Effective Link Prediction with Topological and Temporal Information using Wavelet Neural Network Embedding

Xian Mo[1,3], Jun Pang[2], Zhiming Liu[1,3,*]

[1]College of Computer & Information Science, Southwest University, Chongqing, China
[2]Faculty of Science, Technology and Medicine & Interdisciplinary Centre for Security, Reliability and Trust, University of Luxembourg, Esch-sur-Alzette, Luxembourg
[3]Centre for Research and Innovation in Software Engineering, Southwest University, Chongqing, China
[*]Corresponding author: zhimingliu88@swu.edu.cn

Temporal networks are networks that edges evolve over time, hence link prediction in temporal networks aims at inferring new edges based on a sequence of network snapshots. In this paper, we proposed a graph wavelet neural network (TT-GWNN) framework using topological and temporal features for link prediction in temporal networks. To capture topological and temporal features, we developed a second-order weighted random walk sampling algorithm (SWRW). It combines network snapshots with both first-order and second-order weights into one weighted graph. Moreover, it incorporates a damping factor to assign greater weights to more recent snapshots. Next, we adopted graph wavelet neural networks (GWNN) to embed the vertices and used gated recurrent units (GRUs) for predicting new links. Extensive experiments demonstrated that TT-GWNN can effectively predict links on temporal networks.

Keywords: Temporal networks; link prediction; network embedding; topological and temporal features; graph wavelet neural networks; gated recurrent units

1. INTRODUCTION

Networks are often used to describe complex systems, where each node represents an entity and each edge represents the interaction between a pair of entities. The vast majority of the real-world network is not static but evolving, which can be modeled as temporal networks [1]. In recent years, temporal networks have been extensively studied and applied in many research fields, such as social networks [2], biological networks [3] and co-authorship networks [4]. Link prediction is an important analytical tool in temporal networks, which aims at inferring new links based on a sequence of network snapshots and can help us better understand network evolution [5].

Many methods for link prediction in temporal networks have been proposed in the literature, including [6, 7, 8, 9, 10, 11, 12]. However, most earlier studies [13] ignore the connection information among the consecutive network snapshots, thereby resulting in undesirable performance in link prediction. Network structure representation, including both topological structure features and temporal evolution features, is the key information for effective link prediction in temporal networks. The topological structure features represent the topology information of the network, while the temporal evolution features represent network topology evolving pattern from the current snapshot to the next snapshot. It is thus essential to use both topological and temporal features to comprehend complex behaviours of temporal networks. One common approach is to explore networks' topology information based on non-negative matrix factorisation [6, 11, 8, 9, 10]. However, real-life networks are often sparse and large, methods using matrix factorisation may cause high computational cost. In addition, these methods have limited ability in extracting the correlation of high dimensional features, as recently discussed in [14]. Different from the matrix factorisation-based methods, the DynamicTriad model [12] adopts the triadic closure process mechanism for the formation and evolution of temporal networks. However, it is inefficient in dealing with sparse networks, due to the fact that the closed triad is difficult to form from open triad in more

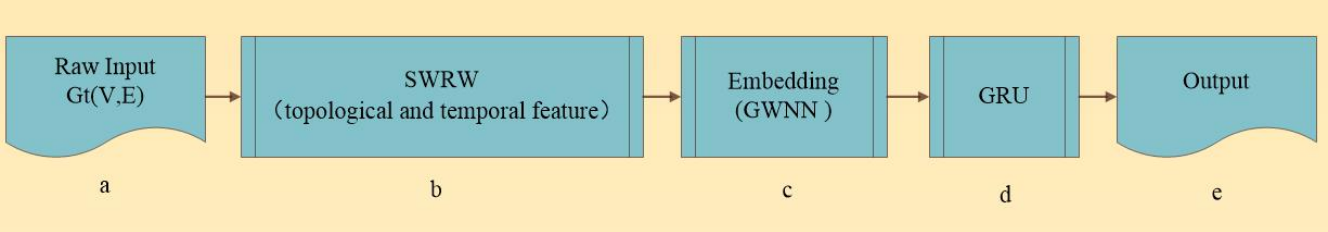

FIGURE 1. (a) Raw input: a temporal network G_t with a number of network snapshots. (b) A topological and temporal features sampling layer which samples features according to the first-order and second-order weights of each snapshot. (c) An embedding layer which maps each node in the network to its low dimensional representation. (d) A GRU layer which builds the model for link prediction. (e) The model output.

sparse networks over time (see [12] for the definition of closed triad and open triad). Learning node representations to effectively encode high-dimensional and non-Euclidean network information has become a challenging problem in temporal networks. However, the emergence of neural network techniques, especially deep learning techniques, brings new insights into this field. Some representation learning methods focus on static networks, such as DeepWalk [15], Node2vec [16], SDNE [17], and cannot obtain temporal features in temporal networks directly. By leveraging the temporal features of temporal networks, the conditional temporal restricted Boltzmann machine (ctRBM) [18] extends the structure of a standard RBM. However, it is a shallow model with limited ability to extract nonlinear features. The deep belief network (DBN) [19] explores the topological features of the static network at each timestamp, but it fails to concurrently capture the topological and temporal features from several network snapshots. Li et al. [7] developed a deep dynamic network embedding (DDNE) model to capture both topological and temporal features using gated recurrent units (GRUs). However, the input to the DDNE model is still the adjacent matrix of the networks, hence this still suffers from the problem of high computational cost. Recently, Chen et al. [20] developed an end-to-end E-LSTM-D model to integrate a stacked LSTM into the architecture of encoder-decoder. However, the input to this model is also an adjacent matrices of networks. The flexible deep embedding approach (NetWalk) [21] utilises an improved random walk to extract the topological and temporal features of the network. However, it does not take the previous snapshot into account to extract features, so the feature representation capability is insufficient.

