Link Prediction with Spatial and Temporal Consistency in Dynamic Networks

Wenchao Yu[1], **Wei Cheng**[2*], **Charu C. Aggarwal**[3], **Haifeng Chen**[2], and **Wei Wang**[1*]
[1]Department of Computer Science, University of California Los Angeles
[2]NEC Laboratories America, Inc.
[3]IBM T.J. Watson Research Center
{wenchaoyu, weiwang}@cs.ucla.edu, {weicheng, haifeng}@nec-labs.com, charu@us.ibm.com

Abstract

Dynamic networks are ubiquitous. Link prediction in dynamic networks has attracted tremendous research interests. Many models have been developed to predict links that may emerge in the immediate future from the past evolution of the networks. There are two key factors: 1) a node is more likely to form a link in the near future with another node within its close proximity, rather than with a random node; 2) a dynamic network usually evolves smoothly. Existing approaches seldom unify these two factors to strive for the spatial and temporal consistency in a dynamic network. To address this limitation, in this paper, we propose a link prediction model with spatial and temporal consistency (LIST), to predict links in a sequence of networks over time. LIST characterizes the network dynamics as a function of time, which integrates the spatial topology of network at each timestamp and the temporal network evolution. Comparing to existing approaches, LIST has two advantages: 1) LIST uses a generic model to express the network structure as a function of time, which makes it also suitable for a wide variety of temporal network analysis problems beyond the focus of this paper; 2) by retaining the spatial and temporal consistency, LIST yields better prediction performance. Extensive experiments on four real datasets demonstrate the effectiveness of the LIST model.

1 Introduction

Evolutionary network analysis [Aggarwal and Subbian, 2014] has become increasingly important in recent years. One of the major tasks is temporal link prediction which is to predict the future network structure based on a sequence of observed networks. Formally, in this paper, the problem of temporal link prediction is defined as: given a sequence of networks from timestamps 1 through T, the task is to predict the link weights at timestamp $T + \alpha$, where $\alpha \geq 1$. Note that a special case of this definition is to predict whether a new link will emerge or not (when we restrict the link weight to be either 0 or 1).

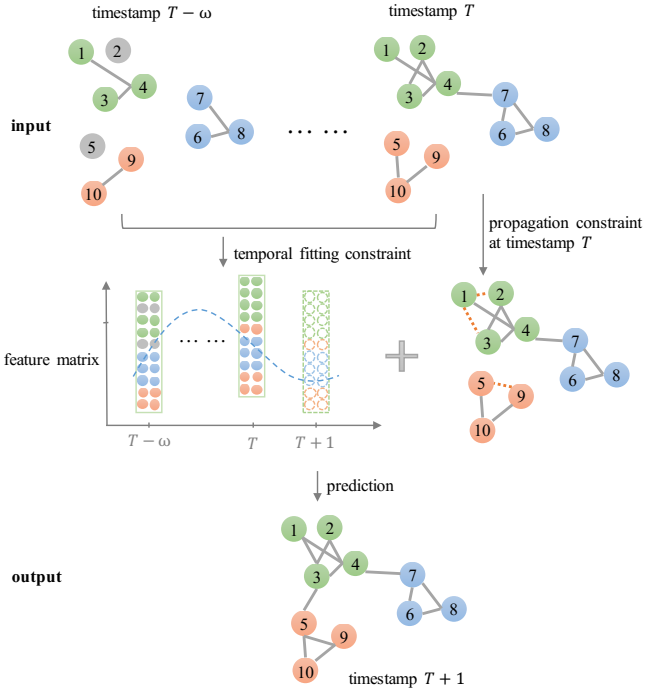

Figure 1: Illustration of the LIST model.

Extensive research efforts have been devoted to the temporal link prediction task including nonparametric [Sarkar *et al.*, 2012] and parametric [Tylenda *et al.*, 2009; Oyama *et al.*, 2011] models. The key factors to link prediction task in temporal networks are spatial and temporal consistencies, which mean: 1) a node has higher probability to form a link with a nearby node than with a remote node in the near future; 2) a network usually evolves smoothly over time. The first factor is akin to that in static network link prediction [Lü and Zhou, 2011; Hasan and Zaki, 2011]. It encodes the *local* network propagation constraints from the network topology at each timestamp. The second one *globally* enforces the *smoothness* of network evolution over time [Aggarwal and Subbian, 2014; Yu *et al.*, 2017].

Existing approaches, however, seldom unify these two fac-

*Corresponding authors.

tors to strive for the spatial and temporal consistency in the dynamic network. Their prediction performance thus degrades. Moreover, most link prediction algorithms focus on the very next timestamp (i.e., $\alpha = 1$), and are unable to predict when $\alpha > 1$. To address the limitation, we propose LIST, a model for link prediction with spatial and temporal consistency. We leverage the time-dependent matrix factorization technique, which has shown to be a powerful means to characterizing dynamic structural data [Koren, 2010; Yu *et al.*, 2017], to decompose the network adjacency matrices into time-dependent matrices that capture the features of vertices in the dynamic networks. At the same time, we introduce the network propagation constraint [Zhou *et al.*, 2003; Kashima *et al.*, 2009; Cheng *et al.*, 2016] which ensures vertices to be within close proximity to their neighbors in the hidden feature space to be learned by the time-dependent matrix factorization. As depicted in Figure 1, we learn the feature vector of each vertex by simultaneously optimizing the temporal fitting constraint and network propagation constraint. The temporal fitting constraint can be expressed as time-dependent matrix factorization with network adjacency matrix, while the propagation constraint preserves similarities between the connected pairs of vertices in the feature space. The learned feature matrices are parameterized with time and can be used to reconstruct the network structure at any given timestamp t. This allows us to make far more general predictions. Additionally, this feature matrix can be viewed as a complete profile of the network dynamics over time, which may also find its utility in other application settings of evolutionary network analysis.

