

Inferring Network Structure with Unobservable Nodes from Time Series Data

Mengyuan Chen,¹ Yan Zhang,¹ Zhang Zhang,¹ Lun Du,² Shuo Wang,¹ and Jiang Zhang¹

¹*School of System Science, Beijing Normal University, No. 19, Xijiekou Wai Street, Beijing, China, 100875*

²*Microsoft Research, No. 5 Danling Street, Haidian District, Beijing, China, 10080*

(*Electronic mail: zhangjiang@bnu.edu.cn)

(Dated: 22 February 2022)

Network structures play important roles in social, technological and biological systems. However, the observable nodes and connections in real cases are often incomplete or unavailable due to measurement errors, private protection issues, or other problems. Therefore, inferring the complete network structure is useful for understanding human interactions and complex dynamics. The existing studies have not fully solved the problem of inferring network structure with partial information about connections or nodes. In this paper, we tackle the problem by utilizing time-series data generated by network dynamics. We regard the network inference problem based on dynamical time series data as a problem of minimizing errors for predicting states of observable nodes and proposed a novel data-driven deep learning model called Gumbel-softmax Inference for Network (GIN) to solve the problem under incomplete information. The GIN framework includes three modules: a dynamics learner, a network generator, and an initial state generator to infer the unobservable parts of the network. We implement experiments on artificial and empirical social networks with discrete and continuous dynamics. The experiments show that our method can infer the unknown parts of the structure and the initial states of the observable nodes with up to 90% accuracy. The accuracy declines linearly with the increase of the fractions of unobservable nodes. Our framework may have wide applications where the network structure is hard to obtain and the time series data is rich.

A complex system is composed of many components that interact with each other. In general, the nodes of a complex network are used to represent the elements in the system, and the edges between nodes are used to represent the interactions, social system, economic system, and biological system are all complex systems. The dependence of nodes in systems is complex. How to mine the network structure of the complex system to help us better understand the behavior of the system has always been a research and challenging problem. This paper aims to study the problem of network structure inference under the absence of system node information – network completion. The existing research on network completion is to infer the complete network structure in the complex system from the observed information. According to the information obtained, the existing studies can be divided into three categories: one is to use rich node feature information, the other is to use observed structural information, and the third is to make inferences under the scenario in which only temporal sequence information can be observed. Most of the current research assumes that all the node information can be observed, and there is less discussion about the missing information of multiple nodes due to artificial or technical constraints. The contribution of this paper is that it proposes a data-driven, end-to-end method to solve the problem of network completion, and puts forward the application of subgraph matching algorithm to the network completion method, which effectively solves the evaluation problem in network completion.

I. INTRODUCTION

Network structure plays more and more important roles in social, technological, and economic systems¹⁻⁴. The connection patterns of a social network determine how fast the opinions or ideas can spread in social media⁵⁻⁷; the structure of a supply chain network between companies influences the safety of the whole market because risk may propagate along with the links^{8,9}; the topology of the cooperation network plays a critical role for scientific innovation and individual development for young scientists^{10,11}. Nowadays, many big data analysis, such as recommendation, node importance mining, community clustering, etc., rely on high-quality link data¹². However, the data of network structure is always incomplete or even unavailable either because measuring binary links is costly or the data of weak ties is unobservable^{9,13,14}. Therefore, it is urgent to find a way to infer the complete network structure according to non-structural information^{15,16}.

Link prediction, as the traditional task in network inference, tries to infer the lost links in network structure according to the linking patterns of existing connections^{17,18}. Although numerous algorithms have been developed to complete the unobservable links of a large network with high accuracy^{19,20}, all of these approaches require the complete node information but it is always unavailable in practice^{13,21}. Link prediction cannot solve the inference problem under the condition that the network contains unobservable nodes. In real cases, we can either obtain node information of the only partial network or without any information about links^{1,22}, as a result, conventional link prediction algorithms cannot work.

Network completion methods have been developed in recent years trying to tackle the problem we discussed above, that is, to infer the missing connections on unobservable nodes

according to the linking patterns between observable nodes. The methods can be categorized into the traditional expectation maximum (EM) method^{23,24} and graph neural network based methods²⁵⁻²⁷. As an example of the expectation maximum method²³, Kronecker Graph Expectation Maximum (KronEM) algorithm based on Kronecker Graph model²⁸ can complete the network according to the observable links. Although their algorithm can obtain a relatively high accuracy of recovering missing links, the self-similarity property is required for the underlying network structure as an implicit condition, which is always violated by some networks²⁸. On the other hand, with the booming development of deep learning on graphs²⁵⁻²⁷, researchers applied graph convolution network (GCN) liked models on the network completion problem. Xu et al. proposed a GCN based model, which regards the process of completing a graph as a network growth process, and learns the rules of the growth to complement the full network²⁹. Tran et al. solved the problem by training a graph generative model to learn the connection patterns among a large set of similar graphs and use these patterns to infer the missing connections²¹. All of these network completion methods depend on a partially observable network structure because they try to discover the latent patterns of the observable connections and to infer the unknown structures. Nevertheless, in some cases, the network structures are totally unknown and only some signals of observable nodes can be obtained, such as biological network³⁰ and social network¹³. How can we infer the whole network structure without any information on connection patterns?

In fact, time-series data of observable node behaviours can be another important information source^{31,32} which is more or less ignored by previous studies. For example, in an online social network, we can only observe the discrete retweet events between a large set of users, neither their features like sex, education, etc. nor their connection information is unavailable; In a stock market, all the information that we can obtain is the prices of different stocks, the connections between the stocks are unknown. Thus, can we develop a method to infer the network structure according to the time series data representing the observable states of nodes? A large number of methods such as Granger causality^{33,34}, correlation measurements³⁵⁻³⁷, driving response³⁸, compressed sensing^{19,39-41}, and graph network^{42,43} etc., have been proposed for reconstructing network from time series data. However, these methods can only recover the network structure of observed networks and the functional forms of dynamics are always limited by methods. Can we infer all network structure including unobserved part and unobserved node states with partial time series data of observable nodes? Some works try to recover the information of hidden variables by learning the dynamics of a system⁴⁴. However, these works are always based on a grid network and leave the general heterogeneous network structure never discussed. Actually, completing or refining links in a network by node properties and labels is possible as shown in^{45,46}, better network can be obtained if we only try to improve the performance of node classification task. Thus, a general framework for reconstructing network topology, completing missing structures, and learning various

types of dynamics, from the time series data is possible and necessary.

In this paper, we develop a universal framework called Gumbel-softmax Inference for Network (GIN) to infer the network structure and node information from the time series data with missing nodes. We solve the problem by finding an optimized network structure, a set of appropriate initial states, and an approximator of the network dynamic such that the errors between the observed time series of the observable nodes and the generated time series according to the GIN model is minimized. GIN consists of a network generator, an initial state generator, and a dynamics learner. The network generator is implemented by Gumbel-softmax technique, which can use stochastic gradient descent to differentially optimize a network. Dynamics learning is realized by a Graph Network (GN) model. This paper is organized as follows: in section II we will formulate the network inference problem in an optimization framework and illustrate the concrete design of each module; the experimental results are discussed in section III. We also point out the advantages and weak points of this work which left for future works in section IV.

II. PROBLEM AND METHODS

In this paper, we focus on the inference problem of network structure, initial states, and the network dynamics based on state time series of observable nodes.

A. Problem Definition

At first, a formal definition is given. Suppose our studied system has an interaction structure described by a binary graph $G = (V, E)$ with an adjacency matrix A , where $V = \{v_1, \dots, v_N\}$ is the set of nodes, or interchangeably referred to as vertices, and N is the total number of nodes, $E = \{e_{ij}\}$ is the set of edges between the nodes, and A is a binary matrix of which each entry equals 0 or 1.

The network dynamic $\mathcal{S}(\psi, A)$ is defined on the graph G , where ψ is the dynamical rule which mapping the states of nodes $\mathbf{x}^t = (x_1^t, x_2^t, \dots, x_N^t) \in \mathcal{R}^{n \times d}$ at time t to the states \mathbf{x}^{t+1} at time $t+1$, where $\mathbf{x}^{t+1} = \psi(\mathbf{x}^t)$, and d is the dimension of the states. Thus, time series can be generated by the network dynamic \mathcal{S} , which are denoted by $\mathbf{x}^{0:T} = (\mathbf{x}^0, \mathbf{x}^1, \dots, \mathbf{x}^T)$, where \mathbf{x}^0 is the initial state and T is the total time length of the series.

