Spatio-Temporal Attentive RNN for Node Classification in Temporal Attributed Graphs

Dongkuan Xu[1*] , **Wei Cheng**[2*] , **Dongsheng Luo**[1] , **Xiao Liu**[1] and **Xiang Zhang**[1*]
[1]The Pennsylvania State University
[2]NEC Laboratories America, Inc.
{dux19, dul262, xxl213, xzz89}@psu.edu, weicheng@nec-labs.com

Abstract

Node classification in graph-structured data aims to classify the nodes where labels are only available for a subset of nodes. This problem has attracted considerable research efforts in recent years. In real-world applications, both graph topology and node attributes evolve over time. Existing techniques, however, mainly focus on static graphs and lack the capability to simultaneously learn both temporal and spatial/structural features. Node classification in temporal attributed graphs is challenging for two major aspects. First, effectively modeling the spatio-temporal contextual information is hard. Second, as temporal and spatial dimensions are entangled, to learn the feature representation of one target node, it's desirable and challenging to differentiate the relative importance of different factors, such as different neighbors and time periods. In this paper, we propose STAR, a spatio-temporal attentive recurrent network model, to deal with the above challenges. STAR extracts the vector representation of neighborhood by sampling and aggregating local neighbor nodes. It further feeds both the neighborhood representation and node attributes into a gated recurrent unit network to jointly learn the spatio-temporal contextual information. On top of that, we take advantage of the dual attention mechanism to perform a thorough analysis on the model interpretability. Extensive experiments on real datasets demonstrate the effectiveness of the STAR model.

1 Introduction

A large amount of graphs have been generated because of the rapid growth of information, such as co-author graphs [Ni *et al.*, 2018], brain graphs [Wang *et al.*, 2017a] and post graphs [Hamilton *et al.*, 2017]. Node classification approaches aim to classify the nodes into different categories according to graph topology and node attributes. There have been a lot of efforts on node classification [Perozzi *et al.*, 2014; Grover and Leskovec, 2016; Kipf and Welling, 2017; Veličković *et al.*, 2018]. node2vec [Grover and Leskovec, 2016] adopts a biased random walk to learn richer node representations and classifies nodes via a logistic regression model. GCN [Kipf and Welling, 2017] introduces a convolutional architecture directly applied on graphs and node labels are predicted by a softmax function. An attention-based network that performs node classification in graphs is proposed in [Veličković *et al.*, 2018].

In many real-world applications, graphs are often dynamic and evolve rapidly over time. Hence, basic graph concepts are further extended to temporal graphs. Some recent progresses have been made on the node classification in temporal graphs [Du *et al.*, 2018; Goyal *et al.*, 2018b; Zhou *et al.*, 2018; Zhu *et al.*, 2018]. For example, DynamicTriad [Zhou *et al.*, 2018] extracts the evolution patterns of temporal graphs and uses a logistic regression model to classify nodes. DNE [Du *et al.*, 2018] extends the skip-gram based graph embedding methods into dynamic settings. The work in the literature is mostly designed for the case where graph topology evolves over time. However, a vast majority of real-world networks are coupled with a rich set of vertex attributes, which also evolve over time. As temporal and spatial dimensions are entangled, it's desirable to design an effective approach that can jointly model the spatio-temporal contextual information for both graph topology and node attributes, at the same time provide model interpretability for the classification results.

In this work, we focus on the problem of node classification in temporal attributed graphs. Given a sequence of attributed graphs for the same set of nodes, where each node has a unique class label over a period of time, the task is to predict the labels of unlabeled nodes by learning from the labeled ones. Both graph topology and node attributes change over time. The problem is challenging for two major reasons. First, effectively modeling the spatio-temporal contextual information is hard. For the temporal attributed graphs, the evolution lies in both graph topology and node attributes. Along with the changes, the temporal and spatial dimensions are entangled. How to jointly learn the node representations on both spatial and temporal aspects simultaneously is challenging. Second, differentiating the relative importance of different factors that influence node representations for effective classification is also challenging. Different neighbors at different time periods will influence node representations diversely. As the changes in temporal and spatial dimensions

[*]These authors contributed equally to this work

jointly determine the node labels, it is desirable to recognize which neighbors in which time periods exert more influence on the representation learning of the target node.

