeddings for previously unseen data. It learns a function that generates embeddings by sampling and aggregating features from a node's local neighborhood. We set its parameters as $K = 2$, $S_1 = 25$, and $S_2 = 10$ in our experiments.
- **TADW** [48]: This method incorporates text features of vertices into network representation learning under the framework of matrix factorization. We set the parameters of this model as $k = 80$ and $\lambda = 0.2$.

In addition, we contrive the following variants of our model DCTAN for ablation studies:

- **DCTAN-N** [16]: This model is a simplified version of original DCTAN that learns latent state representations for nodes only, which applies TD-VAE model [16] by replacing a feasible decoder that could reconstruct node-to-node links. In other words, this variant only preserves the structural information.
- **DCTAN-A** [16]: This method is a simplified variant of DCTAN by applying TD-VAE [16] to reconstruct the node-to-attribute associations only but neglect the structural information, i.e., the links among nodes.
- **DCTAN-NTI**. This variant removes the smoothing inference networks and the state transition networks. Its objective function is the first two terms of Equation. 17, i.e., the reconstruction loss.
- **DCTAN-NT**. This variant removes the smoothing inference networks but retains the state transition networks. Its objective function consists of the first two terms of Equation. 17 and the last two terms of Equation. 16.
- **DCTAN-NI**. This variant removes the state transition networks but retains the smoothing inference networks. Its objective loss contains the first two terms of Equation. 17 and the first two terms of Equation. 16.

For those static network embedding baselines (i.e. DeepWalk, Node2vec, GraphSage, and TADW) that cannot directly embed temporal networks, we ignore the timestamps of temporal edges to generate static networks, i.e., all the temporal links are regarded as static links. To investigate the importance of different components of our DCTAN, we conduct detailed ablation studies, where five variants of DCTAN, i.e., DCTAN-A,DCTAN-N, DCTAN-NTI, DCTAN-NI, and DCTAN-NT, are built by removing some components of DCTAN as baselines for comparisons; see the above descriptions.

All the baselines are implemented by the codes published by the authors, and the parameters in all these baselines are tuned to be optimal for best performance. For our DCTAN, we train it for 3000 iterations by Adam [49], with the learning rate being set to 0.0005, which follows that in TD-VAE [16] model. For our neural networks, we use 64-dimensional latent variables in all the experiments. For fair comparisons, the dimension of embedding for all methods is fixed to be 64. We implement DCTAN [2], DCTAN-N, DCTAN-A, DCTAN-NTI, DCTAN-NI, and DCTAN-NT by Pytorch [50].

VI. RESULTS AND ANALYSIS

In the following, we report and analyse our experimental results, where we answer the research questions (**RQ1**-**RQ5**).

[2]The code of our DCTAN model is published at https://github.com/tnnls2020/DCTAN

TABLE III
RESULTS OF LINK RECONSTRUCTION. THE BEST PERFORMANCE OF EACH METRIC IS IN **BOLD**. WE USE $*$ TO DENOTE A SIGNIFICANT DIFFERENCE BETWEEN OUR BEST MODEL AND THE BEST BASELINES, ACCORDING TO THE PAIRED T-TEST FOR $p < 0.05$.

Methods	Email		Email-D		College		School		DBLP-4000		DBLP-10000	
	AUC	AP	AUC	AP	AUC	AP	AUC	AP	AUC	AP	AUC	AP
DeepWalk	.887	.872	.935	.867	.820	.750	.915	.907	.905	.898	.901	.889
Node2Vec	.904	.888	.967	.962	.900	.894	.922	.919	.913	.898	.915	.898
CAN	N.A	N.A	N.A	N.A	N.A	N.A	.719	.725	.921	.922	.922	.924
TNE	.911	.861	.849	.863	.813	.803	.639	.706	.925	.924	.925	.928
CTDNE	.775	.774	.937	.891	.925	.873	.944	.900	.928	.929	.921	.924
GraphSage	N.A	N.A	N.A	N.A	N.A	N.A	.921	.924	.911	.908	.909	.905
TADW	N.A	N.A	N.A	N.A	N.A	N.A	.944	.949	.938	.944	.930	.937
DCTAN-N	.955$*$	.954$*$	.986$*$	.985$*$	.944$*$	.953$*$	.955	.957	.946	.957	.945	.953
DCTAN	N.A.	N.A.	N.A.	N.A.	N.A.	N.A.	**.979**$*$	**.985**$*$	**.958**$*$	**.969**$*$	**.955**$*$	**.965**$*$

TABLE IV
RESULTS OF LINK PREDICTION. THE BEST PERFORMANCE OF EACH METRIC IS IN **BOLD**. WE USE $*$ TO DENOTE A SIGNIFICANT DIFFERENCE BETWEEN OUR BEST MODEL AND THE BEST BASELINES, ACCORDING TO THE PAIRED T-TEST FOR $p < 0.05$.