However, not all node states can be observed by us. Therefore, the set of nodes V can be divided into two parts: observed nodes V_o and unobserved nodes V_u , where $V_o \cup V_u = V$ and $V_o \cap V_u = \emptyset$, and the corresponding state vectors \mathbf{x}^t can also be decomposed into two parts: $\mathbf{x}^t = \mathbf{x}_o^t \oplus \mathbf{x}_u^t$, where \oplus is the vector concatenation. Thus, only the partial vector \mathbf{x}_o can be observed.

Similarly, all of the connections E can also be decomposed into observable connections E_o and unobservable connections E_u , and $E_o \cup E_u = E$, and $E_o \cap E_u = \emptyset$. The corresponding adjacency matrix can also be decomposed, $A = A_o \oplus A_u$. Where,

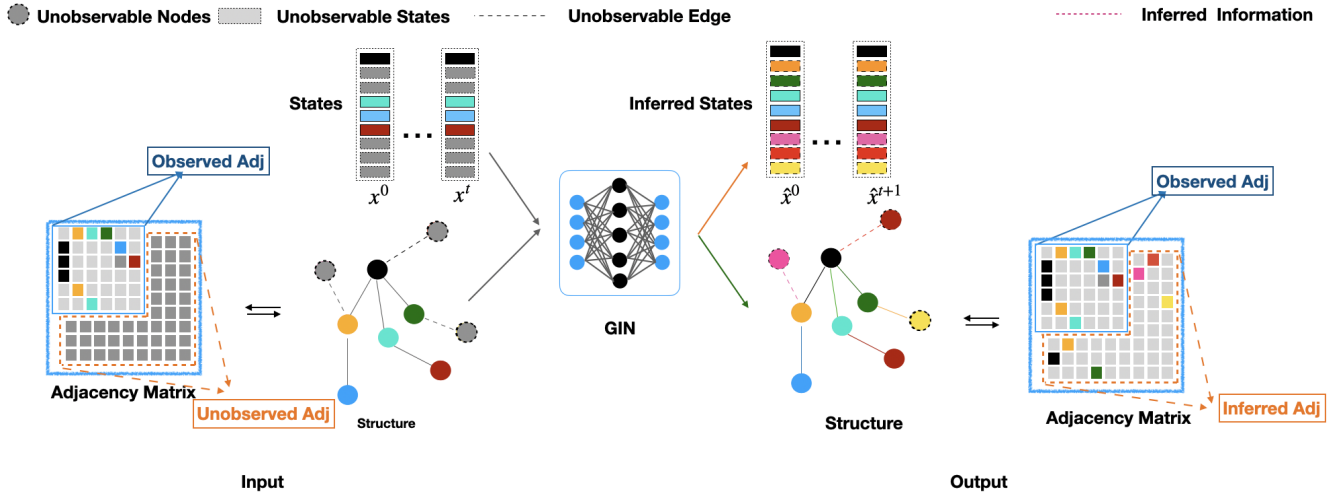


FIG. 1. The states of some nodes are missing (with dashed box) and only partial network structure can be observed (with bold circles), and the aim is to infer the missing information (the dashed colored circles). In practice, network completion means to infer the “missing” element (with dark grey color) in the adjacency matrix.

A is the adjacency matrix of the whole graph G , and A_o is the adjacency matrix of the observed connections, and A_u is the one of unobserved connections, \oplus is the matrix concatenation after appropriate rearrangement of matrix entries (see the example adjacency matrix in Figure 1, the gray colored entries with the inverted L shape is the unobserved part).

Note that the two parts of V and the two parts of E do not necessarily have a corresponding relationship. Figure 1 shows a general case with the overlap between observable nodes and unobservable connections.

Then, our task is to infer all the unobserved information including node dynamical rules ψ , unobserved connections A_u , and unobserved node states x_u^t according to all the known information including time series of observed nodes $x_o^{0:T}$ and observed connections A_o .

B. Optimization Problem Formulation

The network inference problem can be formulated as an optimization problem that finds a set of optimal parameters α, β, γ , to minimize the error value between the state estimation value and the ground-truth, which is the objective function Equation 1.

$$\min_{\alpha, \beta, \gamma} L(\alpha, \beta, \gamma) = \sum_{t=1}^T D(x_o^t, \hat{x}_o^t(\alpha, \beta, \gamma)) + \lambda \|\hat{A}(\beta)\| \quad (1)$$

such that:

$$\hat{x}_o^t \oplus \hat{x}_u^t = \hat{\psi}_\alpha(x_o^{t-1} \oplus \hat{x}_u^{t-1}, \hat{A}(\beta)), \forall t > 1, \quad (2)$$

$$\hat{x}_u^0 = \rho(\gamma) \quad (3)$$

Here, $D(x, y)$ is a measure of the closeness between the state x_o^t and $\hat{x}_o^t(\alpha, \beta, \gamma)$, and it can be a cross-entropy measure when the states are binary or Mean Absolute Error (MAE) when the states are real numbers. $\hat{\psi}_\alpha(\cdot)$ is a dynamical rule approximator to estimate ψ parameterized by α . Notice that, to estimate a better state at time step t , we use the information from real data of the observable nodes x_o^{t-1} at step $t-1$. In this way, we can iteratively apply $\hat{\psi}_\alpha$ to the estimated state in the previous time to obtain an estimated evolutionary trajectory $(\hat{x}^1, \hat{x}^2, \dots, \hat{x}^T)$ starting from the state $\hat{x}^0 = x_o^0 \oplus \hat{x}_u^0$. And this trajectory must be similar to the real trajectory (x^1, x^2, \dots, x^T) . Further, $\hat{A}(\beta)$ is the estimate of the adjacency matrix A with the parameter β . And $\hat{x}^0 = \rho(\gamma) \in \mathcal{R}^{M \times d}$ is an estimate of the initial states of unknown nodes which are parameterized by γ . The second term in Equation 1 is the structural loss which can compel the generated adjacency matrix to be sparse, and $\lambda > 0$ is the parameter to balance the relative importance between the structural loss and the prediction error.

We have converted the network inference problem into an optimization problem, but this is a very general framework in which concrete implementation should be given.

C. Network Inference Framework

1. Implementation Gumbel-softmax Inference for Network

To implement the framework mentioned in the previous paragraph, we propose a concrete implementation called Gumbel-softmax Inference for Network (GIN). GIN is composed of a network generator based on gumbel softmax technique and a dynamics learner based on graph network technique^{43,47}.

The Framework is shown in Figure 2. The inputs of our

model are the states x_o^t of observable nodes and the observable adjacency matrix A_o . Correspondingly, the outputs are the complete network structure and the future states of all nodes.

At first, the candidate network is generated by a series of gumbel softmax sampling processes parameterized by a matrix $\beta_{N \times N}$, that is,

$$A_{ij} = \frac{\exp((\log(\beta_{ij}) + \xi_{ij})/\tau)}{\exp(\log(\beta_{ij}) + \xi_{ij})/\tau) + \exp(\log(\beta_{ij}) + \xi'_{ij})/\tau}, \quad (4)$$

where β_{ij} is the probability of the connection between node i and node j , and ξ_{ij} is the i.i.d random variable of the standard Gumbel distribution, and τ is the temperature parameter. When τ goes to zero, A_{ij} will converge to 0 or 1. Equation 4 simulates the sampling process of generating A_{ij} with the probability ξ_{ij} , however, it is differentiable such that it can be adjusted by the gradient descent method.

The second module of GIN is the initial state generator. Because the iteration of the dynamics learner ψ_α requires the initial state of all nodes, however, the states of unobserved nodes are missing. We generate the initial states of these unobserved nodes with the initial state generator $\rho(\gamma)$ parameterized by γ . Here, ρ can be simply an identity function (this is equivalent to sample initial state by γ directly) or a function to specify the value boundary of the initial states (for example, a sigmoid function can limit the initial states to the interval $(0, 1)$).