To address the above challenges, we propose a spatio-temporal attentive recurrent neural network model (STAR). STAR aims to learn node representations for classification by jointly considering both the temporal and spatial patterns of the node. Specifically, STAR embeds the sampling and aggregation of local neighbor nodes into a gated recurrent unit (GRU) network [Cho *et al.*, 2014] to jointly learn the spatio-temporal contextual information. By feeding sequential node attributes at different time periods into the GRU network, the temporal features of attributes evolution can be effectively extracted. The process is entangled with the graph topology learning, in which the neighborhood representations are sampled and aggregated to learn the spatial information of the target node. Moreover, a dual attention mechanism on both temporal and spatial aspects is developed to selectively aggregate different time periods and different neighbors, which helps to perform a thorough analysis on the model interpretability. In particular, based on the temporal attention, we can detect the time periods that are more important for the node representations for classification, while the spatial attention can help us put more emphasis on the important neighbors. To summarize, the major contributions are as follows.

- We propose STAR. STAR is the first model that can provide the model interpretability for the node classification in temporal attributed graphs.
- We design a spatio-temporal GRU network that can jointly model the temporal evolution of both node attributes and graph topology.
- We perform extensive experiments on real datasets to evaluate the effectiveness of the proposed model. The results demonstrate the effectiveness of STAR.

2 The Problem

A temporal attributed graph is a collection of snapshots of an attributed graph at different time steps, denoted by $\mathbb{G} = (\mathcal{G}^1, \mathcal{G}^2, \cdots, \mathcal{G}^T)$. $\mathcal{G}^t = (\mathcal{V}, \mathbf{A}^t, \mathbf{X}^t)$ is the graph at time step t. The set of nodes $\mathcal{V}$ is fixed for all time steps. Each node has its consistent label across T time steps. $\mathbf{A}^t \in \mathbb{R}^{N\times N}$ is the adjacency matrix and $\mathbf{X}^t \in \mathbb{R}^{N\times d}$ is the node attribute matrix. Both $\mathbf{A}^t$ and $\mathbf{X}^t$ are different at different time steps. Given $\mathbb{G}$ and the labels of a subset of nodes $\mathcal{V}_L$, the goal of node classification in temporal attributed graphs is to classify the nodes in subset $\mathcal{V}_U$ whose labels are unknown, where $\mathcal{V} = \mathcal{V}_L \cup \mathcal{V}_U$.

3 Spatio-Temporal Attentive RNN

The framework of STAR is shown in Figure 1. It includes a spatio-temporal GRU and a dual attention module. First, we introduce the basic GRU. Then, we introduce the spatio-temporal GRU and discuss the dual attention mechanism.

3.1 Gated Recurrent Unit

The GRU network is an effective approach to capture the temporal patterns of sequential data. Given a sequence of input data $\mathbf{x}_1, ...\mathbf{x}_T \in \mathbb{R}^d$, a state vector $\mathbf{h}_t \in \mathbb{R}^{d_h}$ is calculated for each input data by applying the following equations iteratively.

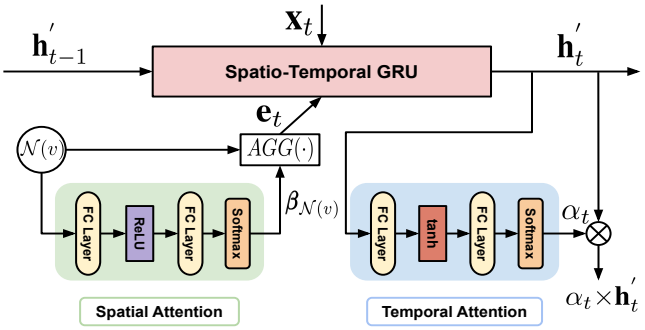


Figure 1: Overall architecture of STAR.