Methods	Email		Email-D		College		School		DBLP-4000		DBLP-10000	
	AUC	AP	AUC	AP	AUC	AP	AUC	AP	AUC	AP	AUC	AP
DeepWalk	.875	.849	.937	.898	.687	.707	.879	.838	.890	.884	.891	.879
Node2Vec	.884	.851	.950	.918	.709	.718	.926	.903	.904	.886	.905	.888
CAN	N.A	N.A	N.A	N.A	N.A	N.A	.689	.701	.915	.918	.921	.922
TNE	.923	.899	.849	.884	.719	.738	.609	.695	.920	.913	.925	.928
CTDNE	.732	.719	.951	.910	.816	.755	.923	.901	.923	.925	.917	.915
GraphSage	N.A	N.A	N.A	N.A	N.A	N.A	.907	.905	.903	.900	.897	.896
TADW	N.A	N.A	N.A	N.A	N.A	N.A	.937	.943	.934	.937	.927	.932
DCTAN-N	.944$*$	.945$*$	.990$*$	.989$*$	.964$*$	.958$*$	.959	.951	.942	.951	.941	.951
DCTAN	N.A.	N.A.	N.A.	N.A.	N.A.	N.A.	**.970**$*$	**.975**$*$	**.952**$*$	**.964**$*$	**.943**$*$	**.958**$*$

A. Link Reconstruction

We first answer research question **RQ1** by evaluating our DCTAN model on the link reconstruction task, which aims at determining if there is an edge between two nodes at a given time step t based on the learned embeddings of nodes at that time step. Following the works in [15, 25], to evaluate the performance of our DCTAN, for each snapshot, we randomly divide all edges of the target temporal network into three sets, i.e., the training set (85%), the validating set (5%) and the testing set (10%). We also randomly sample an equal number of non-existing links, which are referred to the negative instances (two nodes that do not share a link). As shown in Equation (33), our DCTAN model learns a loss function reconstructing both the positive edges and negative edges at each time step during training, and hence we directly use our inner-product decoder network to predict the probability of the existence of a link based on the learned embeddings of nodes. For other baseline methods, we rank both positive and negative links, according to the L2-norm/euclidean distance of the learned embedding vectors between two nodes at the time step of testing link, e.g., the euclidean distance between the i-th node and the j-th node at time step t, i.e., $(\mathbf{Z}_t^{\mathcal{N}})_i \cdot (\mathbf{Z}_t^{\mathcal{N}})_j^{\mathbf{T}}$.

Based on the result showing in Table III, we can obtain the following observations: (1) In temporal plain networks (i.e. the Email-D, Email and College networks), DCTAN-N performs better than any of the static and dynamic baseline methods. This result indicates that DCTAN-N can effectively take advantage of the temporal information of network to improve the quality of the learned representations. (2) Our DCTAN performs the best in the temporal attributed network School, DBLP-4000, and DBLP-10000, which demonstrates that our DCTAN model is able to learn high-quality representations of nodes by utilizing both the structure and attribute information of non-plain networks in the dynamic environment. DCTAN performs better than dynamic baselines, e.g., DCTAN-N, TNE and CTDNE, because DCTAN further incorporates the node attributes to improve the latent embeddings of nodes. Compared to the static co-embedding model CAN, our DCTAN takes the evolution of temporal attributed network (i.e., the change of links and node attributes) into account by dynamically co-embedding nodes and attributes for inferring better network representations in the dynamic environment.