To tackle the above identified problems, we proposed in this work a novel model named TT-GWNN for link prediction in temporal networks, an overview of our TT-GWNN model is described in Figure 1. The model adoptes efficient neural networks to deeply embed topological and temporal features in order to effectively predict links for temporal networks. Inspired by the recent result [22], we proposed a second-order weighted random walk sampling algorithm (SWRW) to effectively capture both topological and temporal features. The work [22] only considered the weight of direct neighbors of a given node, so it was insufficient to capture topological and temporal features of temporal networks. However, in the real-life networks, a node's neighbors of neighbors also have some useful information about the node. Hence we proposed the SWRW algorithm to extract topological and temporal features from a given node from its neighbors and neighbors of neighbors in the previous network snapshots. More specifically, our SWRW algorithms combine previous snapshots of first-order and second-order weight into a weighted graph, i.e., the weighted graph combines the topological and temporal features of the temporal network with weights. It also incorporates a damping factor to assign greater weights to more recent snapshots, which can better preserve the evolving weights of temporal networks. Particles then walke according to the weights. In this way, SWRW can better preserve both topological structure and temporal evolution features of the networks. In addition, compared with the model input being the adjacency matrix, it can decrease the input dimension of the model. We then adopted graph wavelet neural networks (GWNN) as proposed [23] to embed the topological and temporal features into vectors. During the link prediction phase, we used gated recurrent units (GRUs) [24] to build the model and applied the main idea of the C3D approach [25] for training, which can effectively capture the time dependence among network snapshots.

Our major contributions in this work can be summarised as follows.

- We proposed a model TT-GWNN to perform link prediction in temporal networks. The model adopts a graph wavelet neural network (GWNN) to deeply embed nodes in the networks. GWNN takes graph wavelets instead of the eigenvectors of graph Laplacian as a set of bases and defines the convolution operator via wavelet transform and convolution theorem. Comparing with traditional Graph Convolutional Networks (GCN) [26], GWNN does not require the eigendecomposition of the Laplacian matrix and

thus is more efficient [23].

- We proposed a second-order weighted random walk sampling algorithm (SWRW) for both topological and temporal feature extraction, which can effectively capture the evolving behavior of temporal networks. It samples neighbours of a given node according to the weight coefficient. More specifically, for a current snapshot, the SWRW combines its previous snapshots of first-order and second-order weight into a weighted graph and uses a damping factor to assign greater weights to more recent snapshots, which can better preserve the evolving weights of temporal networks
- Experiments on four real-world datasets (i.e., Facebook friendships, Hep-Ph, Digg and Facebook wall posts) demonstrated that our TT-GWNN consistently outperformes a few state-of-the-art baseline models.

The rest of the paper is organised as follows. Section 2 summarises several related works. We formulate the problem and presents some notations in Section 3. Sections 4 presents in detail our proposed framework TT-GWNN for link prediction in temporal networks. Section 5 discuss the experimental results, and we conclude the paper in Section 6.

2. RELATED WORK

In this section, we briefly summarise related work for link prediction in temporal networks. Link prediction in static networks has been extensively studied [5, 27]. Since networks are continually evolving with time in real life, it is important and necessary to study link prediction in temporal networks. Sharan and Neville [13] summarizing the matrices associated to prediction link in temporal networks. Yu et al. [28] proposed a regularized non-negative matrix factorization (NMF) algorithm, which improves the accuracy of link prediction. However, both methods heavily rely on tedious and error-prone hand-crafted features. Other approach is based on matrix factorisation to explore the topology of the networks [8, 9, 10]. The main idea is that the closer two nodes are, the more likely they are to form a link in the near future. However, real-life networks are often evolving, methods considering only topological information may have poor performance. There exist a few other methods focusing on both topological and temporal evolution features proposed in the literature [6, 11] – the work [6] constructs a sequence of higher-order proximity matrices to capture the implicit relationships among nodes, while the work [11] defines the network dynamics as a function of time, which integrates the topology of networks at each timestamp and the temporal network evolution. However, both of them have limited ability in extracting the correlation of high dimensional features due to the fact that they are still based on matrix factorisation. Another model called DynamicTriad [12] preserves both topological features and temporal evolution patterns of a given network. It models how a closed triad, which consists of three nodes connected with each other, develops from an open triad that has two of three vertices not connected with each other. However, it is difficult to form such a triad mechanism for more sparse networks over time, so this method is not effective when dealing with sparse networks.

In recent years, the network embedding approach based on neural networks has gained a lot of popularity [15], and it aims to embed each node of a network into a low-dimensional space. This approach has also proved to be very effective in temporal networks for link prediction [7]. Many different network embedding methods have been proposed in the literature, including DeepWalk [15], node2vec [16], and SDNE [17]. However, most of these methods focus on representation learning for static networks and cannot obtain temporal features in temporal networks directly. Recently, the conditional temporal restricted Boltzmann machine (ctRBM) [18] extends the structure of a standard RBM by leveraging the temporal features of temporal networks. However, the ctRBM model is considered as a shallow model with limited ability to extract nonlinear features. Thus, it is necessary to extend neural network-based methods for aggregating both topological and temporal features using deep model for link prediction. The deep belief network (DBN) [19] explores the topological features of the static network at each timestamp for link prediction, which has shown a good generalisation ability. However, the method merely explores the topological features of the static network at each timestamp and weak ability to capture time dependencies. Li et al. [7] developed a link prediction model capturing both topological and temporal features using GRUs inspired by the machine translation problem of encoder-decoders methods [24]. However, the input to this model is the adjacent matrix of the networks, hence it can incur high computational cost. Most recently, Chen et al. [20] developed an end-to-end E-LSTM-D model to integrate a stacked LSTM into the architecture of encoder-decoder. It imposes more penalty to exist links in the objective to cope with the problem of sparsity. However, the input to this model is still the adjacent matrixes of networks. NetWalk [21] is an flexible deep embedding approach, and it uses an improved random walk to extract the topological and temporal features of the network. The approach can update the network representation dynamically as the network evolves by clique embedding. However, it does not take the previous snapshot into account to extract features, so the feature representation capability is still insufficient.

3. PROBLEM DEFINITION

In this section, we present the necessary definitions and formally describe our research problem in this paper.

DEFINITION 3.1 (Network). *A network can be represented graphically:* $G = \langle V, E\rangle$, *where* $V = \{v_1, \ldots, v_n\}$ *represents a set of nodes, and* n *is the number of nodes in the network, and* $E \subseteq V \times V$ *represents a set of links (edges).*

DEFINITION 3.2 (Temporal network). *A temporal network is defined as* $G_t = \langle V, E_t\rangle$, *which represents a network* $G = \langle V, E\rangle$ *evolving over time and generates a sequence of snapshots* $\{G_1, \ldots, G_T\}$, *where* $t \in \{1, \ldots, T\}$ *represents the timestamps.*

Note that in the above definition the set of nodes is fixed, while the edges E_t can evolve over time.