$$\mathbf{z}_t = \sigma(\mathbf{W}_z[\mathbf{x}_t \oplus \mathbf{h}_{t-1}] + \mathbf{b}_z), \quad (1)$$

$$\mathbf{r}_t = \sigma(\mathbf{W}_r[\mathbf{x}_t \oplus \mathbf{h}_{t-1}] + \mathbf{b}_r), \quad (2)$$

$$\tilde{\mathbf{h}}_t = \tanh(\mathbf{W}_h[\mathbf{x}_t \oplus (\mathbf{r}_t \odot \mathbf{h}_{t-1})] + \mathbf{b}_h), \quad (3)$$

$$\mathbf{h}_t = (\mathbf{1} - \mathbf{z}_t) \odot \mathbf{h}_{t-1} + \mathbf{z}_t \odot \tilde{\mathbf{h}}_t, \quad (4)$$

where $\mathbf{W}_z, \mathbf{W}_r, \mathbf{W}_h \in \mathbb{R}^{d_h\times(d+d_h)}$ and $\mathbf{b}_z, \mathbf{b}_r, \mathbf{b}_h \in \mathbb{R}^{d_h}$ are parameters. $\mathbf{z}_t$, $\mathbf{r}_t \in \mathbb{R}^{d_h}$ are called update, reset gates separately and their values are in the range of [0,1]. $\mathbf{h}_0 = \mathbf{0}$. $\oplus$ denotes concatenation operator. $\odot$ denotes element-wise multiplication. The concatenation of all the state vectors $[\mathbf{h}_1 \oplus \cdots \oplus \mathbf{h}_T]$ or the last state vector $\mathbf{h}_T$ is usually used as the representation for the whole sequence. GRU can be utilized to classify the nodes in temporal attributed graphs. The intuitive way is to first feed the node attributes at different time steps into GRU to generate the node representations. Then apply a classifier to the learned representations. However, the basic approach ignores the neighborhood information of the node. Next we will introduce how to jointly integrate the neighborhood information so that the graph structural information can also be encoded for node classification.

3.2 Vector Representation of The Neighborhood

We extract a neighborhood vector for each node at each time step to represent its neighborhood information. The key idea is to aggregate the neighbors' representations. We will consider K-hop neighbors.

First, we prepare the K-hop neighbors. Given a set of nodes $\mathcal{B}$ to be classified, we sample the immediate neighbors of the nodes in $\mathcal{B}$. These sampled neighbors and the nodes in $\mathcal{B}$ together form a new set $\mathcal{B}^1$. We do the same for $\mathcal{B}^1$ and get another new set $\mathcal{B}^2$. As a result, we can get a sequence of sets denoted by $\mathcal{B}^0$ ... $\mathcal{B}^K$, where $\mathcal{B}^0 = \mathcal{B}$. The neighbors are sampled by sampling function $\mathcal{N}(\cdot)$. The process is described in Algorithm 1.

Then, based on the sequence of node sets, we generate the neighborhood vectors for all the nodes in $\mathcal{B}$. It is described in Algorithm 2, where $\mathbf{g}^k_{t(v)}$ is the representation of node v after aggregating its k-th hop neighbors at time step t. The key idea is to apply aggregator $AGG_k(\cdot)$ to aggregate the neighbors' representations (Line 5) and concatenate the aggregation with

Algorithm 1: Preparing K-hop Neighbors

Input: Temporal attributed graph $\mathbb{G} = (\mathcal{G}^1, \mathcal{G}^2, ..., \mathcal{G}^T)$, A set of input nodes $\mathcal{B}$, Depth K
Output: K-hop neighbors at different time steps, $\mathcal{B}_t^0$ ... $\mathcal{B}_t^K$

1 **for** $t = 1, ..., T$ **do**
2 $\mathcal{B}_t^0 \leftarrow \mathcal{B}$
3 **for** $k = 1, ..., K$ **do**
4 $\mathcal{B}_t^k \leftarrow \mathcal{B}_t^{k-1}$
5 **for** $v \in \mathcal{B}_t^{k-1}$ **do**
6 $\mathcal{B}_t^k \leftarrow \mathcal{B}_t^k \cup \mathcal{N}(v)$

the node's representation (Line 6). The new representation is generated by a nonlinear transformation. $\mathbf{W}_{trans}^k \in \mathbb{R}^{d_g \times 2d_g}$ is the transformation matrix that we need to learn. We will introduce $AGG(\cdot)$ in Section 3.4.