Third, when the candidate network and initial state are generated, they will be fed into the dynamics learner module. We assume that the dynamics ψ is node symmetric, therefore, we can use a graph network $\hat{\psi}_\alpha$ parameterized by α to implement the dynamics learner as shown in Figure 3.

We train and update the parameters of the three modules in each epoch. After the predicted states of the unobservable nodes are obtained, the loss function can be calculated by comparing the predicted states and the real ones. And we implement the back-propagation algorithm to obtain the gradient values and update the parameters of the three modules simultaneously. We layout the pseudo-codes of GIN in Algorithm 1 to show the details.

2. Graph Matching Problem in the Evaluation Process

After training the GIN framework, we need to evaluate the effect of the network inference. However, it is difficult to evaluate a network completion algorithm in real life because the missing connections are unknown. Our strategy is to find a real network as the ground truth, cut off some of the nodes and edges for testing, and then compare the inferred part of the algorithm with the real network.

However, a new problem, graph matching between the inferred sub-graph and the real one arises during this comparison, because the unobservable nodes between the inference and the ground truth should be aligned before evaluating.

To solve the problem, we can search for all node alignment and find the best one. Here, the best alignment means that

Algorithm 1 : GIN algorithm

```

1 Input: the observed adjacency matrix  $A_o$  if have;
           the time series of all or partial nodes  $x_o^{0:T}$ ;
           the number of observed nodes  $N_o$  if have.
           the number of unobserved nodes  $N_u$  if have.
2 Output: the predict adjacency matrix  $\hat{A}$ ;
           the initial states of the unobserved nodes and the predict
           states of all nodes  $\hat{x} = \{\hat{x}_u^0, \hat{x}_u^{1:T+1}, \hat{x}_o^{1:T+1}\}$ .
# Initialization
4 Initialize Dynamics Learner parameters  $\alpha$ 
5 Initialize Initial States Generator parameters  $\gamma$ 
6 Initialize Network Generator parameters  $\beta(N_u)$ 
# Training
7 for each epoch do
8   Get initial states of unobserved nodes:  $\hat{x}_u^0 = \rho(\gamma)$ 
9   Sample unobserved adjacency matrix:
      $\hat{A}_u = \text{Network Generator}(\beta)$ 
10   $\hat{A} \leftarrow (A_o \oplus \hat{A}_u)$ 
11  for  $t=0, \dots, T$  do
12    Concatenate nodal states:  $\hat{x}^t \leftarrow (x_o^t \oplus \hat{x}_u^t)$ 
13    for  $i=1, \dots, N_o$  do
14       $\hat{x}_o^t[i] \oplus \hat{x}_u^t[i] \leftarrow \text{Dynamics Learner}(\hat{A}[i], \hat{x}^0, \alpha)$ 
15      loss  $\leftarrow \text{Compute Loss}(\{x_o^t[i]\}, \{\hat{x}_o^t[i]\})$ 
16      update  $\gamma, \beta, \alpha$  with the gradient of loss
17    end
18  end
19 end

```

each node pair in the alignment has the most similar neighbor relationship with each known node. Thus, the graph matching problem can be formulated as another optimization problem:

$$\min_{P \in \mathcal{P}(N_u)} \|A - (I_{N_o} \oplus P)\hat{A}(I_{N_o} \oplus P)^T\|_F^2 \quad (5)$$

Where, A and \hat{A} represent the adjacency matrices of the ground truth and the inference, respectively. P is a permutation matrix (node alignment) with size $n \times n$, and $\mathcal{P}(N_u)$ is the set of all possible permutation matrices with size N_u . I_{N_o} is an identity matrix with size N_o . The symbol \oplus represents the concatenation of matrices. Therefore, $A(I_{N_o} \oplus P)^T$ means the rearrangement of the rows and columns of the matrix A with the observable part unchanged. Thus, formula 5 means to find an optimized permutation P of the unobservable nodes such that the adjacency matrices between the inference and the ground truth can be as similar as possible.

However, there are $N_u!$ possible permutations such that finding an optimized permutation by brute force searching is impossible. Therefore, we use Seed Graph Matching (SGM) algorithm to solve this NP-hard problem⁴⁸.

At first, the objective function in Equation 5 can be expanded as:

$$\|A - (I_{N_o} \oplus P)\hat{A}(I_{N_o} \oplus P)^T\|_F^2 = \|A\|_F^2 + \|\hat{A}\|_F^2 - 2 \cdot \text{Tr}\left(A^T(I_{N_o} \oplus P)\hat{A}(I_{N_o} \oplus P)^T\right) \quad (6)$$

where $\|\cdot\|_F$ is the Frobenius norm on matrices. Then, the

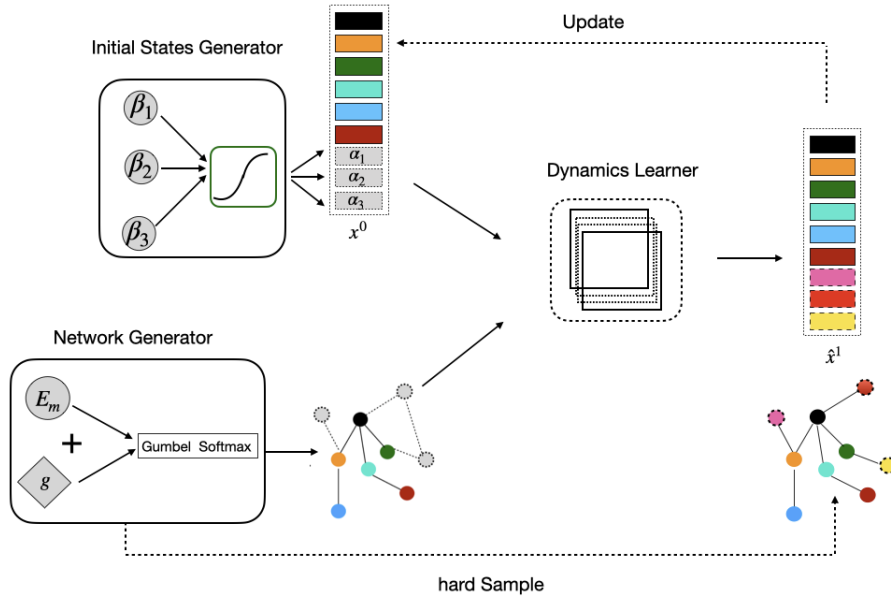


FIG. 2. The modules of the GIN model. At first, the network structure and the initial state can be generated by the network generator and initial state generator modules, respectively. After that, they are input into the dynamics learner to output the predicted value of the node states at the next time step.

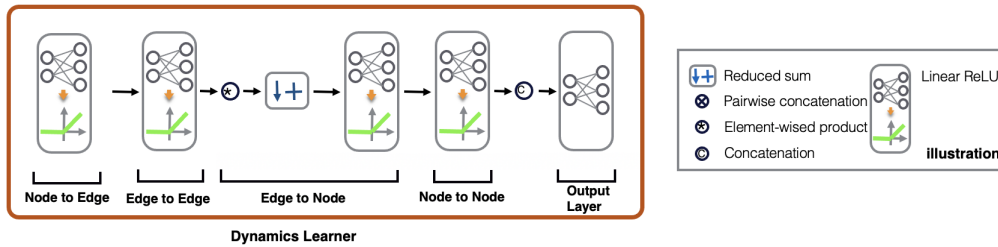


FIG. 3. The dynamics learner consists of four parts: (1) Node to Edge: aggregating the original information of nodes to form representations of edges; (2) Edge to Edge: update the edge representations; (3) Edge to Node: aggregate all information on neighbouring edges of each node to form a new feature vector of the current node; (4) node to node: update the node representations; (5) Output: finally, concatenate the node representations and the input state vectors of node i to feed into a feedforward network, and output the prediction of the next state all nodes.

minimization problem can be further simplified to the maximization problem:

$$\max_{p \in \mathcal{D}(N_u)} J(P) = \text{Tr} \left(A^T (I_{N_o} \oplus P) \hat{A} (I_{N_o} \oplus P^T) \right). \quad (7)$$

Nevertheless, this optimization problem is also hard to solve because P is a permutation matrix with binary entries. We then relax the problem by allowing the matrix P to be a doubly stochastic matrix such that the value range of each entry can be extended to the interval $[0, 1]$ as suggested by the SGM algorithm. After that, the conjugated optimization method can be used to optimize $J(P)$. The details can be referred to⁴⁸. As reported by⁴⁸, when the similarity of the two graphs is more than 90% and the number of matched nodes is more than 15, the matching accuracy of this algorithm can be more than 90%. In our experiment, the parameters are the same as those in⁴⁸.