Link prediction for temporal networks: For a temporal network G_t, and its adjacency matrix can be represented as A_t, where t represents the timestamps. For the A_t, it is a 2-D array that stores the vertex relationships. The element in the A_t can be represented as a_{ij}, where i represents a row and j represents a column. If $a_{ij} = 0$, there is no edge between vertex i and j, otherwise, there is an edge. Given a network sequence of T snapshots $\{A_1, \ldots, A_T\}$, the goal is to predict the adjacency matrix A_{T+1} at time point $T+1$.

4. TT-GWNN

In this section, we introduce our model for link prediction, which is called TT-GWNN (see its overview in Figure 1). In our model, we firstly proposed a second-order weighted random walk sampling algorithm (SWRW) to extract both topological and temporal features for each node in temporal networks (Section 4.1). The sampled topological and temporal features are then fed into a Graph Wavelet Neural Network (GWNN) for network embedding (Section 4.2). Finally, GRUs are adopted to predict new links and GRUs output is compared with ground truth to minimise the square loss.

4.1. Topological and temporal feature extraction

For a given temporal network $G_t = \langle V, E_t\rangle$, the traditional method of sampling neighbour node is to use either the Breadth-First-Search (BFS) algorithm or the Depth-First-Search (DFS) algorithm. Base on the BFS/DFS algorithms, DeepWalk [15], node2vec [16], and role2vec [29] were proposed, and it can be used to embed network nodes into a low dimensional vector. However, for these methods, they sample neighbor node in the current snapshot, which only capture topological structure features and ignore temporal evolution features. Instead, we proposed the SWRW algorithm to sampling nodes for each node v of each snapshot. More specifically, the SWRW combines previous snapshots of first-order weight and second-order weight into a weighted graph and uses a damping factor γ to assign greater weight to more recent snapshots, which can combine topological and temporal features on a weighted graph. Particles then walk according to the weights. Based on the above ideas, we designed the weight matrix.

Weight matrix: For a temporal network $G_t = \langle V, E_t\rangle$, which evolves over time and generates a sequence of snapshots $\{G_1, \ldots, G_T\}$, where $t \in \{1, \ldots, T\}$ represents the timestamps. Adjacency matrixes of each snapshot can be represented as $\{A_1, \ldots, A_T\}$. The element in the A_t can be represented as a_{ij}, which represents a binary value for unweighted networks, and connection strength value of the links between nodes for weighted networks. For our experiments, we transformed unweighted networks into weighted networks. What we do was we used the number of shared neighbors of nodes i and j plus the element a_{ij} of the adjacency matrix A_t as the weight for nodes of each snapshot G_t. The weight matrix W_t for each snapshot is defined as 1

$$W_t = \sum_{k=1}^{t} \gamma^{t-k}(A_k + A_k^2) \qquad 0 < \gamma < 1, \quad 1 \le t \le T \tag{1}$$

Where γ is the damping factor. A proper value is very important for capturing networks evolving and the larger the value, the more efficient it is for networks that are more stable over time, and vice versa. The A_k^2 represents a 2-hop relationship between nodes in timestamp k [30]. The elements in the W_t can be represented as w_{ij}^t, where i represents a row and j represents a column and its value represents the strength of the relationship between node i and node j in timestamp t. The W_t preserves previous snapshots of first-order weight and second-order weight into a weighted graph, which can combine topological and temporal features on a weighted graph.

Random walk by weight matrix: For a random walk of length L, we defined the transition probability as s_{ij}, which represents the probability of node i walking to the node j.

$$s_{ij}^t = \frac{w_{ij}^t}{\sum_{v_k \in N(v_i)} w_{ik}^t} \tag{2}$$

Where w_{ij}^t is the (i, j) element of the weight matrix W_t, and $N(v_i) = \{v \mid v \in V, (v, v_i) \in E\}$ is the set neighbors of v_i. The detailed algorithm description is shown in Algorithm 1.

Algorithm 1 has two parameters: L the length of the sampled paths defining the path length of a random walk for a given start node v, and R the number of sampled paths defining how many random walks there are for a given start node v. In Algorithm 1, each $X[i]$ represents the sets of sampled neighbors for all nodes

Algorithm 1: Second-order weighted random walk sampling algorithm (SWRW)

Input : $G_t\langle V, E_t\rangle$: a temporal network, we divide it evenly into a sequence of snapshots $\{G_1, \ldots, G_t\}$ by timestamp;
L: length of the sampled paths;
R: number of sampled paths;

Output: $X[i]$ with $i \in \{1, \ldots, T\}$, where each $X[i]$ consists of the sample sets for every node v in the network;

1 **for** $i \in \{1, \ldots, T\}$ **do**
2 Compute the transition probability matrix S^i according to Eq. 2;
3 Select a snapshot $G_i(V, E_i)$;
4 **for** $v \in V$ **do**
5 **for** $k \in \{1, \ldots, R\}$ **do**
6 $v_{k,1} = v$;
7 **for** $j \in \{1, \ldots, L\}$ **do**
8 Select a node $v_{k,j+1}$ in $N(v_{k,j})$ according to the transition probability S^i;
9 Add the node $v_{k,j+1}$ it to x_i^v;
10 **end**
11 **end**
12 $X[i]$.add(x_i^v)
13 **end**
14 **end**

in the network at time i, and the x_i^v represents the sets of sampled neighbors for node v in the network at time i, and the $N(v)$ represents set neighbors of v. The first layer loop is used to select snapshot at time i from the temporal network $G_t(V, E)$, and the purpose is to traverse all the snapshots. The second loop is used to traverse each node $v \in V$ for each snapshot, and the purpose is to obtain topological and temporal features for each $v \in V$. The third loop and fourth loop are used to sample topological and temporal features for each $v \in V$ of each snapshot according the transition probability matrix.

Time complexity analysis: Let T represent the number of snapshots, n represents the number of nodes, L represents the length of the sampled paths, and R represents the number of sampled paths. The time complexity for the SWRW algorithm to generate a set of samples $X[i]$ ($i \in \{1, \ldots, T\}$) is $O(T \cdot R \cdot L \cdot n)$ for each node v of each snapshot in temporal networks. For a temporal network, Equation 1 constructs the combined weight matrix W_t for each snapshot. We adopted sparse matrix multiplication with complexity approaching $O(n^2)$. Line 2 of the algorithm computes the transition probability S_{ij}^t for each node v with other nodes in $O(T)$ time. Therefore, it requires $O(T \cdot R \cdot L \cdot n)$ time to compute the transition probability matrix for all n nodes of each snapshot in temporal networks. Because T, R, L, and n can be treated as constants, the time complexity of our proposed algorithm SWRW is $O(n^2)$.