Our contributions are as follows:

- *Spatial and temporal consistency in link prediction*: We propose a novel temporal link prediction model in dynamic networks which simultaneously consider the network structure at each timestamp and the evolutionary pattern across all timestamps. We leverage network propagation constraint which ensures that the connected vertices will have similar feature vectors. This "locality-preserving" property captures the spatial structure of networks at each timestamp. The feature vector of each vertex can be learned via time-dependent matrix factorization across all recent timestamps, which reveals the temporal structure of networks.
- *Computational speed-up*: We develop an efficient algorithm to learn the LIST model which reduces the time complexity from $O(n^3)$ to $O(mn + n^2)$, where n is the number of vertices in the dynamic network, and m is the number of nonzero entries in the network adjacency matrix. This makes the LIST model applicable for large-scale datasets.
- *Empirical improvements over previous work*: We evaluate the LIST model on four real datasets, and show that LIST outperforms all competitors, which demonstrates the effectiveness of our model.

The rest of the paper is organized as follows. Section 2 introduces the link prediction model with spatial and temporal consistency. Section 3 presents the experimental results of link weight prediction with four real-world dynamic networks. The related work is described in Section 4, followed by the conclusions in Section 5.

2 The LIST Model

Let the observed sequence of temporal networks be $G(t) = (\mathcal{N}, \mathbf{A}(t))$, where $\mathcal{N}$ is a set of vertices, and $\mathbf{A}(t)$ is the adjacency matrix of the network at timestamp $t \in [1, T]$, which is defined as a function of time. We assume that the size of vertex set $|\mathcal{N}| = n$, therefore $\mathbf{A}(t) \in \mathbb{R}^{n \times n}$. The elements a_{ijt} of $\mathbf{A}(t)$ is the link weight between vertices i and j at timestamp t. Our goal is to predict the links at timestamp $T + \alpha$ given $\mathbf{A}(1), \mathbf{A}(2), ..., \mathbf{A}(T)$.

We adopt the label propagation principle [Zhou *et al.*, 2003; Cheng *et al.*, 2016] which states that two vertices similar to each other are likely to have the same label. We consider a practical assumption that *two vertices that are connected are likely to have similar features*. Under this assumption, each vertex adjusts its feature vector based on its neighbors. Suppose that the initial feature vector of vertex i is $\mathbf{v}_i(t)$ and the final state is $\mathbf{f}_i(t)$ at timestamp t. Then the propagation process from $\mathbf{v}_i(t)$ to $\mathbf{f}_i(t)$ can be modeled by the following optimization problem.

$$\min_{\mathbf{f}(t)} \quad \lambda \sum_{i,j} \mathbf{A}_{ij}(t) \| \frac{1}{\sqrt{\mathbf{D}_{ii}(t)}} \mathbf{f}_i(t) - \frac{1}{\sqrt{\mathbf{D}_{jj}(t)}} \mathbf{f}_j(t) \|_2^2 + (1-\lambda) \sum_i \|\mathbf{f}_i(t) - \mathbf{v}_i(t)\|^2 \quad (1)$$

where $\mathbf{D}(t) \in \mathbb{R}^{n \times n}$ is the degree matrix of $\mathbf{A}(t)$. $\lambda \in (0, 1)$ is the regularization weight. The first term is the smoothness constraint, which enforces the neighboring vertices to have similar feature vectors. The second term is the fitting constraint, which penalizes large deviation from the initial feature vectors. The analytical solution of Eq. (1) is:

$$\mathbf{f}_i(t) = (1-\lambda)(\mathbf{I} - \lambda \tilde{\mathbf{A}}(t))^{-1} \mathbf{v}_i(t) \quad (2)$$

where $\mathbf{I} \in \mathbb{R}^{n \times n}$ is the identity matrix. $\tilde{\mathbf{A}}(t)$ is the normalized version of $\mathbf{A}(t)$ which is defined as $\sqrt{\mathbf{D}(t)}\mathbf{A}(t)\sqrt{\mathbf{D}(t)}$. This explicit solution shows that the final feature vector is a transformation of the initial one based on the network structure at timestamp t.

Then how do we determine $\mathbf{f}_i(t)$? In this paper we leverage the time-dependent matrix factorization method which naturally expresses the evolving network by learning a low rank representation of the underlying adjacency matrix. Let's focus on undirected networks for now. In this case, the symmetric adjacency matrix $\mathbf{A}(t)$ can be reconstructed by the feature vectors $\{\mathbf{f}_i(t)\}_{i=1}^n$,

$$\mathbf{A}(t) = \mathbf{F}(t)\mathbf{F}(t)^\top \quad (3)$$

Here $\mathbf{F}(t) = [\mathbf{f}_1, \mathbf{f}_2, ..., \mathbf{f}_n] \in \mathbb{R}^{n \times k}$ is a time-depend feature matrix.