III. EXPERIMENTAL RESULTS

Our framework and algorithms are universal because they can be applied to any network structure and any type of time series data such as continuous or binary states.

A. Data Set

We test our method on both synthetic and empirical social networks.

1. Synthetic Network

We generate synthetic networks by well-known network models such as: ER(Erdos-Renyi network)⁴⁹, WS(Small

TABLE I. Network parameter of social network

Network	N^a	E^b	$\langle k \rangle^c$	$\langle C \rangle^d$	$\langle r \rangle^e$
Karate	34	78	4	0.5706	-0.4756
Dolphins	62	159	5	0.2589	-0.0435
Email-partial	143	623	8	0.4339	-0.01953
Dorm	217	2672	24	0.399	0.1195
Email	1133	10902	4	0.220	0.0782
Blog	1224	19025	15	0.210	-0.2200

^a number of nodes

^b number of edges

^c average degree

^d average clustering coefficient

^e degree-degree correlation coefficient

world network)⁵⁰ and BA(Scale-free network)⁵¹. Next, we summarized the parameter settings for generating synthetic networks.

- **ER.** In the ER model, nodes are linked to each other with the probability $p = 0.04, 0.013, 0.01$ for the networks with the number of nodes as 100, 300 and 1000, respectively.
- **WS.** The WS model can be used to generate locally clustered networks. In our setting, we at first connect 4 closest neighbors on a ring, and we rewire a link randomly with 0.3 probability.
- **BA.** The BA model can simulate the scale-free property of real networks. The preferential attachment rule is used to grow a BA network. At the beginning, there are $m_0 = 20$ nodes have been existed as the seeds. Then, a new node is added which will connect $k = 2$ existing nodes with preferential attachment rule.

2. Empirical Social Networks

We also select 6 real social networks with different structures as the representatives of the empirical networks. Except for Dorm, the connections of these networks are undirected, which means that the transmission of information between nodes is mutual rather than one-way.

The basic structural parameters of the three social networks are shown in Table I

B. Time Series Data of Network Dynamics

A large number of time series data is required to implement our approach, however, it is hard to obtain from the real scenario because the problems of privacy and measurement are concerned. Therefore, we use synthetic time series data generated from the network dynamics instead of real data. Two different types of time series data (binary and real-valued) are tested because both types can be processed by our approach.

The first type of time series is binary which simulates the process of opinion spreading on a social network. We use the

well known Voter model to simulate the opinion dynamic on network. The Voter model is introduced by Richard A. Holley and Thomas M. Liggett in 1975^{52,53}. Suppose there are N interacting agents connected to form a network. Initially, each agent has a distinct ‘‘opinion’’ represented by $x_i = \{0, 1\}$. At each time t , any agent i will have a chance to change his ‘‘opinion’’, and the probability to adopt the opinion is determined by the relative fraction of the same opinion in all of i ’s neighbors.

The second type of dynamics on the social network has real-valued state. This models the cases that the psychological states of different people (e.g., the expectation price of a stock or a commodity) can influence each other via social connections.

We choose the chaotic network dynamic Coupled Mapping Network (CMN) as our candidate to generate the time series. A Coupled map network (CMN) model is a network dynamic with discrete time and continuous state which is proposed by Kaneko in 1992⁵⁴. Each element on a network consists of a logistic map coupled to their neighbors, this can be written as

$$x_i^{t+1} = (1 - \varepsilon)f(x_i^t) + \frac{\varepsilon}{|N_i|} \sum_{j \in N_i} f(x_j^t) \quad (8)$$

where $x_i^t \in [0, 1]$ is the state of node i at time t , N_i represents node i ’s neighbors, $\varepsilon \in (0, +\infty)$ is the coupling constant which can tune the system behavior, and the local map $f(x)$ is the logistic map:

$$f(x) = x(1 - x). \quad (9)$$

In our experiments, we set $\varepsilon = 3.5$

C. Data Preparation

We have evolved discrete Voter dynamics on the synthetic network and real social networks, and continuous CMN dynamics on synthetic networks. In order to generate time series data for each node, we first generate s initial states for each node. In each initial state, we evolve T time steps forward through the dynamic function. In the process of model training, t time-step state information is used. The number of training data we derived from the dynamics of CMN is $S = s \times T/t$, the number of data evolving from the Voter dynamics is $S = (T - t + 1) \times s$. In all experiments, we set $t = 2$. The T values of CMN and Voter dynamics are 100 and 51, respectively. On networks of different sizes and tasks, we generate different size data sets by adjusting s .

On network completion tasks, including part of the network structure is known and no network structure, we set s to 50 in a synthetic network of 100 nodes, and generated 2.5k data sets with a time step of 2, and in 300 nodes set s to 400 on the synthetic network and generate 20k data sets based on CMN dynamics. For Voter dynamics, we generated 5k and 15k volume data sets with steps of 100 and 300 on the synthesized network of 100 nodes and 300 nodes, respectively. In particular, we used 15K of data on a 100-node ER network. On the karate, Dolphin, and Email networks, s is 20, 200, and 300,

and 1k, 10k, and 15k data set are generated respectively. The division ratio of the training set, test set and validation set is 5:1:1.

For the network reconstruction task, on the continuous data set generated on the CMN dynamics, we generate 12k, 10k and 60K data on 10,100 and 1000 nodes of the WS network respectively. The ratio of the training set, test set and verification set is 10:1:1. We set the value at 10 and 100 respectively. 12k, 10k, and 60k data are generated on the WS network of 1000 nodes. The division ratio of training set, test set and verification set is 10:1:1. For Voter dynamics, the division ratio of the data set is 5:1:1. The s values of the WS network with 10, 100, and 1000 nodes are 400, 200, and 2000, and the data volume is 20k, 10k, and 100k. On the real network Dorm, Blog s is 1200, 3200, data volume is 60k, 160k respectively.

In appendix, we summarize the amount of experimental data. For each experiment, we repeat three times and then calculate the average value to report.

D. Experimental Setup

To implement the experiments, several representative tasks are set. The parameters in different situations are also given in this subsection.

1. Tasks

Our framework GIN can be applied to any case with or without observable nodes and partial networks. Without lose of generality, three specific tasks are designed as follows:

- **Network Completion with Partial Structural Information:** In this task, we randomly select a fraction of nodes as unobservable nodes, and we remove the corresponding time series and the corresponding entries of the adjacency matrix (all connections related to the unobservable nodes). And the incomplete data is fed on the framework to ask GIN to infer the unknown adjacency matrix and the unobservable initial node states.
- **Network Completion without Structural Information:** In this task, we need to randomly select some nodes as unobservable, and remove the corresponding time series to feed into GIN. Nevertheless, different from the previous task, the partial network is never known for the framework.
- **Network Reconstruction:** In this task, we need to reconstruct all links from the observed time series, and all nodes are observable.

2. Parameter Settings

To achieve good results in the tasks mentioned, we need to set up the parameters in GIN.

In general, we set parameters in all experiments as follows:

(1) In the dynamics learner module, we use a 4-layered MLP as a node shared function as shown in Figure 3. The activation function of each layer is ReLU. The model parameter α is randomly initialized. We use 32 hidden units in each layer in most cases. However, it is 64 on the Voter model.

(2) In the initial state learning module of the network completion task, discrete and continuous datasets have different initial state generation methods. The nodes' states are usually coded by one-hot vectors on the data set generated by Voter. For the data sets of CMN, We use sigmoid as the function ρ to map the parameters γ into the interval $[0, 1]$

(3) In the network generator module, because the adjacency matrix constructed is symmetric, we just randomly generate $N(N-1)/2$ elements of the upper triangular part of the adjacency matrix via the gumbel-softmax sampling process parameterized by β which are sampled by a normal distribution $N(0, 0.1)$, and we add up the transposed triangular matrix to obtain a complete symmetric adjacency matrix. For example, in Karate Network, we generate only $33 * 17$ parameters, which is the number of triangular elements on the adjacency matrix except for diagonals.