B. Link Prediction

We now turn to answer research question **RQ2** by evaluating our DCTAN model on link prediction task. This task is similar to the link reconstruction task, with the only difference that we aim to predict the existence of an edge at a given time step $t+1$ based on the embedding vectors of two nodes at time step t. Like the link reconstruction task in Section VI-A, we randomly divide the edges into three sets (the testing set, the training set and the validating set) at each time step and sample an equal number of non-existing links as the negative instances, then we predict the probability of the existence of the edges in testing set based on the learned embeddings of them. Table IV reports the AUC and AP scores for each method. As we can see, our DCTAN achieves

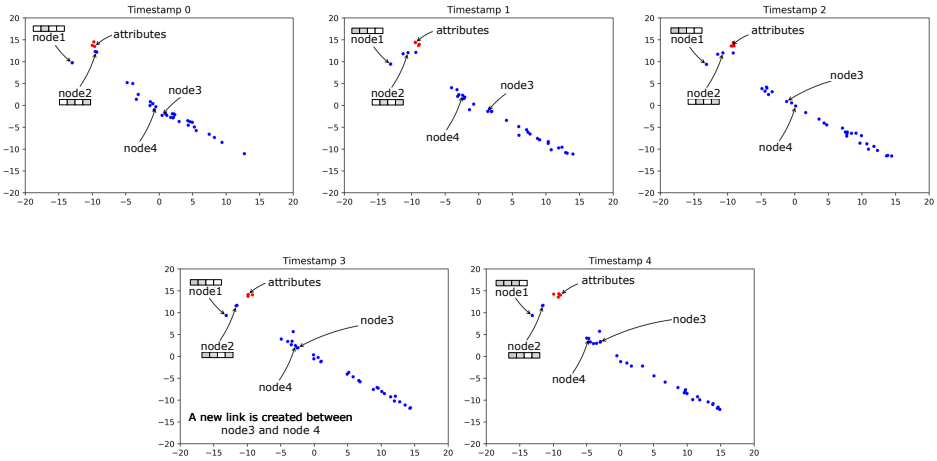


Fig. 4. 2-dimensional visualization of Gaussian embeddings of nodes and attributes at 5 time steps for an example temporal attributed network, the structure of which is shown in Figure 1. The red nodes are the attributes and the blue nodes represent the nodes in the toy network. The small rectangles around node represent the attributes of node.

better performance than any of the static or dynamic baseline models. This result demonstrates that the way of capturing the temporal information of the target network used in our our proposed DCTAN benefits the link prediction performance and is better than other dynamic methods. Moreover, we find that DCTAN achieves better performance than DCTAN-N on temporal attributed network School, DBLP-4000, and DBLP-10000. This is mainly because that DCTAN improves the quality of the learned embeddings of nodes by utilizing not only topological network structure but also the associated node attribute in dynamic environment. It is found that DCTAN outperforms the static co-embedding model CAN in the task of link prediction, and this could be explained by that our model dynamically learns the node representations via leveraging the temporal information caused by the evolving of dynamic attributed networks over time.

In addition, we notice that compared with the substantial improvement of our model over the second-best baseline DCTAN-N in terms of link reconstruction on other datasets, the improvement on link prediction task on DBLP-10000 is relatively less conspicuous. We speculate that when performing link prediction, the manually-operated node attributes which are extracted from the keyterms of paper titles are not informative enough to facilitate the inference of the future links. Moreover, another possible reason is that the scales of nodes and attributes of DBLP-10000 are both 2.5 times larger than DBLP-4000, which makes precisely predicting the future more challenging.

C. Attribute Inference

Subsequently, we answer research question **RQ3** by analysing the performance result of our model in the task of attribute inference. The attribute inference task aims to predict the probability of a node having an attribute at a given time step T based on the learned embeddings of them at the current time step T. As most of temporal network embedding methods, e.g., TNE [5], CTDNE [12], DynamicTriad [6] and HTNE [7], learn embeddings for nodes only, which cannot obtain the affinities between nodes and attributes, and existing traditional attribute inference algorithms, e.g., SAN [51], EdgeExp [52], and BLA [53] performs worse than the static co-embedding model CAN (the experimental results in work [15]), therefore we conduct experiments for attribute inference task with only our DCTAN model and CAN to verify if our model is able to capture the dynamic affinities between nodes and attributes in the attribute-oriented task by utilizing both structure and attribute information. For evaluation purpose, we employ the AUC and AP metrics to measure the performance of DCTAN. Table V reports the experimental results on the School network, a synthetic network from the static network Cora, where we randomly reduce 10% node-attribute associations and edges for 9 times to generate the dynamic data for 10 time steps, and two dynamic academic networks DBLP-4000 and DBLP-10000. From the performance results, We could find that our DCTAN is able to achieve better performance on Cora, School, DBLP-4000, and DBLP-10000 datasets in terms of evaluation metrics AUC and AP. This result can be explained by the reason that our DCTAN optimizes a loss function consisting of the reconstruction error of all the node-attribute associations at different time steps, and our model utilizes temporal information of attribute change and dynamic link by building the belief states for all attributes and nodes at each time step.