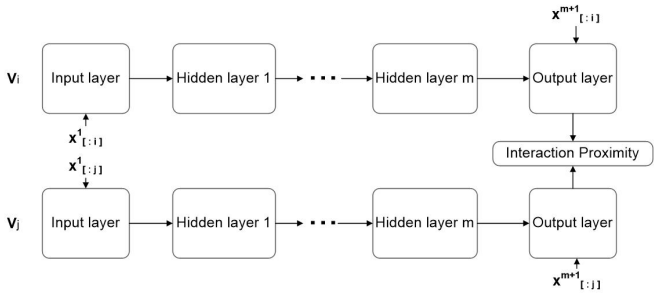

FIGURE 2. GWNN model for each network snapshot embedding: The model input is X^1, and the output is X^{m+1}, where $X^1_{[:,i]}$ and $X^{m+1}_{[:,i]}$ are the i-th column of X^1 and X^{m+1} respectively and $X^1_{[:,j]}$ and $X^{m+1}_{[:,j]}$ are the j-th column of X^1 and X^{m+1} respectively, where i and j represent any two nodes; we train the model and update the parameters through Formula 5.

4.2. Neural network model

Our neural network model consists of the embedding layer and the GRU layer as described in Figure 1.

Embedding layer (GWNN). Network embedding [15] encodes network structural properties into a low-dimensional matrix $X \in R^{D \times |V|}$, where each column represents the representation of a node in the network. In the model, we adopted Graph Wavelet Neural Network (GWNN) to map nonlinearly a node u to its $D-$dimensional representation $x_u \in R^D$. GWNN is a novel graph convolution neural network, which leverage wavelet transformer to instead of the Fourier transformer of GCN. Comparing with traditional Graph

Algorithm 2: Graph wavelet neural network embedding for temporal networks

Input : $G_t\langle V, E_t\rangle$: a temporal network, we divide it evenly into a sequence of snapshots $\{G_1, \ldots, G_t\}$ by timestamp;
$X^{1,t}$: it is a feature matrix of each node of the snapshot t for model input;
m: the number of layers;

Output: Y_t with $t \in \{1, \ldots, T\}$, where each $Y_t \in R^{N\times k}$ (N is the number of nodes and K is the dimension) is the representation of G_t;

```
for t ∈ {1, ..., T} do
    X^1 = X^{1,t};
    Compute X^{m+1} from the our design model (see Equations 3 and 4);
    Y_t = X^{m+1}
end
```

Convolutional Networks (GCN) [26], GWNN does not require the eigendecomposition of the Laplacian matrix and thus is more efficient (see [23] for the time complexity analysis of the GWNN).

The GWNN focused on static networks, we designed a m-layer GWNN for each snapshot for unsupervised node learning for temporal networks embedding. For the m-th layer GWNN, the input to each GWNN layer is a node feature matrix, X^1, with dimensions $n \times p$ and the output tensor is X^{m+1} with dimensions $n \times c$. The formulation of our model for each snapshot is (the framework of GWNN is described in Figure 2)

$$X^2_{[:,j]} = ReLU(\psi_s \sum_{i=1}^{p} F^1_{i,j} \psi^{-1} X^1_{[:,i]}) \quad j = 1, \ldots, q \quad (3)$$

$$\vdots$$

$$X^{m+1}_{[:,k]} = ReLU(\psi_s \sum_{i=1}^{q} F^m_{i,j} \psi^{-1} X^m_{[:,i]}) \; k = 1, \ldots, c \quad (4)$$

where $X^1_{[:,i]}$ with the dimensions $n \times 1$ is the i-th column of X^1, $ReLU$ is a non-linear activation function, ψ_s is wavelet bases, ψ^{-1} is the graph wavelet transform matrix at scale s which projects signal in vertex domain into spectral domain, $F^n_{i,j}$ is a diagonal filter matrix learned in spectral domain in layer n [23], c is the embed dimension of each node, X^{m+1} of dimensions $n \times c$ is the embedding matrix of networks. We defined the following loss function for each snapshot embedding to train the model:

$$Loss = \frac{1}{n} \sum_{i=1}^{n} (X^{m+1}_{[:,i]} - Average(Adj(X^{m+1}_{[:,i]})))^2 \quad (5)$$

where n is the number of nodes, and $X^{m+1}_{[:,i]}$ represents the vector representation of the node i in layer $m+1$,and $Adj(X^{m+1}_{[:,i]})$ obtains neighborhood node representation of i. *Average* means an average processing operation.

For all snapshots, we used Algorithm 2 for node embedding. Algorithm 2 has two parameters: $X^{1,t}$: it is a features matrix of each node of the snapshot t, and m: the number of layers defining how many layers GWNN has. In Algorithm 2, the $Y_t \in R^{N\times k}$ (N is the number of nodes and K is the dimension) is the representation of G_t with $t \in \{1, \ldots, T\}$. The loop is used to select snapshot at time t from the temporal network $G_t(V, E)$, and the purpose is to compute X^{m+1} from the our design model 2.

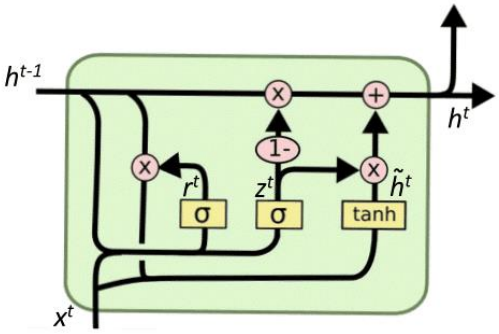

FIGURE 3. The GRU framework [7]

GRU layer. GURs [24] is an improved Recurrent Neural Network (RNNs), which can solve the problem that RNNs cannot deal with long distance dependence. GRU implements two gates, the updated gate, and the reset gate. The update gate is used to control the state information at the previous timestamp is brought into the current state. Reset gate is used to control the state information at the previous moment is ignored. The GRU framework shown in Figure 3.

For the GRU, the computational process can be treated as a black box. The current x^t and previous hidden state h^{t-1} are fed into GRU, and they merge two inputs and compute the current hidden state h^t. This mechanism can effectively preserve historical information for each node.