The above three modules are optimized by using Adam optimizer, the learning rates of (1) to (3) module are set to 0.004, 0.1, and 0.001, respectively. The learning rate of the state learner is higher than that of the other two modules. This is because states have too many parameters, and it is easy to fall into local optimums. For each epoch, we randomly select 1,024 samples to train and can have better results after training about 500 epochs.

The structural loss parameter λ is set to be zero in all tasks except the network reconstruction task which is set to be 0.0001.

3. Performance on Network Completion with Partial Structural Information

First, we test the performance of GIN on the network completion task with partial structural information. According to our investigation so far, we do not find a method that can be applied directly to solve the network completion task based on time series data, thus, we do not compare it to other models. We carry out the experiment of the GIN model on network completion on 100-scale and 300-scale networks, where the percents of missing nodes are 10.

In table II, the first numbers in the $N-N_u$ column represent the total network size, and the second numbers are the numbers of nodes being removed. For example, 100-10 indicates that we remove 10 nodes from a whole network with 100 nodes. The indicator of unobs-AUC refers to the Area Under the ROC Curve of the prediction for the unobservable part of the network. Also, we display some details about the results, such as the unobs-ACC(net) which is the proportion of elements that correctly estimated adjacency matrix \hat{A} , and the values of the True Positive Rate (TPR), False Positive Rate (FPR) of the unobservable part. The obs-ACC/MSE (states) is the accuracy of the predicted future states of observed nodes for the binary dynamic (Voter), and it is the mean square error

TABLE II. Network Completion Performance on different networks and dynamics

Dynamics	$N - N_u$	Network	Unobs-AUC	Unobs-ACC(net)	Unobs-TPR	Unobs-FPR	Obs-ACC/MSE
Binary	100-10	ER	0.8355±0.103	0.8933±0.0152	0.6241	0.1002	81.01%
		WS	0.9218±0.005	0.8333±0.006	0.8869	0.1691	82.07%
		BA	0.9233±0.006	0.8133±0.023	0.8993	0.1844	82.19%
	300-20	ER	0.9550±0.009	0.9600±0.000	0.6385	0.0387	77.29%
		WS	0.9583±0.005	0.9400±0.006	0.7672	0.0539	79.33%
		BA	0.9387±0.004	0.9567±0.006	0.6054	0.0397	79.31%
		34-3	Karate	0.8241±0.05	0.6800±0.144	0.7847	0.4042
	62-6	Dolphins	0.8742±0.011	0.7900±0.010	0.8366	0.2157	83.50%
	143-14	Email-partial	0.7915±0.010	0.8550±0.007	0.4964	0.1238	76.52%
	Continuous	100-10	ER	0.9839±0.008	0.9500±0.008	0.8723	0.0441
WS			0.9463±0.032	0.9033±0.052	0.8509	0.1185	9.68E-06
BA			0.9282±0.003	0.9333±0.0205	0.8282	0.0580	7.99E-06
300-20		ER	0.9902±0.008	0.9533±0.023	0.9579	0.0464	1.42E-06
		WS	0.9463±0.082	0.9033±0.137	0.8509	0.1185	9.68E-06
		BA	0.9286±0.017	0.9467±0.032	0.8259	0.0487	4.13E-06

for the continuous dynamic (CMN).

GIN can perform well on both tasks of network inference and observed node states prediction. The accuracy of the state of observable recovery is over 77% for the binary data set. And the relative error rate is close to 0 in the continuous data sets. Besides, our model maintains a stable accuracy on different network structures. Table II demonstrates that the accuracy of the binary data set is lower than that of the CMN data sets especially on the FPR results and also has a larger deviation of the dynamic state prediction value. It can be explained by the fact that the information propagation process of the Voter dynamic is stochastic, however, the CMN dynamic is deterministic.

We show a heat map Figure 4 comparing the inferred adjacency matrix graph with the real adjacency matrix to describe the effect of our completion more clearly. We set a reasonable threshold to turn the inferred probability adjacency matrix into an adjacency matrix that is either zero or one, then stack it on top of ground truth and the following heat map is plotted finally. Both the brown and yellow boxes in the figure represent the observed local network structure and the remaining invert "L" shape is the unobservable part in which the green and red squares represent elements that are incorrectly inferred. On the Dolphin network, red and green elements account for about 20%, and on the Karate network, the proportion of red and green elements is about 24%. Notice that, all correctly labeled matrix units are concentrated in the same row or column, which shows that our GIN algorithm can infer that some observed nodes have some connections with some unobservable nodes. However, it is hard to know which unobservable nodes are.

We also do experiments on three empirical social networks, where the percent of missing nodes is set to be 10%. Table II includes the metrics to evaluate the structural accuracy and the state recovery accuracy. GIN achieves over 80% accuracy, which is less than in the synthetic network. One of the possible reasons is that the real social network is denser than the synthetic network, which increases the uncertainty of state transition compared to a sparse network. Our model obtains

about 80% accuracy in extrapolating the state values, which means that our model learned the dynamics of the nodes from discrete dynamics. Both in the inference of missing structure and known nodes' states, our model has achieved high inference accuracy on the empirical social network.

Generally speaking, inference accuracy should decrease with the increase of the proportion of unobservable nodes in the network completion problem. Thus, we investigate how fast the accuracy decrease with the missing proportion increase from 10% to 90%. Figure 5 shows the effect of the accuracy of network inference on the proportion of observed nodes. It can be seen that the AUC value decreases in a linearly way approximately. That means, each time the percentage of observed nodes increases by 10%, the AUC decreases by 0.05.

Through the above experiments, we found that GIN can infer network structure information with a higher accuracy rate. In addition, the effect of the inference is affected by the observed information. When the ratio of the observed information is less than 50%, it is difficult to infer the complete dynamic system accurately.

4. Network Completion without structural information

In the previous task, partial network structures can be known. However, in some real cases, we can only know the time series of the observed nodes. Therefore, we test the task of network completion without structural information in this experiment. To finish this task, we first use GIN to reconstruct the observable nodes and then make the completion of the whole network according to the reconstructed network. We outline our results by comparing the network completion task with or without network structure between observed nodes in Figure 6. It can be seen that the performance of task 2 is significantly better than that of task 1. However, the performance of task 1 is also very good, and the AUC is above 80%.

Further, we show the AUCs of network inference for different parts as shown in Table III. It can be seen that on different