In our experiment, we validate the fact that attribute inference can help to improve link prediction; that is, link prediction accuracy is further improved by first inferring missing attributes [15, 51] even in dynamic environment. Moreover, our model could trade the weights between node-to-node links and node-to-attribute associations without any free parameters such that our model can lead to notable improvement of the performance in both node-oriented network problems (e.g., link reconstruction) and attribute inference problem (e.g., predicting the attributes of nodes).

TABLE V
RESULTS OF ATTRIBUTE INFERENCE. THE BEST PERFORMANCE OF EACH METRIC IS IN **BOLD**.

Methods	Cora		School		DBLP-4000		DBLP-10000	
	AUC	AP	AUC	AP	AUC	AP	AUC	AP
CAN	.932	.916	.976	.968	.952	.946	.958	.954
DCTAN	**.942**	**.933**	**.986**	**.986**	**.976**	**.968**	**.982**	**.975**

TABLE VI
RESULTS OF ATTRIBUTE PREDICTION. THE BEST PERFORMANCE OF EACH METRIC IS IN **BOLD**.

Methods	Cora		School		DBLP-4000		DBLP-10000	
	AUC	AP	AUC	AP	AUC	AP	AUC	AP
CAN	.927	.911	.963	.959	.929	.927	.923	.912
DCTAN	**.943**	**.938**	**.975**	**.967**	**.962**	**.960**	**.957**	**.953**

D. Attribute Prediction

Next, we turn to research question **RQ4** by performing the task of attribute prediction, which is similar to attribute inference but aims at predicting the attributes for nodes at time step $T+1$ based on the learned latent embeddings of nodes and attributes at the previous time step T. Following the attributed inference task, the node-attribute associations are divided into the training set (85%), the testing set (10%) and the validating set (5%), and the equal number of negative associations are sampled randomly. The performance results of attribute prediction are presented in Table VI. This result indicates that our DCTAN performs the best compared to other baseline methods, which demonstrate that our model could dynamically co-embed the nodes and attributes in the same semantic space, where their affinities are preserved. As we can see in Table V and Table VI, the most of baseline methods perform worse in attribute prediction than in attribute inference due to the evolution of attributed network.

TABLE VII
ABLATION STUDY OF LINK RECONSTRUCT. THE BEST PERFORMANCE OF EACH METRIC IS IN **BOLD**.

Methods	DBLP-4000		DBLP-10000	
	AUC	AP	AUC	AP
DCTAN-N	.946	.957	.945	.953
DCTAN-NTI	.931	.953	.922	.930
DCTAN-NT	.934	.955	.924	.934
DCTAN-NI	.930	.952	.921	.933
DCTAN	**.958**	**.969**	**.955**	**.965**

TABLE VIII
ABLATION STUDY ON LINK PREDICTION. THE BEST PERFORMANCE OF EACH METRIC IS IN **BOLD**.

Methods	DBLP-4000		DBLP-10000	
	AUC	AP	AUC	AP
DCTAN-N	.942	.946	.958	.954
DCTAN-NTI	.924	.942	.910	.928
DCTAN-NT	.923	.942	.913	.929
DCTAN-NI	.909	.932	.902	.921
DCTAN	**.952**	**.964**	**.943**	**.958**

E. Dynamic Network Visualization

Here, we turn to answer research question **RQ5** by embedding a toy temporal attributed network (shown in Figure 1) with 34 nodes and 4 attributes for 5 time steps and visualizing the dynamic changes of the learned latent embeddings over time. To simulate the evolving of real-word network, we randomly add links and attributes to the toy network for each time steps. To intuitively evaluate the result embeddings of our DCTAN by network visualization, we first obtain 2-dimensional Gaussian embeddings for nodes and attributes of the toy network. Then, we plot the means of the resulting representations into a 2-dimensional latent space. Figure 4 presents the visualization result of DCTAN. From the visualization results, we have the following observations and conclusions:

(1) From Figure 4, we can find that all the representations of nodes and attributes are highly close to each other and their positions and the distances between each other on the 2-dimensional plane change over time. This indicates that our model is able to embed them in the same semantic space, and the learned embedding by our DCTAN can effectively capture the dynamic changes of nodes and attributes in a dynamic environment.