In this paper, we adopted GRU to predict new links. We also adopted the idea of the C3D [25] approach to train our model, which was similar to the convolution processing of 3-D convolution kernel and can better capture the time dependence of network snapshots. The

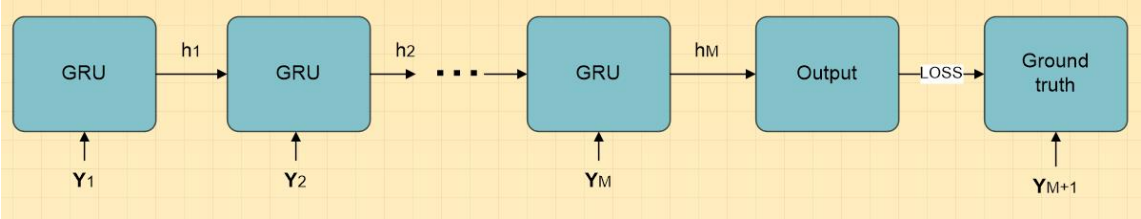

FIGURE 4. Our training framework for GRUs

main idea of C3D is embodied in the training process below. The details of the model for training was shown in Figure 4.

The input of the model is $\{Y_1, \ldots, Y_T\}$, where each Y_t with $t \in \{1, \ldots, T\}$ represents the representation of the G_t. In the training phase, we iteratively selected M snapshots $\{Y_1, \ldots, Y_M\}$ from $\{Y_1, \ldots, Y_T\}$ to train the model, where where $M < T$ represents the number of training sets. We started with using $\{Y_1, \ldots, Y_M\}$ as the training sample and used Y_{M+1} for labelling. After the completion of the training, we continued to use the $\{Y_2, \ldots, Y_{M+1}\}$ as the training sample, Y_{M+2} for labelling, and so on, until the training sample was $\{Y_{T-M}, \ldots, Y_{T-1}\}$, and the label was Y_T. The output of our model was $\hat{Y}_w$. For objective function we used square loss function given by the following formula. After the model training, we inputted $\{Y_{T-M+1}, \ldots, Y_T\}$ into the model to predict the new link.

$$Loss = \frac{1}{T-M} \sum_{w=M+1}^{T} (\hat{Y}_w - Y_w)^2 \qquad (6)$$

5. EXPERIMENTS

In this section, we described the datasets and baseline models, and presented the experimental results to demonstrate TT-GWNN's effectiveness in link prediction of temporal networks.

5.1. Datasets

We selected four temporal networks from different domains in the KONECT project.[1] We utilised two undirected temporal networks (Facebook friendships and Hep-Ph) and two directed networks (Digg and Facebook wall posts). All networks have different sizes and attributes. Their statistic properties are shown in Table 1.

- The Facebook friendships dataset contains friendship data of Facebook users. The nodes represent users and edges are friendship between two users. The dataset contains a very small subset of the total Facebook friendship network. We split it by year and denote them as F_1 to F_5 for our experiments. The snapshot from F_1 to F_4 was used to train the model and the F_5 was used as ground-truth of network inference.
- The arXiv hep-ph dataset is the collaboration network of authors of scientific papers from the arXiv's High Energy Physics-Phenomenology section. Nodes represent authors and links represent common publications. Timestamps denote the date of a publication. The dataset contains 10 years (1991 - 2002) of data. For our experiment, we selected 5 years (1995 - 1999) and denoted them as A_1 to A_5. Each snapshot contains a one-year network structure, and the snapshot of the first 4 years was used to train the model and the last snapshot was used as ground-truth of network inference.
- The Digg dataset is the reply network of the social news website Digg. The nodes represent users of the website, and edges between two nodes denote that a user replied to another user. The dataset contains sixteen-days records, and we split it by day. We evenly merged it into five snapshots by day and denote it as D_1 to D_5. For our experiment, the snapshot from D_1 to D_4 was used to train the model and the D_5 was used as ground-truth of network inference.
- The Facebook wall posts dataset is the directed network of a small subset of posts to other user's wall on Facebook. The nodes represent Facebook users, and each directed edge is one post, linking the users writing a post to the users whose wall the post is written on. The dataset contains 6 years (2004 - 2009) of data. For our experiment, we combined 2004 and 2005 data into one network snapshot and defined it as W_1. The rest of the data was defined as W_2 to W_5 by year and each snapshot contains a one-year network structure. The snapshot of the first 4 years was used to train the model and the last snapshot was used as ground-truth of network inference.

For our experiments on the above datasets, the last snapshot was used as ground-truth of network inference and the other snapshots was used to train the model.

5.2. Evaluation metric and baseline models

In our paper, we adopted the area under the receiver operating curve (AUC) [31] to evaluate the performance

[1] http://konect.uni-koblenz.de/

TABLE 1. The statistics of four temporal networks.

Network	#Nodes	#Links	Clustering coefficient	Format
Facebook friendships	63,731	817,035	14.8%	undirected
Hep-Ph	28,093	4,596,803	28.0%	undirected
Digg	30,398	87,627	0.56%	directed
Facebook wall posts	46,952	876,993	8.51%	directed

of different methods. The AUC relates to the sensitivity (true positive rate) and the specificity (true negative rate) of a classifier. This metric is strictly bounded between 0 and 1. The largera the AUC is, the better the model performs.

We compared TT-GWNN with the following five baseline models: CP-tensor [10], BCGD [8], LIST [11], STEP [6] and NetWalk [21]. Since the input of DDNE [7] and E-LSTM-D [20] are adjacency matrixes, which has a high computational cost when dealing with large-scale networks, we did not use it as the baseline.

- **CP-tensor** [10]: It explores matrix-based and tensor-based approaches to solving the link prediction problem, which the model stacks all adjacency matrices of historical snapshots into a tensor with the time as the third dimension. Then the matrix factorisation method is used to predict links.
- **BCGD** [8]: BCGD is considered as a scalable approach with a temporal latent space model for link prediction, which assumes two nodes are more likely to form a link if they are close to each other in their latent space.
- **LIST** [11]: LIST describes the network dynamics as a function of time, which integrates the topological and temporal consistency in temporal networks. In its implementation, it utilises a linear time function to model network evolving.
- **STEP** [6]: STEP is a relatively new framework for link prediction in temporal networks, which considers both topological and temporal features at the same time. It utilises a joint matrix factorisation algorithm to simultaneously learn the topological and temporal constraints to model network evolution.
- **NetWalk** [21]: The NetWalk model preserves both topological feature and temporal evolution patterns of a given temporal network. The model updates the network representation dynamically as the network evolves by clique embedding. It focuses on anomaly detection, and we adopt its representation vector to predict the link.