3.3 Spatio-Temporal GRU

To integrate the neighborhood vector, a spatio-temporal GRU (ST-GRU) is proposed. The intuition behind ST-GRU is that we consider the neighborhood information besides the node attributes and the previous state vector when generating the new proposal to update the state vector (Eq. (8)). ST-GRU has two advantages. The state vectors explicitly contain both the temporal information of node attributes and the spatial information encoded in the neighborhoods. And it is practical to further implement a dual attention mechanism on both temporal and spatial aspects.

ST-GRU is described in Eqs. (5)-(9). Given a sequence of node attributes $\mathbf{x}_1$, ..., $\mathbf{x}_T \in \mathbb{R}^d$ and a sequence of neighborhood vectors $\mathbf{e}_1$, ..., $\mathbf{e}_T \in \mathbb{R}^{d_g}$, a state vector $\mathbf{h}_t^{'} \in \mathbb{R}^{d_h}$ is calculated for each time step by applying the following equations iteratively, where $\mathbf{h}_0^{'} = \mathbf{0}$.

$$\mathbf{z}_t^{'} = \sigma(\mathbf{W}_z^{'}[\mathbf{x}_t \oplus \mathbf{h}_{t-1}^{'} \oplus \mathbf{e}_t] + \mathbf{b}_z^{'}), \quad (5)$$

$$\mathbf{r}_t^{'} = \sigma(\mathbf{W}_r^{'}[\mathbf{x}_t \oplus \mathbf{h}_{t-1}^{'} \oplus \mathbf{e}_t] + \mathbf{b}_r^{'}), \quad (6)$$

$$\mathbf{s}_t^{'} = \sigma(\mathbf{W}_s^{'}[\mathbf{x}_t \oplus \mathbf{h}_{t-1}^{'} \oplus \mathbf{e}_t] + \mathbf{b}_s^{'}), \quad (7)$$

$$\tilde{\mathbf{h}}_t^{'} = \tanh(\mathbf{W}_h^{'}[\mathbf{x}_t \oplus (\mathbf{r}_t^{'} \odot \mathbf{h}_{t-1}^{'}) \oplus (\mathbf{s}_t^{'} \odot \mathbf{e}_t)] + \mathbf{b}_h^{'}), \quad (8)$$

$$\mathbf{h}_t^{'} = (\mathbf{1} - \mathbf{z}_t^{'}) \odot \mathbf{h}_{t-1}^{'} + \mathbf{z}_t^{'} \odot \tilde{\mathbf{h}}_t^{'}, \quad (9)$$

where $\mathbf{b}_z^{'}, \mathbf{b}_r^{'}, \mathbf{b}_s^{'}, \mathbf{b}_h^{'} \in \mathbb{R}^{d_h}$, $\mathbf{W}_s^{'} \in \mathbb{R}^{d_g \times (d+d_h+d_g)}$, $\mathbf{W}_z^{'}, \mathbf{W}_r^{'}$, $\mathbf{W}_s^{'} \in \mathbb{R}^{d_h \times (d+d_h+d_g)}$ are parameters. $\mathbf{z}_t^{'}, \mathbf{r}_t^{'} \in \mathbb{R}^{d_h}$, $\mathbf{s}_t^{'} \in \mathbb{R}^{d_g}$ are called update, reset, neighborhood gates respectively.

Eqs. (5)-(7) describe the calculation process of the three gates. The gates control the information when generating the state vector. The intuition of considering the node attributes, the previous state vector and the current neighborhood vector is that the current state vector is influenced by the current information of the node, the previous state vector and the current neighborhood information.

Eq. (8) describes the calculation process of a new proposal. The values in all gates are in the range of [0,1]. $\mathbf{r}_t^{'} \odot \mathbf{h}_{t-1}^{'}$ indicates how much information to keep from the previous state vector. $\mathbf{s}_t^{'} \odot \mathbf{e}_t$ indicates how much information to keep from its neighborhoods. Eq. (9) describes the calculation of a new state vector. $(\mathbf{1}\text{-}\mathbf{z}_t^{'}) \odot \mathbf{h}_{t-1}^{'}$ indicates how much information to throw away from the previous state vector. $\mathbf{z}_t^{'} \odot \tilde{\mathbf{h}}_t^{'}$ indicates how much new information to add from the new proposal.