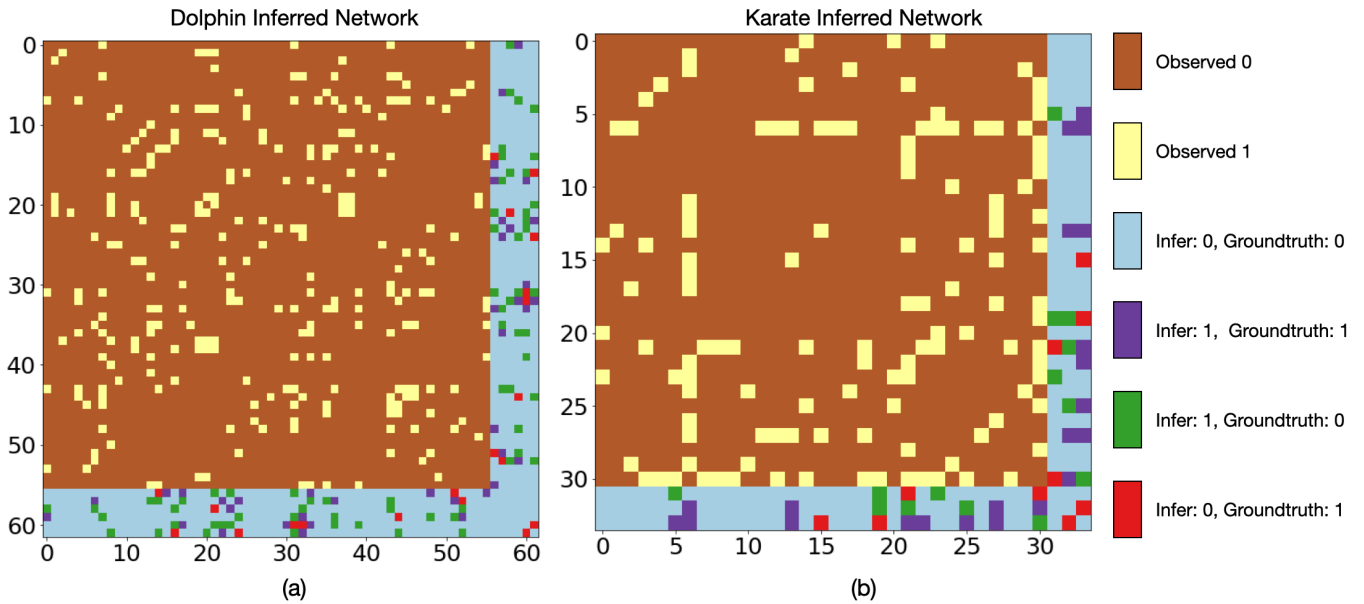


FIG. 4. Contrast matrices of the adjacency matrices between the inference and the ground truth for Dolphin network (a), and the Karate network (b). The invert L-shaped part in the figure has four colors, among which blue is the True Positive element, purple is the True Negative element, green is False True element, and red is the False Negative element.

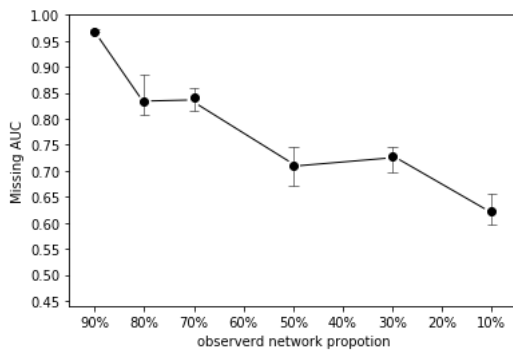


FIG. 5. AUC of unobservable network decreases with the proportion of observed nodes.

social networks, the AUC of the whole network has reached above 0.9. While, the AUCs of the unobservable part are 0.6, 0.6, and 0.8 on Karate, Email, and Dolphin networks, respectively. Reconstruction AUC represents the accuracy of the network reconstruction task for the links between observable nodes. We also compare the inferred network and the ground truth on a set of representative statistical indicators, and the results are similar as shown in the table IV.

In summary, our framework can also perform certain completion work when structural information is missing.

5. Network Reconstruction

On the task of network reconstruction, we compare with the state-of-the-art methods such as neural relational inference

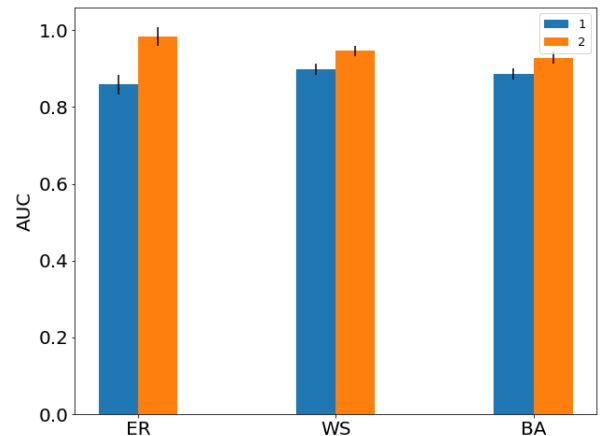


FIG. 6. Comparison of the performance of GIN under the network completion task with (1) or without (2) network structure information.

model (NRI)⁴². and the algorithm for revealing network interactions (ARNI)⁵⁵. In addition, we also compared with two traditional methods, namely mutual information method⁵⁶ and partial correlation method⁵⁷.

- **NRI**(Neural Relational Inference Model) applies a variational auto-encoder method to learn the underlying interaction graph and the complex system dynamics from the observational dynamical data. We ran the NRI Model by using the settings which are consistent with the original paper in⁴².
- **ARNI**(Algorithm for Revealing Network Interactions)

TABLE III. Network completion without structural information

Dynamics	Network	Nodes-Missing nodes	Missing AUC	Reconstruction AUC	All AUC
Discrete	Karate	34-3	0.7602	0.9930	0.9524
	Dolphins	62-6	0.8237	0.9989	0.9766
	Email-partial	143-14	0.5899	0.9819	0.9231
	ER	100-10	0.8923	0.9908	0.9850
	WS	100-10	0.8622	0.9883	0.9957
	BA	100-10	0.9189	0.9956	0.9875
Binary	ER	100-10	0.8585	0.9931	0.9717
	WS	100-10	0.8979	0.9943	0.9808
	BA	100-10	0.8863	0.9795	0.9776

TABLE IV. Comparison of the statistical properties for Dolphin network

Statistical characteristics	GIN	Real
Average Degree	5.067	5.129
Graph Distance	3.26	3.357
Graph Density	0.086	0.084
Clustering Coefficient	0.294	0.303

is a state-of-the-art method of network reconstruction task by regressing the gradient information of node state with the state in the previous time step⁵⁵.

- **Pcorr & MI** (Partial Correlation & Mutual Information) are all measures of correlation between node states. Partial correlation refers to the process of removing the influence of the third variable when two variables are simultaneously related to the third variable and only analyzing the degree of correlation between the other two variables. Mutual information (MI) is an information theoretic measure of the correlation of two variables. On network reconstruction, the two methods can be used to measure the similarity between the time series of two nodes. The less the similarity, the greater the probability that the two nodes connect.

In Table V, we show the performances of our model on the network reconstruction task. AUCs of GIN can reach above 99% on WS small world networks with different sizes. However, ARNI and NRI models can work on small networks with continuous dynamics. Our Framework can also handle large networks with more than a thousand of nodes while maintaining performance.

6. Utility analysis of seed graph matching algorithm

In order to analyze the effectiveness of our proposed evaluation algorithm, we conducted ablation experiments on the existing model by deleting the seed graph matching algorithm module, and the inferred adjacency matrix is directly compared with the real adjacency matrix. In addition, we also show the accuracy of matching by SGM between a randomly generated adjacency matrix and the real adjacency matrix.

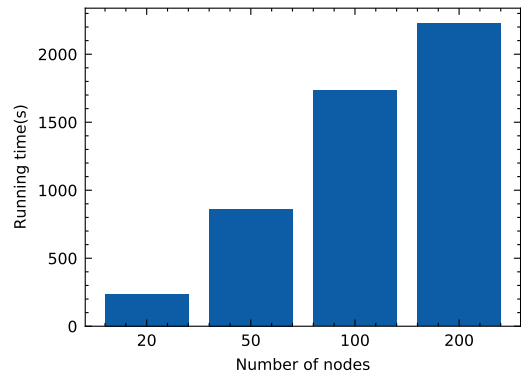


FIG. 7. The running time of GIN increases with the network size and the number of missing nodes.

We conducted experiments on a synthetic network with 100 nodes. The specific results are shown in table VI. The column "AUC (GIN without SGM)" shows the AUC value inferred from the adjacency matrix of the missing network structure without the Seeded Graph Matching module (Seeded Graph Matching, referred to as SGM), and uses SGM to match the randomly generated adjacency matrix. It is clear that GIN with SGM can get highest value of AUC on network completion. The result reflects the Network Completion -The optimal effect can be achieved only when GGN is coupled with the SGM evaluation algorithm, which shows the effectiveness of the SGM algorithm.

7. Computational complexity analysis

Training the the neural network requires time, and the time complexity increases with network size. To test how the time complexity depends on network size, we conduct experiments on a GPU of Tesla V100(16G) on WS small-world networks with 20,50,100, 200,300 nodes. And on the networks, 10% of nodes are unknown.

Fig 7 shows the curve of time complexity. Training the model requires an hour when the network size is not exceed 200. However, the time complexity increases dramatically on a network with 300 nodes due to the increase of the requirement on time series data, and it takes almost five hours.