Parameter settings. The connected links in the last snapshot are often very sparse with a lot of nodes that are not connected. In the evaluation process, we randomly generated the number of non-linked edges smaller than twice linked edges to ensure data balance [8]. Our proposed TT-GWNN framework in this paper can embed network into a low dimension vector. For the Hep-Ph (28,093 nodes) and Digg (30,398 nodes), we set the output of 256 dimensions. For the Facebook wall posts dataset (46,952 nodes) and Facebook friendships (63,731 nodes), we set the output of 512 dimensions. If we increased or decreased the dimensions, the performance remained the same or even becomes worse. For different datasets, the parameters for baselines are tuned to be optimal. The BCGD method is only for undirected networks. For the directed networks, we transfered the adjacency matrix of directed networks to undirected networks by $(A^{\mathsf{T}}+A)/2$ [6]. Other settings include: the learning rate of the model was set as 0.0001; the length of the sampled paths L was set 80;the number of sampled paths R was set 10; the M for the training model was set as 3; the M for layers of the GWNN was set as 5; the damping factor γ was set as 0.75. For the result of our experiment, we carried out five times independently and reported the average AUC values for each dataset. The experiments were conducted on the Ubuntu 16.04 operating system with a 3.7GHz machine, 64GB memory, GeForce GTX 1080 Ti, and Python 3.5.

5.3. Experimental results

For experiments, we compared the performance of five baselines on four temporal networks for link prediction. We first embed each vertex into a vector at each snapshot. For each dataset, we divided by timestamp, and the last snapshot was used as ground-truth of network inference, and the previous snapshots are used to train the TT-GWNN model. After training, we shifted the window one step towards the future to obtain the vector representation of each node for last snapshot. Last, we used the obtained representations to predict the network structure.

Table 2 compares AUCs over the four datasets. Compared with the baselines, our method, TT-GWNN, achieved the best performance. Especially on the Facebook Friendships dataset, the AUC value of our model was higher than 11% compared to other baseline models. The reason may be that when γ was fixed, the smaller the value, the more efficient it was for networks that were more unstable over time. However, the dataset becomes more sparse over time compared to other datasets, thus achieving the good performance. Essentially, we used the SWRW algorithm to sampling topological and temporal features for each node v

TABLE 2. Prediction results for the four datasets (AUC value).

Model	Hep-Ph	Digg	Facebook wall posts	Facebook friendships
CP-tensor	0.54	0.66	0.71	0.52
BCGD	0.60	0.68	0.74	0.61
LIST	0.63	0.73	0.72	0.55
STEP	0.61	0.74	0.75	0.57
NetWalk	0.69	0.71	0.74	0.70
TT-GWNN	**0.75**	**0.77**	**0.76**	**0.81**

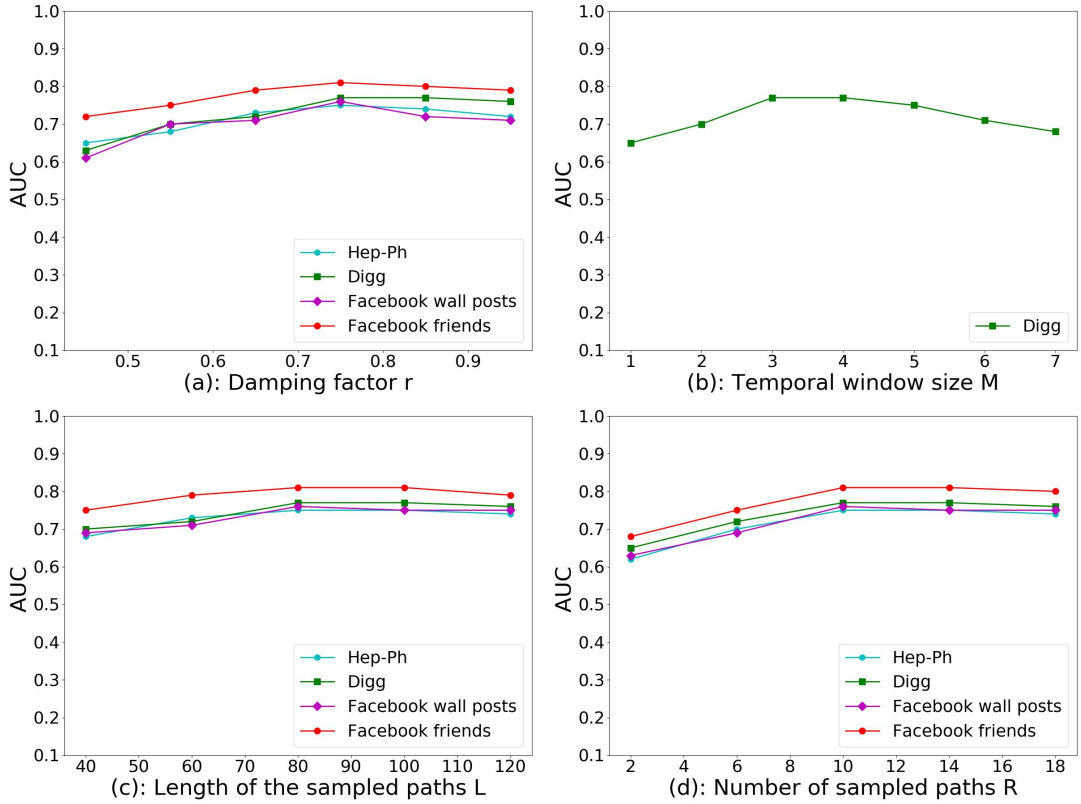

FIGURE 5. Experiments on parameters sensitivity. (a) TT-GWNN's performance on four datasets when increasing γ. (b)TT-GWNN's performance on the Digg dataset when increasing M. (c) TT-GWNN's performance on four datasets when increasing L. (d) TT-GWNN's performance on four datasets when increasing R.

of temporal networks, which samples neighbors of a given node according to the first-order and second-order weight coefficient. The algorithm preserves the evolving weights of temporal networks and can better capture topological and temporal features. TT-GWNN also adopts GWNN to deeply embed node topological and temporal features, this can better capture the nonlinear network attributes due to it was a deep model [7] and had a relatively low computational cost. As such, it had advantages over the above baseline models. Moreover, our model adopts the idea of C3D for training, which can capture the evolution of the network and also contributed to the improvement of performance.