We then follow a standard approach to set up a least squares optimization problem so that $\mathbf{A}(t)$ and $\mathbf{F}(t)\mathbf{F}(t)^\top$ are as

close as possible. This can be achieved by minimizing the Euclidean distance between all entries in $\mathbf{A}(t)$ and $\mathbf{F}(t)\mathbf{F}(t)^\top$. Therefore, the optimization problem can be written as,

$$\min_{\mathbf{F}} \sum_{t=\max(1,T-\omega)}^{T} \frac{h(t)}{2} \|\mathbf{A}(t) - \mathbf{F}(t)\mathbf{F}(t)^\top\|_F^2 \quad (4)$$

Here, $h(t) = e^{-\theta(T-t)}$ is an exponential decay function with time t that regulates the importance of the current timestamp of the network with respect to the past timestamps. ω is the sliding window size which only takes the recent ω timestamps into consideration. This is pragmatic because there is no need to store all network timestamps.

As it can be seen from Eq.(4), in order to learn $\mathbf{F}(t)$, one has to regulate the time-dependent form of this matrix. Let $\mathbf{P}(t) = (1-\lambda)(\mathbf{I} - \lambda\tilde{\mathbf{A}}(t))^{-1}$, we have $\mathbf{F}(t) = \mathbf{P}(t)\mathbf{V}(t)$ based on Eq.(2). Then the optimization problem described by Eq.(4) has the following form,

$$\min \sum_{t=\max(1,T-\omega)}^{T} \frac{h(t)}{2} \|\mathbf{A}(t) - \mathbf{P}(t)\mathbf{V}(t)\mathbf{V}(t)^\top\mathbf{P}(t)^\top\|_F^2 \quad (5)$$

Here $\mathbf{V}(t) \in \mathbb{R}^{n\times k}$ is a time-dependent matrix including all initial states of vertex feature vectors. It characterizes the network dynamics by modeling the changes in the vertex feature space. The function $\mathbf{V}(t)$ can take on any canonical form, such as linear models, polynomial models etc. based on the specific tasks. We choose polynomial function for $\mathbf{V}(t)$ because we are trying to fit the network dynamics within a small sliding window of length ω [Montgomery *et al.*, 2015]. Thus $\mathbf{V}(t)$ can be represented as follows:

$$\mathbf{V}(t) = \mathbf{W}^{(0)} + \mathbf{W}^{(1)}t + ... + \mathbf{W}^{(d)}t^d = \sum_{i=0}^{d} \mathbf{W}^{(i)}t^i \quad (6)$$

Here $\{\mathbf{W}^{(i)}\}_{i=0}^{d} \in \mathbb{R}^{n\times k}$, $d \in \mathbb{N}_+$. $\mathbf{V}(t)$ is the simple linear function if $d = 1$.

The challenge of optimizing the objective function defined in Eq.(5) is that the network $\mathbf{A}(t)$ could be very large and sparse, and the optimization of a $O(n^2)$ objective function is often too computationally expensive. Generally, the existence of a link provides more information than the absence of a link which conveys far more noises. Therefore, Eq.(5) should be tuned up to focus on the nonzero entries in the adjacency matrix $\mathbf{A}(t)$. Let m be the number of nonzero entries in $\mathbf{A}(t)$. However we still need a sample of zero entries to properly train the model. In this paper, the sample size is set to be equal to the size of nonzero entries m. Let $S(t)$ be the sample indices at timestamp t such that $a_{ijt} = 0, \forall(i,j) \in S(t)$. Let $E(t)$ be the set of indices that need to be optimize in Eq.(5), then we have $E(t) = \{(i,j)|a_{ijt} > 0\} \cup S(t)$. Note that the size of $E(t)$ is much smaller than $O(n^2)$ because the networks are often very sparse in practice. Then the aforementioned objective function can be presented as follows,

$$\sum_{t=\max(1,T-\omega)}^{T} \frac{h(t)}{2} \sum_{(i,j)\in E(t)} (a_{ijt} - (\mathbf{P}(t)\mathbf{V}(t)\mathbf{V}(t)^\top\mathbf{P}(t)^\top)_{ij})^2 \quad (7)$$

For directed networks, $\mathbf{A}(t)$ can be decomposed into two matrices: a constant matrix $\mathbf{U}$ and a time-dependent matrix $\mathbf{V}(t)$, both are $n \times k$ matrices. In this case, the optimization problem comes to be,

$$\sum_{t=\max(1,T-\omega)}^{T} \frac{h(t)}{2} \sum_{(i,j)\in E(t)} (a_{ijt} - (\mathbf{U}\mathbf{V}(t)^\top\mathbf{P}(t)^\top)_{ij})^2 \quad (8)$$

Note that one can choose both $\mathbf{U}$ and $\mathbf{V}(t)$ to be time-dependent, but it will double the parameter space and increase the model complexity. Similar results can be achieved if one choose to make $\mathbf{U}$ time-dependent rather than $\mathbf{V}(t)$. In the empirical study section, we focus on the undirected networks.

2.1 Model Learning

In this section, we describe the algorithm to learn the LIST model. The aforementioned objective function depends on the number of links in the networks which is easy to compute. We also need to add weight-decay terms to reduce the variance of our model. Consider the objective function $J(\mathbf{W})$ for undirected networks with weight-decay terms,

$$\sum_{t=\max(1,T-\omega)}^{T} \frac{h(t)}{2} \|\mathbf{1}_{E(t)}(\mathbf{A}(t) - \mathbf{P}(t)\mathbf{V}(t)\mathbf{V}(t)^\top\mathbf{P}(t)^\top)\|_F^2 + \sum_{i=0}^{d} \frac{\beta_i}{2}\|\mathbf{W}^{(i)}\|_F^2 \quad (9)$$

where $\mathbf{1}_{E(t)}(\mathbf{M}) = \begin{cases} \mathbf{M}_{ij} & \text{if } (i,j) \in E(t) \\ 0 & \text{if } (i,j) \notin E(t) \end{cases}$, $\{\beta_i\}_{i=0}^{d}$ are the weights of the weight-decay terms. In order to infer the parameter $\mathbf{W}$, we leverage the symmetric matrix factorization technique [Kuang *et al.*, 2012] to compute the derivatives of Eq.(9). We introduce an "error term" $\psi(t)$ for each timestamp t of undirected networks as follows:

$$\psi(t) = \mathbf{1}_{E(t)}(\mathbf{A}(t) - \mathbf{P}(t)\mathbf{V}(t)\mathbf{V}(t)^\top\mathbf{P}(t)^\top) \quad (10)$$