TABLE V. Network reconstruction performance on different networks and dynamics

Dynamics	Nodes	GIN		MI	PCorr	ARNI		NRI	
		AUC	ACC/MSE	AUC	AUC	AUC	ACC/MSE	AUC	ACC/MSE
Binary	WS-10	1	0.9463	0.525	-	-	-	0.5037	0.9062
	WS-100	0.9961	0.7914	0.508	-	-	-	-	-
	WS-1000	0.9996	0.6623	0.547	-	-	-	-	-
	Dorm-217	0.6918	0.9999	0.5219	-	-	-	-	-
	Blog-1224	0.6366	0.9715	0.4995	-	-	-	-	-
Discrete	WS-10	1	3.31E-04	0.6875	0.785	1	2.35E-09	0.9997	8.40E-08
	WS-100	0.9987	1.48E-06	0.571	0.613	-	-	-	-
	WS-1000	0.9995	2.92E-06	0.567	-	-	-	-	-

TABLE VI. Comparison table of the effect of SGM algorithm module in GIN model

Network	AUC GIN with SGM	AUC GIN without SGM	Random AUC with SGM
ER(100-10)	0.9839±0.008	0.7662±0.062	0.6323±0.004
BA (100-10)	0.9463±0.032	0.8540±0.019	0.6885±0.008
WS(100-10)	0.9282±0.025	0.6268±0.024	0.6519±0.011

IV. DISCUSSION

In the paper, we discuss the problem of network inference with few unobservable nodes, and we propose a universal framework to solve the problem. First, we formulate the network inference problem based on time series data under an optimization framework. Second, we design GIN model by integrating three modules: a Gumbel-softmax based network generator, a graph network based dynamics learner, and an initial state generator. Third, we apply GIN on two vastly different types of time series data. We then reported the performances of GIN framework on three different network inference tasks, and GIN can work while on both network inference and initial state inference.

The benefits of our framework include its lightweight design, high accuracy, and universality to different network structures and dynamics. By using the Gumbel-softmax based network generator and the initial state generator, we can simply set the unknown elements to be learnable parameters, the design can be generalized in more cases. We test our framework on three different kinds of network inference tasks, and it can achieve relatively good results on all the tasks. And the results are robust and universal for different network structures and dynamics.

However, there are still many aspects that can be improved in our current work. For example, our model have a high accuracy in networks where the missing percent is less than 10%. With the missing percent increase, the number of unobserved nodes that have more than one degree of separation from an observed node will increases significantly which results in a major loss in the performance of the GIN model. The performance of the initial state generator can be improved on a larger space of node state. Besides, the amount of data needed for network inference is relatively large, the reduction for data requirement should be solved in the future. Furthermore, we hope that the accuracy of network completion can be further improved by increasing the performance of the initial state

generator.

In future works, we hope to combine the information of node states information and network structural information to infer unknown network structures. The dynamics learner also can be generalized to non-Markovian dynamical processes.

ACKNOWLEDGMENTS

The research is supported by the National Natural Science Foundation of China(NSFC) under the grant numbers 61673070.

DATA AVAILABILITY STATEMENT

The data that support the findings of this study are available from the corresponding author upon reasonable request.

APPENDICES

A. Data number on different task

As shown in table VII, we summarize the details about experimental data.

B. Convergence of the algorithm

Figure 8 shows the convergence scale with increasing network sizes. As the size of nodes increases, loss (MAE) is always maintained at a relatively stable level, which proves that the GIN algorithm has good convergence.

TABLE VII. Data on different task

Task	dynamics	Network	Number of initial states s	Number of datasets S
Network Completion Task	CMN	100-node synthetic	50	2.5k
		300-node synthetic	400	20k
	Voter	100-node WS/BA	100	5k
		100-node ER	300	15k
		300-node synthetic	300	15k
		Karate	20	1k
		Dolphin	200	10k
Email-partial	300	15k		
Network Reconstruction Task	CMN	10-node	240	12k
		100-node	200	10k
		1000-node	1200	60k
	Voter	10-node	400	20k
		100-node	200	10k
		1000-node	2000	100k
		Dorm	1200	60k
Blog	3200	160k		

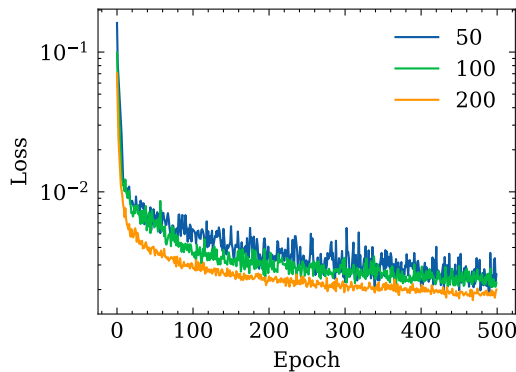


FIG. 8. The convergence scale with increasing network size. As the training epoch increased, loss(MAE) with logarithm shows a clear downward trend. At different node scales, the models converge as the training epoch increased.

REFERENCES

- ¹C. Tran, W.-Y. Shin, and A. Spitz, “Community detection in partially observable social networks,” arXiv preprint arXiv:1801.00132 (2017).
- ²S. Awate and R. Mudambi, “On the geography of emerging industry technological networks: The breadth and depth of patented innovations,” *Journal of Economic Geography* **18**, 391–419 (2018).
- ³K.-C. Chen, M. Chiang, and H. V. Poor, “From technological networks to social networks,” *IEEE Journal on Selected Areas in Communications* **31**, 548–572 (2013).
- ⁴S. Pastore, L. Ponta, and S. Cincotti, “Heterogeneous information-based artificial stock market,” *New Journal of Physics* **12**, 053035 (2010).
- ⁵V. Sood and S. Redner, “Voter model on heterogeneous graphs,” *Physical review letters* **94**, 178701 (2005).
- ⁶A. Lu, C. Sun, and Y. Liu, “The impact of community structure on the convergence time of opinion dynamics,” *Discrete Dynamics in Nature and Society* **2017** (2017).
- ⁷D. Centola, “The spread of behavior in an online social network experiment,” *science* **329**, 1194–1197 (2010).
- ⁸W. Kliabi and A. Martel, “Scenario-based supply chain network risk modeling,” *European Journal of Operational Research* **223**, 644–658 (2012).
- ⁹G. Cimini, T. Squartini, D. Garlaschelli, and A. Gabrielli, “Systemic risk analysis on reconstructed economic and financial networks,” *Scientific reports* **5**, 15758 (2015).
- ¹⁰M. E. Newman, “Scientific collaboration networks. ii. shortest paths, weighted networks, and centrality,” *Physical review E* **64**, 016132 (2001).
- ¹¹S. X. Zeng, X. M. Xie, and C. M. Tam, “Relationship between cooperation networks and innovation performance of smes,” *Technovation* **30**, 181–194 (2010).
- ¹²N. A. Ghani, S. Hamid, I. A. T. Hashem, and E. Ahmed, “Social media big data analytics: A survey,” *Computers in Human Behavior* **101**, 417–428 (2019).
- ¹³G. Kossinets, “Effects of missing data in social networks,” *Social networks* **28**, 247–268 (2006).
- ¹⁴K. Anand, I. van Lelyveld, Á. Banai, S. Friedrich, R. Garratt, G. Hałaj, J. Figue, I. Hansen, S. M. Jaramillo, H. Lee, *et al.*, “The missing links: A global study on uncovering financial network structures from partial data,” *Journal of Financial Stability* **35**, 107–119 (2018).
- ¹⁵T. Squartini, G. Caldarelli, G. Cimini, A. Gabrielli, and D. Garlaschelli, “Reconstruction methods for networks: the case of economic and financial systems,” *Physics Reports* (2018).
- ¹⁶R. Guimerà and M. Sales-Pardo, “Missing and spurious interactions and the reconstruction of complex networks,” *Proceedings of the National Academy of Sciences* **106**, 22073–22078 (2009).
- ¹⁷J. Kunegis and A. Lommatzsch, “Learning spectral graph transformations for link prediction,” in *Proceedings of the 26th Annual International Conference on Machine Learning* (ACM, 2009) pp. 561–568.
- ¹⁸L. Lü, C.-H. Jin, and T. Zhou, “Similarity index based on local paths for link prediction of complex networks,” *Physical Review E* **80**, 046122 (2009).
- ¹⁹D. Wang, D. Pedreschi, C. Song, F. Giannotti, and A.-L. Barabasi, “Human mobility, social ties, and link prediction,” in *Proceedings of the 17th ACM SIGKDD international conference on Knowledge discovery and data mining* (Acm, 2011) pp. 1100–1108.
- ²⁰M. Zhang and Y. Chen, “Link prediction based on graph neural networks,” in *Advances in Neural Information Processing Systems* (2018) pp. 5165–5175.
- ²¹C. Tran, W.-Y. Shin, A. Spitz, and M. Gertz, “Deepnc: Deep generative network completion,” arXiv preprint arXiv:1907.07381 (2019).
- ²²R. Guthke, U. Möller, M. Hoffmann, F. Thies, and S. Töpfer, “Dynamic network reconstruction from gene expression data applied to immune response during bacterial infection,” *Bioinformatics* **21**, 1626–1634 (2004).
- ²³M. Kim and J. Leskovec, “The network completion problem: Inferring missing nodes and edges in networks,” in *Proceedings of the 2011 SIAM International Conference on Data Mining* (SIAM, 2011) pp. 47–58.
- ²⁴Y. Xue and P. Bogdan, “Reconstructing missing complex networks against adversarial interventions,” *Nature communications* **10**, 1738 (2019).