5.4. Parameter sensitivity analysis

We further performed parameter sensitivity analysis in this section, and the results are summarised in Figure 5. Specifically, we estimated how different the damping factor γ and the M for the training and the length of the sampled paths L and the number of sampled paths R can affect the link prediction results.

- The damping factor γ. We varied the damping factor γ range from 0.45 to 0.95, with each step increasing by 0.1 to prove the effect of varying this parameter. When this parameter was verified, other parameters were set with their default values. The results showed that the best result was obtained when $\gamma = 0.75$. As can be seen from the Figure 5a, as γ increases from 0.45 to 0.75, the performance continued to increase. The best result was obtained at $\gamma = 0.75$, after which the performance decreased slightly or remained unchanged, while γ continues to increase.

- The temporal window size M for the training. Due to the fact that the Digg dataset contains sixteen-days records, we split it by day and it will generates 16 snapshots. Since it had more snapshots than other datasets, we selected the Digg dataset to conduct sensitivity analysis for the parameter of the temporal window size M. We varied the window size from 1 to 7 to check the effect of varying this parameter. When this parameter was verified, other parameters were set with their default values. The results showed that the best results are obtained when $M = 3$. The reason might be that the closer snapshot is to the current snapshot, the more information about the current snapshot. As can be seen from Figure 5b, the accuracy no longer increases, when M continuously increases.
- The length of the sampled paths L and the number of sampled paths R. Since L and R jointly determined the sampling size of the current node, we analyzed these two parameters together. When analyzing L, R was set to 10, and when analyzing R, L was set to 80. The experimental results showed that the performance was the best when $L = 80$ and $R = 10$ (the detail was described in Figure 5c and 5d).

5.5. Scalability analysis

In this section, we compared the efficiency of TT-GWNN by using the Facebook friendship dataset. We randomly generated a subset of the original Facebook networks with different numbers of nodes (20,000, 30,000, ···, 60,000) and we tested the running time for each method. We conducted five independent experiments to report average efficiency in runtime (seconds). As seen from the Figure 6, the running time of our model is lower than that of the baseline model. As the number of nodes increases, the running time of our model increases more slowly compared to the baseline model in the Facebook friendship dataset. Therefore, our model has the advantage of handling large-scale networks.

6. CONCLUSIONS

We have proposed an effective framework, TT-GWNN, for link prediction in temporal networks, which captures both topological and temporal evolution features of the networks. Essentially, we proposed the SWRW algorithm to extract both topological and temporal features in each snapshot to model network evolution. The algorithm combined previous snapshots of first-order weight and second-order weight into a weighted graph and used a damping factor to assign greater weight to more recent snapshots. In this way, SWRW can better preserve both topological structure and temporal evolution features of the networks.

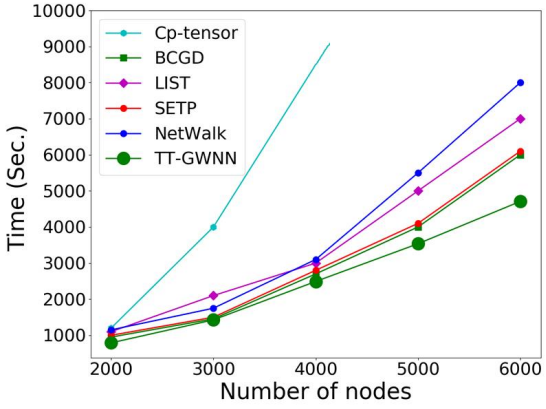

FIGURE 6. Comparison of the runtime of different approaches on the Facebook friendship dataset with respect to the size of the networks.

Experiments demonstrated the effectiveness of our TT-GWNN model and it achieved significant gains in performance than the baseline models.

Our future work will further study how to improve the feature extraction method in order to better capture the topological and temporal features of nodes in temporal networks. As the use of aggregation methods can preserve more useful node features [32], our future work will also focus on how to aggregate topological and temporal features more effectively and study performance improvements from different aggregation approaches. For real-life networks, they usually include some heterogeneous information such as text, locations [27] user attributes [33], etc. Therefore, we also consider aggregating heterogeneous features to improve the representation of features and pay more attention to time and space complexity by conducting more comprehensive experiments.

ACKNOWLEDGEMENTS

This work has been supported by the Chongqing Graduate Research and Innovation Project (Grant No. CYB19096), the Capacity Development Grant of Southwest University (Grant No. SWU116007), the Fundamental Research Funds for the Central Universities (Grant No. XDJK2020D021), the National Natural Science Foundation of China (Grant No. 61672435, 61732019, 61811530327), and the China Scholarship Council (Student No. 202006990041).

REFERENCES

[1] Holme, P. and Saramäki, J. (2012) Temporal networks. *Physics reports*, **519**, 97–125.
[2] Oliveira, M. and Gama, J. (2012) An overview of social network analysis. *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, **2**, 99–115.
[3] Pavlopoulos, G. A., Wegener, A. L., and Schneider, R. (2008) A survey of visualization tools for biological network analysis. *Biodata Mining*, **1**, 12.

[4] Yang, J. and Leskovec, J. (2012) Defining and evaluating network communities based on ground-truth. *Proceedings of the ACM SIGKDD Workshop on Mining Data Semantics*, Beijing, China, 12-16 August, pp. 181–213. ACM, NY, USA.

[5] Martínez, V., Berzal, F., and Cubero, J.-C. (2017) A survey of link prediction in complex networks. *ACM Computing Surveys*, **49**, 69.

[6] Chen, H. and Li, J. (2018) Exploiting structural and temporal evolution in dynamic link prediction. *Proceedings of the 27th ACM International Conference on Information and Knowledge Management*, Torino, Italia, 22-26 October, pp. 427–436. ACM, NY, USA.

[7] Li, T., Zhang, J., Philip, S. Y., Zhang, Y., and Yan, Y. (2018) Deep dynamic network embedding for link prediction. *IEEE Access*, **6**, 29219–29230.

[8] Zhu, L., Guo, D., Yin, J., Ver Steeg, G., and Galstyan, A. (2016) Scalable temporal latent space inference for link prediction in dynamic social networks. *IEEE Transactions on Knowledge and Data Engineering*, **28**, 2765–2777.

[9] Deng, D., Shahabi, C., Demiryurek, U., Zhu, L., Yu, R., and Liu, Y. (2016) Latent space model for road networks to predict time-varying traffic. *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, San Francisco, USA, 24-27 August, pp. 1525–1534. ACM, NY, USA.