Note that the error matrix in $\psi(t)$ has already projected to indices set defined by $E(t)$. Consider the derivative calculation of a simplified loss function $J = \frac{1}{2}\|\mathbf{A} - \mathbf{P}\mathbf{V}\mathbf{V}^\top\mathbf{P}^\top\|_F^2$ without timestamp t, then

$$\begin{aligned}\frac{\partial J}{\partial \mathbf{V}} &= -\mathbf{P}^\top\mathbf{A}\mathbf{P}\mathbf{V} - \mathbf{P}^\top\mathbf{A}^\top\mathbf{P}\mathbf{V} + 2\mathbf{P}^\top\mathbf{P}\mathbf{V}\mathbf{V}^\top\mathbf{P}^\top\mathbf{P}\mathbf{V} \\ &= -\mathbf{P}^\top((\mathbf{A} - \mathbf{P}\mathbf{V}\mathbf{V}^\top\mathbf{P}^\top) + (\mathbf{A} - \mathbf{P}\mathbf{V}\mathbf{V}^\top\mathbf{P}^\top)^\top)\mathbf{P}\mathbf{V}\end{aligned}$$

Here we extract the common factor $\mathbf{A} - \mathbf{P}\mathbf{V}\mathbf{V}^\top\mathbf{P}^\top$ for each term in J. The reason is that we can apply projection $\mathbf{1}_{E(t)}(\cdot)$ on this common factor. So that,

$$\begin{aligned}\frac{\partial J(\mathbf{W})}{\partial \mathbf{W}^{(i)}} &= \frac{\partial J(\mathbf{W})}{\partial \mathbf{V}(t)}\frac{\partial \mathbf{V}(t)}{\partial \mathbf{W}^{(i)}} \\ &= \sum_{t=\max(1,T-\omega)}^{T} h(t)\mathbf{P}(t)^\top\left(-\psi(t) - \psi(t)^\top\right)\mathbf{P}(t)\mathbf{V}(t)t^i \\ &+ \beta_i\mathbf{W}^{(i)}\end{aligned} \quad (11)$$

We now have the derivative over $\mathbf{W}$ needed to run gradient descent. The pseudocode for LIST model is presented in Algorithm 1.

Algorithm 1: Algorithm for LIST model

Input: temporal adjacency matrices $\{\mathbf{A}(t)\}_{t=\max(1,T-\omega)}^{T}$, the order d of $\mathbf{V}(t)$ and latent dimension k

Output: factor matrices $\{\mathbf{W}^{(i)}\}_{i=1}^{d}$ and the prediction $\mathbf{A}(T+1)$.

Set k, d and ω.
Randomly initialize $\{\mathbf{W}^{(i)}\}_{i=0}^{d}$.
while *not stopping criterion* **do**
  Compute "error term" $\psi(t)$ for each time stamp t.
  Compute partial derivatives $\frac{\partial J(\mathbf{W})}{\partial \mathbf{W}^{(i)}}$ using $\psi(t)$ by Eq.(11).
  Determine the step size λ by line search.
  foreach i *in* $\{1, ..., d\}$ **do**
    Update $\mathbf{W}^{(i)} = \mathbf{W}^{(i)} - \lambda \frac{\partial J(\mathbf{W})}{\partial \mathbf{W}^{(i)}}$.
  Compute prediction result $\mathbf{A}(T+1) = (\sum_{i=0}^{d} \mathbf{W}^{(i)}(T+1)^i)(\sum_{i=0}^{d} \mathbf{W}^{(i)}(T+1)^i)^\top$.

2.2 Computational Speed-up

Observe that the update of $\mathbf{W}^{(i)}$ in Algorithm 1 has to compute the inverse of an $n \times n$ matrix $\mathbf{I} - \lambda\tilde{\mathbf{A}}(t)$ which runs in time $O(n^3)$. In this section, an iterative method is used to approximate the matrix inverse calculation.

Theorem 1 *Given that the eigenvalues of* $\mathbf{Q} \in \mathbb{R}^{n\times n}$ *in* $[-1, 1]$, $0 < \gamma < 1$, *and the iteration number* B, *the inverse of matrix* $\mathbf{I} - \gamma\mathbf{Q}$ *can be approximated by summing up* $(\gamma\mathbf{Q})^b$ *across all iterations [Zhou* et al.*, 2003], which is*

$$(\mathbf{I} - \gamma\mathbf{Q})^{-1} = \lim_{B\to\infty} \sum_{b=1}^{B} (\gamma\mathbf{Q})^{b-1} \tag{12}$$

Given $\lambda \in (0, 1)$ and $\tilde{\mathbf{A}}(t)$ (normalized by degree matrix $\mathbf{D}(t)$), the approximate solution for $\mathbf{P}(t)$ is,

$$\mathbf{P}(t) = (1-\lambda)(\mathbf{I} - \lambda\tilde{\mathbf{A}}(t))^{-1} = (1-\lambda)\sum_{b=1}^{B} (\lambda\tilde{\mathbf{A}}(t))^{b-1} \tag{13}$$

Assume that the number of nonzero entries in the sparse matrix $\tilde{\mathbf{A}}(t)$ is m. The complexity upper-bound of sparse matrix multiplication is $O(mn)$ [Yuster and Zwick, 2005]. Therefore the above solution takes $O((b-1)mn)$ in time for each iteration, $b \in [1, B]$, which results in an overall complexity $O(B^2mn)$ for the calculation of $\mathbf{P}(t)$. If we cache the computed results in previously iteration, then the time complexity is reduced to $O(Bmn)$.