(2) As shown in Figure 4, the result embeddings change over time as the attributes of nodes change. To better understand this, we take two nodes (i.e. Node1 and Node2 in Figure 4) from the toy network as an example and analysis the dynamic change of their embeddings. As seen in Figure 4, the positions of the two nodes' representations on the 2-dimensional plane at time steps $4, 5$ are closer than those at time steps $1, 2, 3$. This is due to the fact that the attributes of Node1 and Node2 change over time. More precisely, at time step 4, Node2 has an attribute which also belongs to Node1, resulting in the more similarities between them. This observation indicates that our DCTAN can learn the dynamic latent representations reflecting the attribute change over time for temporal attributed network.

(3) To show the embedding changes caused by the network structure evolves, we take Node3 and Node4 as another example for analysing the dynamic change of the learned embeddings over time. As shown in Figure 4, a new link is created between Node3 and Node4 at time step 4, resulting in that the two nodes become much closer to each other compared with them in time step 3. This can be explained by the reason that such link change in dynamic network enhances the correlation between two nodes. Moreover, the neighbourhood change of node also influences its latent vector representation. For example, at time step 5, we add a new neighbour node to Node4, which is not adjacent to Node3, resulting in the positions of Node3 and Node4 on the 2-dimensional plane becoming farther. This observation demonstrates that DCTAN is able to effectively capture the temporal information of structure change of network in dynamic environment.

F. Ablation Studies

To answer research question RQ6, we conduct ablation studies on the real-world attributed networks, DBLP-4000 and DBLP-10000, with the variants of our original model DCTAN,

TABLE IX
ABLATION STUDY ON ATTRIBUTE INFERENCE. THE BEST PERFORMANCE OF EACH METRIC IS IN **BOLD**.

Methods	DBLP-4000		DBLP-10000	
	AUC	AP	AUC	AP
DCTAN-A	.952	.946	.958	.954
DCTAN-NTI	.962	.964	.965	.966
DCTAN-NT	.968	.963	.962	.960
DCTAN-NI	.966	.961	.967	.969
DCTAN	**.976**	**.968**	**.982**	**.975**

TABLE X
ABLATION STUDY ON ATTRIBUTE PREDICTION. THE BEST PERFORMANCE OF EACH METRIC IS IN **BOLD**.

Tasks	DBLP-4000		DBLP-10000	
	AUC	AP	AUC	AP
DCTAN-A	.951	.947	.945	.942
DCTAN-NTI	.954	.951	.945	.943
DCTAN-NT	.952	.950	.948	.945
DCTAN-NI	.949	.943	.944	.946
DCTAN	**.962**	**.960**	**.957**	**.953**

i.e., DCTAN-N, DCTAN-A, DCTAN-NTI, DCTAN-NT, and DCTAN-NI, for the tasks of link reconstruction/prediction and attribute inference/prediction. According to the performance results reported in Tables VII to X, we have the following findings: (1) For all the tasks, the original model DCTAN always outperforms all its variants DCTAN-A, DCTAN-N, DCTAN-NTI, DCTAN-NT, and DCTAN-NI, where some components are removed, e.g., smoothing inference networks. This demonstrates that each component of our proposed DCTAN model is necessary for effectively learning higher-quality representations to attain performance improvement on the down-stream graph analysis tasks. (2) It is notable that DCTAN-NTI, removed all the variational components, achieves comparative performance results, compared with the variants, in one of which the state transition network is removed and another the inference network is smoothed, i.e., DCTAN-NT and DCTAN-NI, respectively. This proves that the two variational components should be integrated collaboratively to infer better latent representations for nodes and attributes. (3) Our original model DCTAN, which tries to preserve both structure and attribute information, surpasses its variants DCTAN-N that only considers the topological structure and DCTAN-A that merely considers the node attributes on corresponding tasks. This demonstrates the effectiveness of jointly learning the embeddings of two different categories of entities (i.e., nodes and attributes) at the same time, and such strategy is also utilized in previous work [10, 15].