- ²⁵Z. Wu, S. Pan, F. Chen, G. Long, C. Zhang, and P. S. Yu, “A comprehensive survey on graph neural networks,” arXiv preprint arXiv:1901.00596 (2019).
- ²⁶L. Du, Y. Wang, G. Song, Z. Lu, and J. Wang, “Dynamic network embedding: An extended approach for skip-gram based network embedding,” in *IJCAI* (2018) pp. 2086–2092.
- ²⁷L. Du, Z. Lu, Y. Wang, G. Song, Y. Wang, and W. Chen, “Galaxy network embedding: A hierarchical community structure preserving approach,” in *IJCAI* (2018) pp. 2079–2085.
- ²⁸J. Leskovec, D. Chakrabarti, J. Kleinberg, C. Faloutsos, and Z. Ghahramani, “Kronecker graphs: An approach to modeling networks,” *Journal of Machine Learning Research* **11**, 985–1042 (2010).
- ²⁹D. Xu, C. Ruan, K. Motwani, E. Korpeoglu, S. Kumar, and K. Achan, “Generative graph convolutional network for growing graphs,” in *ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)* (IEEE, 2019) pp. 3167–3171.
- ³⁰F. Geier, J. Timmer, and C. Fleck, “Reconstructing gene-regulatory networks from time series, knock-out data, and prior knowledge,” *BMC systems biology* **1**, 11 (2007).
- ³¹R. S. Krajec and A. G. Gounares, “Highlighting of time series data on force directed graph,” (2016), uS Patent 9,323,863.
- ³²W. Lin, N. Hubacher, and M. E. Khan, “Variational message passing with structured inference networks,” arXiv preprint arXiv:1803.05589 (2018).
- ³³A. Brovelli, M. Ding, A. Ledberg, Y. Chen, R. Nakamura, and S. L. Bressler, “Beta oscillations in a large-scale sensorimotor cortical network: directional influences revealed by granger causality,” *Proceedings of the National Academy of Sciences* **101**, 9849–9854 (2004).
- ³⁴C. J. Quinn, T. P. Coleman, N. Kiyavash, and N. G. Hatsopoulos, “Estimating the directed information to infer causal relationships in ensemble neural spike train recordings,” *Journal of computational neuroscience* **30**, 17–44 (2011).
- ³⁵J. M. Stuart, E. Segal, D. Koller, and S. K. Kim, “A gene-coexpression network for global discovery of conserved genetic modules,” *science* **302**, 249–255 (2003).
- ³⁶V. M. Eguiluz, D. R. Chialvo, G. A. Cecchi, M. Baliki, and A. V. Apkarian, “Scale-free brain functional networks,” *Physical review letters* **94**, 018102 (2005).
- ³⁷B. Barzel and A.-L. Barabási, “Network link prediction by global silencing of indirect correlations,” *Nature biotechnology* **31**, 720 (2013).
- ³⁸M. Timme, “Revealing network connectivity from response dynamics,” *Physical review letters* **98**, 224101 (2007).
- ³⁹W.-X. Wang, R. Yang, Y.-C. Lai, V. Kovanis, and C. Grebogi, “Predicting catastrophes in nonlinear dynamical systems by compressive sensing,” *Physical review letters* **106**, 154101 (2011).
- ⁴⁰W.-X. Wang, Y.-C. Lai, C. Grebogi, and J. Ye, “Network reconstruction based on evolutionary-game data via compressive sensing,” *Physical Review X* **1**, 021021 (2011).
- ⁴¹Z. Shen, W.-X. Wang, Y. Fan, Z. Di, and Y.-C. Lai, “Reconstructing propagation networks with natural diversity and identifying hidden sources,” *Nature communications* **5**, 4323 (2014).
- ⁴²T. Kipf, E. Fetaya, K.-C. Wang, M. Welling, and R. Zemel, “Neural relational inference for interacting systems,” arXiv preprint arXiv:1802.04687 (2018).
- ⁴³Z. Zhang, Y. Zhao, J. Liu, S. Wang, R. Tao, R. Xin, and J. Zhang, “A general deep learning framework for network reconstruction and dynamics learning,” *Applied Network Science* **4**, 1–17 (2019).
- ⁴⁴I. Ayed, E. de Bézenac, A. Pajot, J. Brajard, and P. Gallinari, “Learning dynamical systems from partial observations,” arXiv preprint arXiv:1902.11136 (2019).
- ⁴⁵L. Franceschi, M. Niepert, M. Pontil, and X. He, “Learning discrete structures for graph neural networks,” in *International conference on machine learning* (PMLR, 2019) pp. 1972–1982.
- ⁴⁶R. Wang, S. Mou, X. Wang, W. Xiao, Q. Ju, C. Shi, and X. Xie, “Graph structure estimation neural networks,” (2021).
- ⁴⁷Y. Zhang, Y. Guo, Z. Zhang, M. Chen, S. Wang, and J. Zhang, “Automated discovery of interactions and dynamics for large networked dynamical systems,” arXiv preprint arXiv:2101.00179 (2021).
- ⁴⁸D. E. Fishkind, S. Adali, H. G. Patsolic, L. Meng, D. Singh, V. Lyzinski, and C. E. Priebe, “Seeded graph matching,” *Pattern Recognition* **87**, 203–215 (2019).
- ⁴⁹B. Bollobás and B. Béla, *Random graphs*, 73 (Cambridge university press, 2001).
- ⁵⁰D. J. Watts and S. H. Strogatz, “Collective dynamics of ‘small-world’ networks,” *nature* **393**, 440–442 (1998).
- ⁵¹R. Albert and A.-L. Barabási, “Statistical mechanics of complex networks,” *Reviews of modern physics* **74**, 47 (2002).
- ⁵²R. A. Holley, T. M. Liggett, *et al.*, “Ergodic theorems for weakly interacting infinite systems and the voter model,” *The annals of probability* **3**, 643–663 (1975).
- ⁵³Y. Wang, G. Xiao, and J. Liu, “Dynamics of competing ideas in complex social systems,” *New Journal of Physics* **14**, 013015 (2012).
- ⁵⁴K. Kaneko, “Overview of coupled map lattices,” *Chaos: An Interdisciplinary Journal of Nonlinear Science* **2**, 279–282 (1992).
- ⁵⁵J. Casadiego, M. Nitzan, S. Hallerberg, and M. Timme, “Model-free inference of direct network interactions from nonlinear collective dynamics,” *Nature communications* **8**, 1–10 (2017).
- ⁵⁶J. F. Donges, Y. Zou, N. Marwan, and J. Kurths, “The backbone of the climate network,” *EPL (Europhysics Letters)* **87**, 48007 (2009).
- ⁵⁷S. McCabe, L. Torres, T. LaRock, S. A. Haque, C.-H. Yang, H. Hartle, and B. Klein, “netrd: A library for network reconstruction and graph distances,” arXiv preprint arXiv:2010.16019 (2020).