2.3 Complexity Analysis

We assume that the gradient-descent method in Algorithm 1 is implemented for M iterations, and the rank of the factorization is k. The bottleneck step is to update all parameters for $d+1$ matrices in each of these iterations. It can be seen from Eq.(11) that there are ω timestamps within each sliding window, and in each timestamp the complexity for matrix manipulation is $O(Bmn) + O(n^2k)$. Thus, the asymptotic running time is $O(M\omega(d+1)(Bmn+n^2k))$. In empirical study settings, M, ω, d, B and k are much smaller than n and m, therefore the time complexity is approximately $O(mn+n^2)$.

Table 1: Dynamic network data description

Dataset	#Vertex	#Edge	Max Weight	$\mathcal{T}$	#Link Distribution
Infectious	410	2,765	191	8	
UCI Msg	1,899	20,296	98	7	
Digg	30,398	86,404	25	14	
DBLP	315,159	743,709	159	34	

3 Empirical Study

The ability to predict link weights at a specific time is almost trivial using the LIST model. Once we obtain the factor matrices $\{\mathbf{W}^{(i)}\}_{i=1}^{d}$, the *predicted* structure of the network can be effectively reconstruct by $\mathbf{V}(T+\alpha)\mathbf{V}(T+\alpha)^\top$ for any $\alpha \geq 1$. Of course, as the value of α becomes larger and larger, one can expect the reconstruction to become increasingly challenging. In this section, we will compare with the baselines on single-timestamp link weight prediction ($\alpha = 1$), and show the advantages of the LIST model on multiple-timestamp link weight prediction ($\alpha > 1$).

3.1 Datasets and Baselines

Datasets: To verify the performance of the LIST model, we conduct experiments on four dynamic networks, namely *Infectious* [Isella *et al.*, 2011], *UCI Msg* [Opsahl and Panzarasa, 2009], *Digg*[1] and *DBLP*[2], as shown in Table 1. The vertices in these networks represent users, and edge weights represent the strength of the interactions such as number of messages and co-authorships, among them. Table 1 also shows the distribution of number of edges over time.

Baselines: For comparison, we consider the canonical link prediction method *Weighted Common Neighbors (WCN)* [Murata and Moriyasu, 2007; Zhao *et al.*, 2015], as well as recent algorithms *High-performance Link Prediction (HPLP)* [Lichtenwalter *et al.*, 2010], *CP Tensor Model (CPTM)* [Dunlavy *et al.*, 2011] and *Temporal Matrix Factorization (TMF)* [Yu *et al.*, 2017]. Note that the *HPLP* algorithm used here is a modified version that trains a regression model to predict the link weights. We analyze all algorithms by measuring the accuracy of link weight prediction based on root mean-squared error (RMSE).

3.2 Single-timestamp Link Weight Prediction

To compare the performance of single-timestamp link weight prediction ($\alpha = 1$), we utilize the network timestamps from $T-\omega$ to $T-1$ as the training set, and the T^{th} timestamp as the test set, $T \in [2, \mathcal{T}]$, where $\mathcal{T}$ is the total number of timestamps in each dataset. We set the order d of time-dependent matrix $\mathbf{V}(t)$ to 1. As reported in [Yu *et al.*, 2017], hight order of d may slightly improve the performance, but not always.

[1] http://konect.uni-koblenz.de/networks
[2] http://dblp.uni-trier.de/xml

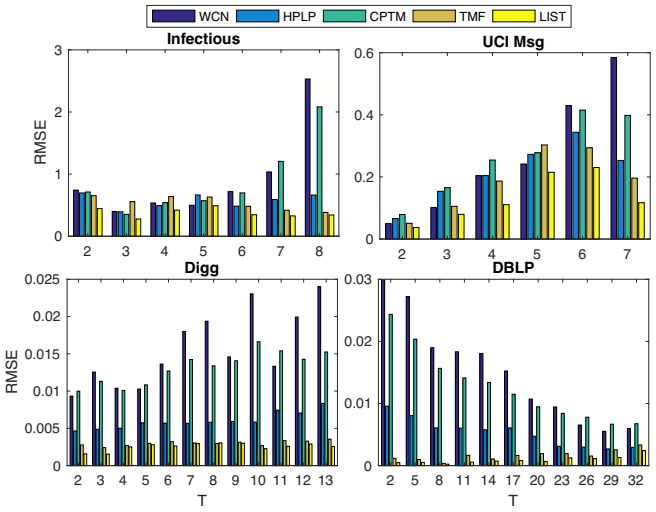

Figure 2: Prediction RMSE at timestamp T while training with previous ω timestamps start at $T - \omega$

Table 2: Average RMSE across all timestamps

Method	Infectious	UCI Msg	Digg	DBLP
WCN	0.9719±0.8300	0.2719±0.2064	0.0161±0.0048	0.0160±0.0104
HPLP	0.5883±0.6767	0.2702±0.2034	0.0064±0.0012	0.0056±0.0147
CPTM	0.8847±0.9367	0.2196±0.2469	0.0133±0.0136	0.0128±0.0099
TMF	0.5309±0.1185	0.1840±0.1311	0.0032±0.0011	0.0017±0.0028
LIST	**0.3824**±0.1114	**0.1345**±0.0930	**0.0026**±0.0005	**0.0009**±0.0009

The other parameter settings are as follows: iteration number $B = 100$ for the computation of $\mathbf{P}(t)$, latent dimension $k = 20$, exponential decay $\theta = 0.3$, sliding window size $\omega = 5$, propagation balancing weight $\lambda = 0.3$, regularizer weights $\beta_i = 0.01$. The maximum number of iterations of the LIST model is set to 200. We analyze the algorithms by measuring the prediction RMSE at different timestamps as shown in Figure 2.