Algorithm 2: Generating Neighborhood Vector

Input: Temporal attributed graph $\mathbb{G} = (\mathcal{G}^1, \mathcal{G}^2, ..., \mathcal{G}^T)$, K-hop neighbors $\mathcal{B}_t^0$ ... $\mathcal{B}_t^K$, where $\mathcal{B}_t^0 = \mathcal{B}$
Output: Neighborhood vector $\mathbf{e}_{t(v)}$ for all $v \in \mathcal{B}_t^0$

1 **for** $t = 1, ..., T$ **do**
2 $\mathbf{g}_{t(v)}^K \leftarrow \mathbf{x}_{t(v)}, \forall v \in \mathcal{B}_t^K$
3 **for** $k = K\text{-}1, ..., 1$ **do**
4 **for** $v \in \mathcal{B}_t^k$ **do**
5 $\mathbf{g}_{\mathcal{N}(t(v))}^{k+1} \leftarrow AGG_{k+1}(\{\mathbf{g}_{t(u)}^{k+1}, \forall u \in \mathcal{N}(v)\})$
6 $\mathbf{g}_{t(v)}^k \leftarrow \sigma(\boldsymbol{W}_{trans}^{k+1}[\mathbf{g}_{t(v)}^{k+1} \oplus \mathbf{g}_{\mathcal{N}(t(v))}^{k+1}])$
7 **for** $v \in \mathcal{B}_t^0$ **do**
8 $\mathbf{e}_{t(v)} \leftarrow AGG_1(\{\mathbf{g}_{t(u)}^1, \forall u \in \mathcal{N}(v)\})$

3.4 Dual Attention Mechanism

Spatial Attention

Different neighbors influence node presentations diversely. Attention technique is capable of adaptively capturing the pertinent information [Vaswani *et al.*, 2017; Shaw *et al.*, 2018]. We design a spatial attention module to detect the important neighbors of a node.

The spatial attention is applied to the aggregator during the aggregation process (Line 5 in Algorithm 2). Based on the attention values, the aggregator sums up the neighbors' representations as follows.

$$AGG_k(\{\mathbf{g}_{t(u)}^k, \forall u \in \mathcal{N}(v)\}) = \sum_{u \in \mathcal{N}(v)} \beta_u^k \mathbf{V}_k \mathbf{g}_{t(u)}^k, \quad (10)$$

where $\sum \beta_u^k = 1$ and $\mathbf{V}_k \in \mathbb{R}^{d_g \times d_g}$ are parameters. β_u^k is the attention value of neighbor u located at the k-th hop. It indicates the importance of u to node v compared to other neighbors located at the k-th hop. β_u^k is produced by our spatial attention module that takes the representations of the node and its neighbors as inputs, which is described as follows.

$$\beta_u^k = \frac{\exp\{F(\mathbf{w}_k^\top[\mathbf{V}_k \mathbf{g}_{t(u)}^k \oplus \mathbf{V}_k \mathbf{g}_{t(v)}^k])\}}{\sum_{v' \in \mathcal{N}(v)} \exp\{F(\mathbf{w}_k^\top[\mathbf{V}_k \mathbf{g}_{t(v')}^k \oplus \mathbf{V}_k \mathbf{g}_{t(v)}^k])\}}, \quad (11)$$

where $F(\cdot)$ is an activation function. $\mathbf{w}_k \in \mathbb{R}^{d_g}$ and $\mathbf{V}_k \in \mathbb{R}^{d_g \times d_g}$ are parameters.