VII. CONCLUSION

In this paper, we propose a dynamic co-embedding model (called DCTAN) for temporal attributed network, which learns latent low-dimensional embeddings of both attributes and nodes while the temporal information of network can be captured. In order to learn high-quality representations and track the latent state transition tendency, our proposed model optimizes an evidence lower bound on the log-likelihood of the dynamic network sequence observation by five well-designed neural networks, i.e. belief state network, belief inference network, state transition network, smoothing inference network and decoder network. Our DCTAN goes beyond temporally inferring embeddings for temporal networks, as it may provide valuable insight and inspiration on how to extend VAE architecture to dynamically learn representations for other types of temporal data. Experimental results show that our proposed model can effectively track the latent state transition tendency for temporal network over time and outperforms both the static and dynamic network embedding baselines in both the static and dynamic graph mining tasks, such as link reconstruction, link prediction, attributed inference and attribute prediction. Moreover, the visualization of an toy example of temporal attributed network demonstrates that our DCTAN is able to learn dynamic latent representations at different time steps for the target network which reflects the change of structure and attribute over time in a dynamic environment.

As to future work, we intend to extend the proposed model to inductively cope with out-of-sample nodes and attributes which are unseen in the previous sequential networks, and enhance the predictive power to handle more intricate scenarios.

REFERENCES

[1] B. Perozzi, R. Al-Rfou, and S. Skiena, "Deepwalk: Online learning of social representations," in *SIGKDD*. ACM, 2014, pp. 701–710.

[2] A. Grover and J. Leskovec, "node2vec: Scalable feature learning for networks," in *SIGKDD*. ACM, 2016, pp. 855–864.

[3] J. Tang, M. Qu, M. Wang, M. Zhang, J. Yan, and Q. Mei, "Line: Large-scale information network embedding," in *WWW*. International World Wide Web Conferences Steering Committee, 2015, pp. 1067–1077.

[4] T. N. Kipf and M. Welling, "Semi-supervised classification with graph convolutional networks," *ICLR*, 2016.

[5] L. Zhu, D. Guo, J. Yin, G. Ver Steeg, and A. Galstyan, "Scalable temporal latent space inference for link prediction in dynamic social networks," *TKDE*, vol. 28, no. 10, pp. 2765–2777, 2016.

[6] L. Zhou, Y. Yang, X. Ren, F. Wu, and Y. Zhuang, "Dynamic network embedding by modeling triadic closure process," in *Thirty-Second AAAI Conference on Artificial Intelligence*, 2018.

[7] Y. Zuo, G. Liu, H. Lin, J. Guo, X. Hu, and J. Wu, "Embedding temporal network via neighborhood formation," in *SIGKDD*. ACM, 2018, pp. 2857–2866.

[8] J. Li, K. Cheng, L. Wu, and H. Liu, "Streaming link prediction on dynamic attributed networks," in *WSDM*. ACM, 2018, pp. 369–377.

[9] X. Wang, P. Cui, J. Wang, J. Pei, W. Zhu, and S. Yang, "Community preserving network embedding," in *AAAI*, 2017.

[10] S. Liang, X. Zhang, Z. Ren, and E. Kanoulas, "Dynamic embeddings for user profiling in twitter," in *SIGKDD*. ACM, 2018, pp. 1764–1773.

[11] L. Zhu, D. Guo, J. Yin, G. Ver Steeg, and A. Galstyan, "Scalable temporal latent space inference for link prediction in dynamic social networks," *TKDE*, vol. 28, no. 10, pp. 2765–2777, 2016.

[12] G. H. Nguyen, J. B. Lee, R. A. Rossi, N. K. Ahmed, E. Koh, and S. Kim, "Continuous-time dynamic network embeddings," in *WWW*. International World Wide Web Conferences Steering Committee, 2018, pp. 969–976.

[13] J. Li, H. Dani, X. Hu, J. Tang, Y. Chang, and H. Liu, "Attributed network embedding for learning in a dynamic environment," in *CIKM*. ACM, 2017, pp. 387–396.

[14] P. Sarkar, D. Chakrabarti, and M. I. Jordan, "Nonparametric link prediction in dynamic networks," in *ICML*, 2012, pp. 1897–1904.