We have several key observations from Figure 2. Firstly, the proposed LIST model outperforms all baselines, which are consistent across all four dynamic networks. It demonstrates the advantages of leveraging network propagation in predicting link weights, and shows that the LIST model can capture the underlying structure of the network evolution. We also notice that the LIST model achieves a higher accuracy margin against the TMF model at early timestamps. It is because there is no enough data to train the TMF model, but the LIST model makes full use of the network structure through the propagation constraint.

Table 2 displays the average prediction RMSE across all timestamps of each dataset. It is evident that the LIST model has a better average RMSE than all four competing methods. Note that the performance of the LIST model is about $18\times$ better than WCN on the *DBLP* dataset.

3.3 Multiple-timestamp Link Weight Prediction

Notably, the LIST model has the capability to effectively reconstruct the structure of the network at any given time. In this section, we run experiments to measure the prediction accuracy in predicting link weights for multiple timestamps. The task is to predict the link weights at timestamp $T + \alpha$, where $\alpha = \{1, 2, 3\}$ in the experiments. That is, we are using the LIST model to predict the next three timestamps. The rest of the parameters are set as Section 3.2. The multiple-timestamp prediction RMSE is presented in Figure 3.

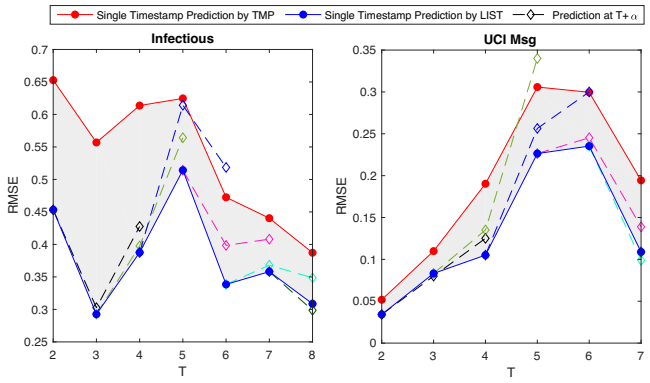

Figure 3: The dots on *dash lines* represent RMSE of multiple-timestamp link prediction at timestamp $T + \alpha$. The blue and red dots on *solid lines* represent the single-timestamp prediction RMSE of LIST and the baseline TMF, respectively.

Since we are predicting the link weight of the upcoming three timestamps, each short dash line has three dots (or fewer than three at the last two timestamps) which plot the RMSE of link prediction at timestamp $T+1$, $T+2$ and $T+3$. There are $\mathcal{T}-2$ dash lines in total for each dataset ($\mathcal{T}$ is the total number of timestamps). The dots on solid lines indicate the single-timestamp prediction RMSE of LIST and TMF, which are used as references. It can be seen that, even when $\alpha > 1$, the LIST prediction accuracy is still better than the baseline TMF most time. It is also worth mentioning that several predictions made by multiple-timestamp link prediction are better than single-timestamp predictions.

3.4 Parameter Analysis

Propagation Balancing Parameter λ: The parameter λ defined by Eq.(1) specifies the relative amount of the information from its neighbors (propagation constraint) and its initial feature vectors (fitting constraint). By varying the parameter λ, we want to examine the relative importance of network structure in single-timestamp link weight prediction task. The special case of $\lambda = 0$ indicates that no network structure information will be considered. We choose different values of λ which varies from 0 to 1 with step size 0.1, and compute the prediction RMSE at timestamp T on *Infections* and *UCI Msg* datasets. All the remaining parameter settings are the same as Section 3.2. The results are summarized in Figure 4. The RMSE curves of both datasets follow a broad "U" shape. This means that as λ increases from 0 to 1, the prediction accuracy increases till an optimal value, after which it starts to decline. It is evident that the propagation constraint can help improve the performance of link prediction in dynamic networks. We also observe that $\lambda \in [0.1, 0.4]$ gives the optimal prediction RMSE.

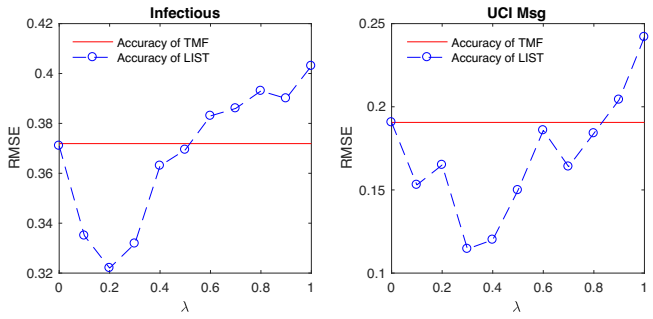

Figure 4: Prediction RMSE of *Infectious* and *UCI Msg* at timestamp T with different values of λ

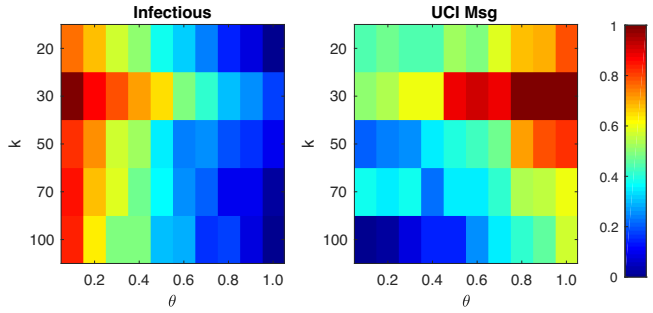

Figure 5: Prediction RMSE of *Infectious* and *UCI Msg* at timestamp T with different parameter pairs θ and k.