Temporal Attention

For a temporal attributed graph $\mathbb{G} = (\mathcal{G}^1, \mathcal{G}^2, ..., \mathcal{G}^T)$, the amount of valuable information provided by different time steps is different. Only some contain the most discriminative information for determining node labels. Take brain graphs as an example. The nodes represent the tidy cubes of brain tissue [Avena-Koenigsberger *et al.*, 2018]. For the nodes related

Dataset	# Nodes	# Edges	# Attributes	# Time Steps	# Classes
Brain	5000	1955488	20	12	10
Reddit	8291	264050	20	10	4
DBLP-5	6606	42815	100	10	5
DBLP-3	4257	23540	100	10	3

Table 1: Description of the datasets

to language processing, the time during which the subject is speaking should have higher importance than the time when the subject does not speak. Based on this insight, a temporal attention module is designed to automatically pay different levels of attention to different time steps.

The temporal attention module takes the state vector $\mathbf{h}_t^{'}$ as input and outputs an attention value for it as follows.

$$\alpha_t = \frac{\exp\{\tilde{\mathbf{w}}^\top \tanh(\tilde{\mathbf{V}}\mathbf{h}_t^{'})\}}{\sum_{i=1}^{T}\exp\{\tilde{\mathbf{w}}^\top \tanh(\tilde{\mathbf{V}}\mathbf{h}_i^{'})\}}, \tag{12}$$

where $\tilde{\mathbf{w}} \in \mathbb{R}^{d_\alpha}$ and $\tilde{\mathbf{V}} \in \mathbb{R}^{d_\alpha \times d_h}$ are parameters. α_t indicates the importance of time step t for determining the target node's label compared to others. The concatenation of all the state vectors is denoted by

$$\mathbf{H} = [\mathbf{h}_1^{'} \oplus \cdots \oplus \mathbf{h}_T^{'}] \in \mathbb{R}^{T \times d_h}. \tag{13}$$

Thus, the attention values of all state vectors are

$$\boldsymbol{\alpha} = \text{softmax}(\tilde{\mathbf{w}}^\top \tanh(\tilde{\mathbf{V}}\mathbf{H}^\top)) \in \mathbb{R}^T. \tag{14}$$

Then we sum up all the state vectors scaled by $\boldsymbol{\alpha}$ to generate a vector representation for the node shown as follows.

$$\mathbf{q} = \boldsymbol{\alpha}^\top \mathbf{H} \in \mathbb{R}^{d_h}. \tag{15}$$

The output of one such attention unit usually focuses on one part of the temporal pattern of the node. However, it is possible that multiple parts together describe the overall pattern. Thus we need multiple attention units to focus on different parts. Suppose there are m parts needed to extract from the input. We use m $\tilde{\mathbf{w}}$'s and concatenate them as $\tilde{\mathbf{W}} = [\tilde{\mathbf{w}}_1 \oplus \cdots \oplus \tilde{\mathbf{w}}_m]$. The resulting attention value matrix is

$$\mathbf{A} = \text{softmax}(\tilde{\mathbf{W}}^\top \tanh(\tilde{\mathbf{V}}\mathbf{H}^\top)) \in \mathbb{R}^{m \times T}, \tag{16}$$

where softmax$(\cdot)$ performs on the 2*nd* dimension of its input. The final representation is then denoted by

$$\mathbf{Q} = \mathbf{A}\mathbf{H} \in \mathbb{R}^{m \times d_h}. \tag{17}$$

3.5 Objective Function

Given the node representations denoted by $\mathbf{Q}_1, \cdots, \mathbf{Q}_N$ and the node labels $\mathbf{y}_1, \cdots, \mathbf{y}_N$, where N is the number of nodes, the objective function of STAR is

$$J = L_{ce} + \lambda_1 P_{att} + \lambda_2 P_{nn}. \tag{18}$$

$L_{ce} = -\frac{1}{N}\sum_{i=1}^{N}\mathbf{y}_i \log(\tilde{\mathbf{y}}_i)$ is the cross-entropy loss. $\tilde{\mathbf{y}}_i$ is the estimate. It is produced by applying softmax$(\cdot)$ to the output of a fully connected layer that takes the node representation as input, i.e., $\tilde{\mathbf{y}}_i = \text{softmax}(\mathbf{W}_o\mathbf{Q}_i + \mathbf{b}_o)$. $\mathbf{W}_o \in \mathbb{R}^{c \times md_h}$ and $\mathbf{b}_o \in \mathbb{R}^c$ are parameters. c is the number of classes. $P_{att} = \|\mathbf{A}\mathbf{A}^\top - \mathbf{I}\|_F^2$ is the penalization term to encourage multiple attentions to diverge from each other. P_{nn} is the penalization term for the parameters to prevent STAR from over-fitting. λ_1 and λ_1 are the hyper-parameters.