[10] Dunlavy, D. M., Kolda, T. G., and Acar, E. (2011) Temporal link prediction using matrix and tensor factorizations. *ACM Transactions on Knowledge Discovery from Data*, **5**, 10.

[11] Yu, W., Cheng, W., Aggarwal, C. C., Chen, H., and Wang, W. (2017) Link prediction with spatial and temporal consistency in dynamic networks. *Proceedings of the International Joint Conference on Artificial Intelligence*, Melbourne,Australia, 19-25 August, pp. 3343–3349. Morgan Kaufmann, San Francisco, USA.

[12] Zhou, L., Yang, Y., Ren, X., Wu, F., and Zhuang, Y. (2018) Dynamic network embedding by modeling triadic closure process. *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence*, Louisiana, USA, 2-7 February, pp. 571–578. AAAI Press, CA, USA.

[13] Sharan, U. and Neville, J. (2008) Temporal-relational classifiers for prediction in evolving domains. *Proceedings of the 8th IEEE International Conference on Data Mining (ICDM 2008), December 15-19, 2008, Pisa, Italy*, Pisa, Italy, 15-19 December, pp. 540–549. IEEE, NJ, USA.

[14] Lei, Y. and Liu, H. (2003) Feature selection for high-dimensional data: a fast correlation-based filter solution. *Proceedings of the 20th International Conference on Machine Learning*, Tianjin, China, 14-17 July, pp. 856–863. ACM, NY, USA.

[15] Perozzi, B., Al-Rfou, R., and Skiena, S. (2014) Deepwalk: Online learning of social representations. *Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, New York, USA, 24-27 August, pp. 701–710. ACM, NY, USA.

[16] Grover, A. and Leskovec, J. (2016) node2vec: Scalable feature learning for networks. *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, San Francisco, USA, 24-27 August, pp. 855–864. ACM, NY, USA.

[17] Wang, D., Cui, P., and Zhu, W. (2016) Structural deep network embedding. *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, San Francisco, USA, 24-27 August, pp. 1225–1234. ACM, NY, USA.

[18] Li, X., Du, N., Li, H., Li, K., Gao, J., and Zhang, A. (2014) A deep learning approach to link prediction in dynamic networks. *Proceedings of the 2014 SIAM International Conference on Data Mining*, Philadelphia, USA, 24-26 April, pp. 289–297. SIAM, PA, USA.

[19] Liu, F., Liu, B., Sun, C., Liu, M., and Wang, X. (2015) Deep belief network-based approaches for link prediction in signed social networks. *Entropy*, **17**, 2140–2169.

[20] Chen, J., Zhang, J., Xu, X., Fu, C., Zhang, D., Zhang, Q., and Xuan, Q. (2019) E-lstm-d: A deep learning framework for dynamic network link prediction. *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, **8**, 1–14.

[21] Yu, W., Cheng, W., Aggarwal, C. C., Zhang, K., Chen, H., and Wang, W. (2018) Netwalk: A flexible deep embedding approach for anomaly detection in dynamic networks. *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, London, United Kingdom, 19-23 August, pp. 2672–2681. ACM.

[22] Ahmed, N. M., Chen, L., Wang, Y., Li, B., Li, Y., and Liu, W. (2016) Sampling-based algorithm for link prediction in temporal networks. *Information Sciences*, **374**, 1–14.

[23] Xu, B., Shen, H., Cao, Q., Qiu, Y., and Cheng, X. (2019) Graph wavelet neural network. *Proceedings of the International Conference on Learning Representations*, Louisiana, USA, 6-9 May, pp. 1–13. Springer, Berlin.

[24] Cho, K., Van Merriënboer, B., Bahdanau, D., and Bengio, Y. (2014) *On the properties of neural machine translation: Encoder-decoder approaches.* arXiv, arXiv:1409.1259.

[25] Tran, D., Bourdev, L., Fergus, R., Torresani, L., and Paluri, M. (2015) Learning spatiotemporal features with 3D convolutional networks. *Proceedings of the IEEE International Conference on Computer Vision*, Santiago, Chile, 7-13 December, pp. 4489–4497. IEEE, NJ, USA.

[26] Kipf, T. N. and Welling, M. (2017) Semi-supervised classification with graph convolutional networks. *Proceedings of the International Conference on Learning Representations*, Toulon, France, 24-26 April, pp. 1–14. Springer, Berlin.

[27] Zhang, Y. and Pang, J. (2015) Distance and friendship: A distance-based model for link prediction in social networks. *Proceedings of the 17th Asia-Pacific Web Conference*, Guangzhou, China, 18-20 September, LNCS, **9313**, pp. 55–66. Springer.

[28] Yu, W., Aggarwal, C. C., and Wang, W. (2017) Temporally factorized network modeling for evolutionary network analysis. *Proceedings of the Tenth ACM In-*

ternational Conference on Web Search and Data Mining, Cambridge, UK, 6-10 February, pp. 455–464. ACM, NY, USA.

[29] Ahmed, N. K., Rossi, R., Lee, J. B., Willke, T. L., Zhou, R., Kong, X., and Eldardiry, H. (2018) *Learning role-based graph embeddings*. arXiv, arXiv:1802.02896.

[30] Abu-El-Haija, S., Perozzi, B., Kapoor, A., Harutyunyan, H., Alipourfard, N., Lerman, K., Steeg, G. V., and Galstyan, A. (2019) Mixhop: Higher-order graph convolution architectures via sparsified neighborhood mixing. *Proceedings of the 36th International Conference on Machine Learning*, Long Beach, California, 10-15 July, pp. 856–863. IEEE, NJ, USA.

[31] Huang, J. and Ling, C. X. (2005) Using auc and accuracy in evaluating learning algorithms. *IEEE Transactions on Knowledge and Data Engineering*, **17**, 299–310.

[32] Hamilton, W., Ying, Z., and Leskovec, J. (2017) Inductive representation learning on large graphs. *Proceedings of the Advances in Neural Information Processing Systems*, Long Beach, USA, 4-9 December, pp. 1024–1034. MIT Press, Massachusetts, USA.

[33] Pang, J. and Zhang, Y. (2017) DeepCity: A feature learning framework for mining location check-ins. *Proceedings of the 11th International AAAI Conference on Web and Social Media*, Montreal, Canada, 15-18 May, pp. 652–655. AAAI Press.