Exponential Decay θ and Factorization Rank k: In this section, we conduct the parameter analysis on θ and k. θ regulates the exponential decay function $h(t)$. A larger θ represents that less weights are assigned to the previous timestamps. k is the latent dimension of the feature matrix $\mathbf{F}(t)$. We choose different values of θ varies from 0 to 1 with interval 0.1, and k varies from 20 to 100. We show the results on datasets *Infectious* and *UCI Msg*. The prediction RMSE is presented as a heat-map in Figure 5. It depicts that θ has a significant impact on prediction accuracy, and it is sensitive to the dataset. For the *Infectious* dataset, as θ increases, the proposed model has a better RMSE. This means a better prediction can be achieved when less weights are assigned to early timestamps. For the *UCI Msg* dataset, we observe a totally opposite trend. It also can be seen that RMSE improves with the increase of k. This is because larger k preserves more information when performs the matrix factorization. But from the complexity analysis in Section 2.3 we can see that a relatively small k takes less running time. Taking both running time and prediction accuracy into consideration, we choose a relative small k ranged from 20 to 100.

4 Related Work

Link prediction problem can be generally divided into two distinct categories: structural and temporal link prediction [Menon and Elkan, 2011]. The structural link prediction problem, which only considers a single network structure as the input, predicts the possible unobserved links within the same network [Liben-Nowell and Kleinberg, 2007; Lichtenwalter *et al.*, 2010]. Temporal link prediction models, on the other hand, analyze the evolution pattern of a sequence of networks over time [Sarkar *et al.*, 2012; Dunlavy *et al.*, 2011; Li *et al.*, 2014; Zhu *et al.*, 2016; Rahman and Al Hasan, 2016]. The probabilistic nonparametric link prediction model can be used to predict the linkage possibility of two nodes only based on the similarity of their local neighborhoods [Sarkar *et al.*, 2012]. One can also extend a local probabilistic model [Wang *et al.*, 2007] based on maximum entropy with temporal information of the past interactions [Tylenda *et al.*, 2009]. Other than the aforementioned statistical approaches, we can also leverage tensor decomposition technique to address the temporal link prediction problem effectively [Dunlavy *et al.*, 2011; Ermiş *et al.*, 2015]. The global network structure has also been considered in the temporal link prediction task [Gao *et al.*, 2011]. It aggregates weighted link matrices, and learns the model with graph regularization. Recently, some interest has been focused on the use of temporal matrix factorization technique for expressing dynamic networks [Yu *et al.*, 2017]. However, it only explores the network evolving pattern across the dynamic network, but ignores the network propagation within each single timestamp.

5 Conclusions

In this paper, we developed a novel link prediction model, LIST, for dynamic networks which simultaneously incorporates network propagation and temporal matrix factorization techniques. This is guaranteed by the joint minimization of the network propagation loss and the temporal network reconstruction error. The proposed model utilizes a user-defined sliding window to learn the parameters, thus supports streaming link prediction as well. Extensive experiments show that the LIST model outperforms the state-of-the-art techniques. One interesting aspect of this model is its ability to explicitly express the network as a function of time, which takes into account both local (spatial) network structure and globe evolving (temporal) patterns. Therefore it has the advantage of generality in addressing various temporal applications like temporal network compression and expanding community detection. We plan to further investigate the utility of LIST in these applications.

Acknowledgements

The work is partially supported by NIH U01HG008488, NIH R01GM115833, NIH U54GM114833, and NSF IIS-1313606. Research of the third author was sponsored by the Army Research Laboratory and was accomplished under Cooperative Agreement Number W911NF-09-2-0053. The views and conclusions contained in this document are those of the authors and should not be interpreted as representing the official policies, either expressed or implied, of the Army Research Laboratory or the U.S. Government. The U.S. Government is authorized to reproduce and distribute reprints for Government purposes notwithstanding any copyright notation here on. We thank the anonymous reviewers for their careful reading and insightful comments on our manuscript.

References

[Aggarwal and Subbian, 2014] Charu Aggarwal and Karthik Subbian. Evolutionary network analysis: A survey. *ACM Computing Surveys*, 47(1):10, 2014.

[Cheng *et al.*, 2016] Wei Cheng, Kai Zhang, Haifeng Chen, Guofei Jiang, Zhengzhang Chen, and Wei Wang. Ranking causal anomalies via temporal and dynamical analysis on vanishing correlations. In *KDD*, pages 805–814. ACM, 2016.

[Dunlavy *et al.*, 2011] Daniel M Dunlavy, Tamara G Kolda, and Evrim Acar. Temporal link prediction using matrix and tensor factorizations. *TKDE*, 5(2):10, 2011.