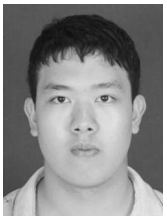
[15] Z. Meng, S. Liang, H. Bao, and X. Zhang, "Co-embedding attributed networks," in *Proceedings of the Twelfth ACM International Conference on Web Search and Data Mining*. ACM, 2019, pp. 393–401.

[16] K. Gregor and F. Besse, "Temporal difference variational auto-encoder," *ICLR*, 2019.

[17] G. Chen, J. Fang, Z. Meng, Q. Zhang, and S. Liang, "Multi-relational graph representation learning with bayesian gaussian process network," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 36, no. 5, 2022, pp. 5530–5538.

[18] J. Fang, Q. Zhang, Z. Meng, and S. Liang, "Structure-aware random fourier kernel for graphs," *Advances in Neural Information Processing Systems*, vol. 34, pp. 17 681–17 694, 2021.

[19] S. Liang, Y. Luo, and Z. Meng, "Profiling users for question answering communities via flow-based constrained co-embedding model," *ACM Transactions on Information Systems (TOIS)*, vol. 40, no. 2, pp. 1–38, 2021.
[20] T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean, "Distributed representations of words and phrases and their compositionality," in *NIPS*, 2013, pp. 3111–3119.
[21] D. Wang, P. Cui, and W. Zhu, "Structural deep network embedding," in *WSDM*. ACM, 2016, pp. 1225–1234.
[22] C. Yang, Z. Liu, D. Zhao, M. Sun, and E. Chang, "Network representation learning with rich text information," in *IJCAI*, 2015.
[23] X. Huang, J. Li, and X. Hu, "Accelerated attributed network embedding," in *Proceedings of the 2017 SIAM international conference on data mining*. SIAM, 2017, pp. 633–641.
[24] W. Hamilton, Z. Ying, and J. Leskovec, "Inductive representation learning on large graphs," in *NIPS*, 2017, pp. 1024–1034.
[25] T. N. Kipf and M. Welling, "Variational graph auto-encoders," *NIPS Workshop on Bayesian Deep Learning*, 2016.
[26] R. Zhang, Y. Zou, and J. Ma, "Hyper-sagnn: a self-attention based graph neural network for hypergraphs," *ICLR*, 2020.
[27] Z. Zhang, H. Yang, J. Bu, S. Zhou, P. Yu, J. Zhang, M. Ester, and C. Wang, "Anrl: Attributed network representation learning via deep neural networks." in *International Joint Conference on Artificial Intelligence*, 2018, pp. 3155–3161.
[28] Z. Meng, S. Liang, J. Fang, and T. Xiao, "Semi-supervisedly co-embedding attributed networks," *Advances in neural information processing systems*, vol. 32, 2019.
[29] J. Fang, S. Liang, Z. Meng, and M. De Rijke, "Hyperspherical variational co-embedding for attributed networks," *ACM Transactions on Information Systems (TOIS)*, vol. 40, no. 3, pp. 1–36, 2021.
[30] M. Rudolph and D. Blei, "Dynamic embeddings for language evolution," in *WWW*, 2018, pp. 1003–1011.
[31] R. Bamler and S. Mandt, "Dynamic word embeddings," in *ICML*. JMLR. org, 2017, pp. 380–389.
[32] Z. Yao, Y. Sun, W. Ding, N. Rao, and H. Xiong, "Dynamic word embeddings for evolving semantic discovery," in *WSDM*. ACM, 2018, pp. 673–681.
[33] W. L. Hamilton, J. Leskovec, and D. Jurafsky, "Diachronic word embeddings reveal statistical laws of semantic change," in *ACL*, vol. 1, 2016, pp. 1489–1501.
[34] Y. Kim, Y.-I. Chiu, K. Hanaki, D. Hegde, and S. Petrov, "Temporal analysis of language through neural language models," in *Annual Meeting of the Association for Computational Linguistics*, 2014, p. 61.
[35] R. Trivedi, H. Dai, Y. Wang, and L. Song, "Know-evolve: Deep temporal reasoning for dynamic knowledge graphs," in *ICML*, 2017.
[36] W. Jin, H. Jiang, C. Zhang, P. Szekely, and X. Ren, "Recurrent event network: Global structure inference over temporal knowledge graph," *ICLR-RLGM*, 2019.
[37] A. Pareja, G. Domeniconi, J. Chen, T. Ma, T. Suzumura, H. Kanezashi, T. Kaler, and C. E. Leisersen, "Evolvegcn: Evolving graph convolutional networks for dynamic graphs," *ICLR*, 2019.
[38] D. Xu, C. Ruan, E. Korpeoglu, S. Kumar, and K. Achan, "Inductive representation learning on temporal graphs," *ICLR*, 2020.
[39] R. Trivedi, M. Farajtabar, P. Biswal, and H. Zha, "Dyrep: Learning representations over dynamic graphs," *ICLR*, 2019.
[40] A. Venkatraman, N. Rhinehart, W. Sun, L. Pinto, M. Hebert, B. Boots, K. Kitani, and J. Bagnell, "Predictive-state decoders: Encoding the future into recurrent networks," in *NIPS*, 2017, pp. 1172–1183.
[41] D. P. Kingma and M. Welling, "Auto-encoding variational bayes," *ICLR*, 2013.
[42] J. Leskovec and A. Krevl, "SNAP Datasets: Stanford large network dataset collection," Jun. 2014.
[43] R. A. Rossi and N. K. Ahmed, "The network data repository with interactive graph analytics and visualization," in *AAAI*, 2015. [Online]. Available: http://networkrepository.com
[44] P. Sen, G. Namata, M. Bilgic, L. Getoor, B. Galligher, and T. Eliassi-Rad, "Collective classification in network data," *AI magazine*, vol. 29, no. 3, pp. 93–93, 2008.
[45] H. Yin, A. R. Benson, J. Leskovec, and D. F. Gleich, "Local higher-order graph clustering," in *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2017, pp. 555–564.
[46] P. Panzarasa, T. Opsahl, and K. M. Carley, "Patterns and dynamics of users' behavior and interaction: Network analysis of an online community," *Journal of the American Society for Information Science and Technology*, vol. 60, no. 5, pp. 911–932, 2009.
[47] W. Hamilton, Z. Ying, and J. Leskovec, "Inductive representation learning on large graphs," in *Advances in neural information processing systems*, 2017, pp. 1024–1034.
[48] C. Yang, Z. Liu, D. Zhao, M. Sun, and E. Y. Chang, "Network representation learning with rich text information," in *IJCAI*, vol. 2015, 2015, pp. 2111–2117.
[49] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *ICLR*, 2014.
[50] A. Paszke, S. Gross, S. Chintala, G. Chanan, E. Yang, Z. DeVito, Z. Lin, A. Desmaison, L. Antiga, and A. Lerer, "Automatic differentiation in pytorch," in *NIPS*, 2017.
[51] N. Z. Gong, A. Talwalkar, L. Mackey, L. Huang, E. C. R. Shin, E. Stefanov, E. R. Shi, and D. Song, "Joint link prediction and attribute inference using a social-attribute network," *TIST*, vol. 5, no. 2, p. 27, 2014.
[52] D. Chakrabarti, S. Funiak, J. Chang, and S. Macskassy, "Joint inference of multiple label types in large networks," in *ICML*, 2014, pp. 874–882.
[53] C. Yang, L. Zhong, L.-J. Li, and L. Jie, "Bi-directional joint inference for user links and attributes on large social graphs," in *WWW*. International World Wide Web Conferences Steering Committee, 2017, pp. 564–573.



Shaowei Tang is currently a postgraduate student at the School of Data and Computer Science, Sun Yat-sen University. His research interests include knowledge graph and graph mining.

Zaiqiao Meng received his Ph.D. degree from the Sun Yat-sen University in 2018. He is currently a lecturer at the University of Glasgow. His research interests lie mainly in the areas of information retrieval, recommender systems, machine learning and graph mining.

Shangsong Liang is with the Sun Yat-sen University and the Mohamed bin Zayed University of Artificial Intelligence. He received a Ph.D. degree from the University of Amsterdam in 2014. His expertise lies in the fields of information retrieval and text mining. He worked as a (visiting) postdoctoral research scientist at the University of Massachusetts Amherst and the University College London, and has extensively published his work in top-tier conferences and journals, including SIGIR, KDD, WWW, CIKM, AAAI, WSDM, NeurIPS, TKDE and TOIS. He is the recipient of an Outstanding Reviewer Award in SIGIR 2017 and is serving as an editor for Information Processing and Management.