[Ermiş *et al.*, 2015] Beyza Ermiş, Evrim Acar, and A Taylan Cemgil. Link prediction in heterogeneous data via generalized coupled tensor factorization. *Data Mining and Knowledge Discovery*, 29(1):203–236, 2015.

[Gao *et al.*, 2011] Sheng Gao, Ludovic Denoyer, and Patrick Gallinari. Temporal link prediction by integrating content and structure information. In *CIKM*, pages 1169–1174. ACM, 2011.

[Hasan and Zaki, 2011] Mohammad Al Hasan and Mohammed J. Zaki. A survey of link prediction in social networks. In *Social Network Data Analytics*, pages 243–275. 2011.

[Isella *et al.*, 2011] Lorenzo Isella, Juliette Stehlé, Alain Barrat, Ciro Cattuto, Jean-François Pinton, and Wouter Van den Broeck. What's in a crowd? analysis of face-to-face behavioral networks. *Journal of theoretical biology*, 271(1):166–180, 2011.

[Kashima *et al.*, 2009] Hisashi Kashima, Tsuyoshi Kato, Yoshihiro Yamanishi, Masashi Sugiyama, and Koji Tsuda. Link propagation: A fast semi-supervised learning algorithm for link prediction. In *SDM*, pages 1100–1111. SIAM, 2009.

[Koren, 2010] Yehuda Koren. Collaborative filtering with temporal dynamics. *Communications of the ACM*, pages 89–97, 2010.

[Kuang *et al.*, 2012] Da Kuang, Haesun Park, and Chris HQ Ding. Symmetric nonnegative matrix factorization for graph clustering. In *SDM*, volume 12, pages 106–117. SIAM, 2012.

[Li *et al.*, 2014] Xiaoyi Li, Nan Du, Hui Li, Kang Li, Jing Gao, and Aidong Zhang. A deep learning approach to link prediction in dynamic networks. In *SDM*. SIAM, 2014.

[Liben-Nowell and Kleinberg, 2007] David Liben-Nowell and Jon Kleinberg. The link-prediction problem for social networks. *JASIST*, 58(7):1019–1031, 2007.

[Lichtenwalter *et al.*, 2010] Ryan N Lichtenwalter, Jake T Lussier, and Nitesh V Chawla. New perspectives and methods in link prediction. In *KDD*, pages 243–252. ACM, 2010.

[Lü and Zhou, 2011] Linyuan Lü and Tao Zhou. Link prediction in complex networks: A survey. *Physica A: Statistical Mechanics and its Applications*, 390(6):1150–1170, 2011.

[Menon and Elkan, 2011] Aditya Krishna Menon and Charles Elkan. Link prediction via matrix factorization. In *PKDD*, pages 437–452. Springer, 2011.

[Montgomery *et al.*, 2015] Douglas C Montgomery, Elizabeth A Peck, and G Geoffrey Vining. *Introduction to linear regression analysis*. John Wiley & Sons, 2015.

[Murata and Moriyasu, 2007] Tsuyoshi Murata and Sakiko Moriyasu. Link prediction of social networks based on weighted proximity measures. In *WI*, pages 85–88. IEEE, 2007.

[Opsahl and Panzarasa, 2009] Tore Opsahl and Pietro Panzarasa. Clustering in weighted networks. *Social networks*, pages 155–163, 2009.

[Oyama *et al.*, 2011] Satoshi Oyama, Kohei Hayashi, and Hisashi Kashima. Cross-temporal link prediction. In *ICDM*, pages 1188–1193. IEEE, 2011.

[Rahman and Al Hasan, 2016] Mahmudur Rahman and Mohammad Al Hasan. Link prediction in dynamic networks using graphlet. In *PKDD*, pages 394–409. Springer, 2016.

[Sarkar *et al.*, 2012] Purnamrita Sarkar, Deepayan Chakrabarti, and Michael I Jordan. Nonparametric link prediction in dynamic networks. In *ICML*, pages 1687–1694, 2012.

[Tylenda *et al.*, 2009] Tomasz Tylenda, Ralitsa Angelova, and Srikanta Bedathur. Towards time-aware link prediction in evolving social networks. In *Proceedings of the 3rd Workshop on Social Network Mining and Analysis*, page 9. ACM, 2009.

[Wang *et al.*, 2007] Chao Wang, Venu Satuluri, and Srinivasan Parthasarathy. Local probabilistic models for link prediction. In *ICDM*, pages 322–331. IEEE, 2007.

[Yu *et al.*, 2017] Wenchao Yu, Charu C Aggarwal, and Wei Wang. Temporally factorized network modeling for evolutionary network analysis. In *WSDM*, pages 455–464. ACM, 2017.

[Yuster and Zwick, 2005] Raphael Yuster and Uri Zwick. Fast sparse matrix multiplication. *ACM Transactions on Algorithms*, 1(1):2–13, 2005.

[Zhao *et al.*, 2015] Jing Zhao, Lili Miao, Jian Yang, Haiyang Fang, Qian-Ming Zhang, Min Nie, Petter Holme, and Tao Zhou. Prediction of links and weights in networks by reliable routes. *Scientific reports*, 5, 2015.

[Zhou *et al.*, 2003] Dengyong Zhou, Olivier Bousquet, Thomas Navin Lal, Jason Weston, and Bernhard Schölkopf. Learning with local and global consistency. In *NIPS*, volume 16, pages 321–328, 2003.

[Zhu *et al.*, 2016] Linhong Zhu, Dong Guo, Junming Yin, Greg Ver Steeg, and Aram Galstyan. Scalable temporal latent space inference for link prediction in dynamic social networks. *TKDE*, 28(10):2765–2777, 2016.