Method	Models Temporal	Models Neighborhood	Handles Attribute	Applies Attention
DeepWalk	×	✓	×	×
node2vec	×	✓	×	×
GCN	×	✓	✓	×
GraphSAGE	×	✓	✓	×
LSTM	✓	×	✓	×
GRU	✓	×	✓	×
DynGEM	✓	✓	×	×
DynAERNN	✓	✓	×	×
STAR	✓	✓	✓	✓

Table 2: Comparison of baseline methods

3.6 Computational Analysis

STAR is local in time and the length of input sequence does not affect its storage requirements. The complexity of STAR per time step is proportional to the number of parameters. The parameters of STAR are from the spatio-temporal GRU and the dual attention mechanism. Specifically, the numbers of parameters for generating the neighborhood vector and updating the spatio-temporal cell are $2(K\text{-}1)d_g^2$ and $(d_g\text{+}3d_h)(d\text{+}d_g\text{+}d_h)\text{+}4d_h$ respectively. Because $d_g\text{=}d_h$ and $d\text{>}d_g$, the complexity of the spatio-temporal GRU per time step is $\mathcal{O}(Kd_g^2\text{+}dd_g)$. The number of parameters of the dual attention is $K(2d_g^2\text{+}d_g)\text{+}d_\alpha(m\text{+}d_g)$. Because $d_\alpha\text{=}d_g$ and $d_g\text{>}m$, the complexity of the dual attention is $\mathcal{O}(Kd_g^2)$. Thus, the complexity per time step of STAR is $\mathcal{O}(Kd_g^2\text{+}dd_g)$.

4 Experiments

4.1 Datasets and Baselines

We use four real datasets as shown in Table 1. In the Brain dataset, the nodes represent the tidy cubes of brain tissue and the edges indicate the connectivity [Preti *et al.*, 2017]. We apply PCA to the functional magnetic resonance imaging (fMRI) data[1] to generate node attributes. Two nodes are connected if they show similar degree of activation [Gonzalez-Castillo *et al.*, 2015]. In the Reddit dataset[2], the nodes represent the posts. Two posts are connected if they contain similar keywords. We apply the word2vec approach [Mikolov *et al.*, 2013] to the comments of a post to generate its node attributes [Hamilton *et al.*, 2017]. The DBLP-3 and DBLP-5 datasets are extracted from the bibliography website DBLP[3]. Both of them are the co-author graphs where nodes represent authors. The paper titles and abstracts are used to generate node attributes by word2vec. The authors in DBLP-3 and DBLP-5 are from three and five research areas respectively.

We compare STAR with the state-of-the-art baseline methods. The comparison between them is shown in Table 2. DeepWalk [Perozzi *et al.*, 2014], node2vec [Grover and Leskovec, 2016], GCN [Kipf and Welling, 2017] and GraphSAGE [Veličković *et al.*, 2018] are originally designed for static graphs. They can not model the temporal patterns of the sequence data. We first apply them to each time step to generate an node representation. Then we concatenate the

[1]https://tinyurl.com/y4hhw8ro

[2]https://www.reddit.com/

[3]https://dblp.uni-trier.de/

Method	Brain			DBLP-3			DBLP-5			Reddit		
	ACC	AUC	F1	ACC	AUC	F1	ACC	AUC	F1	ACC	AUC	F1
DeepWalk	71.4	97.2	70.2	49.7	60.1	50.5	35.4	61.0	26.9	47.5	71.9	46.8
node2vec	71.0	96.8	70.6	51.6	63.0	51.6	36.9	64.2	27.2	48.0	72.2	47.9
GCN	65.0	86.7	60.1	47.4	90.4	51.5	33.7	50.0	28.9	23.9	50.0	17.3
GraphSAGE	69.4	96.7	74.1	71.8	87.0	70.8	71.0	90.7	69.7	42.5	66.8	42.5
LSTM	83.6	98.6	84.6	81.9	92.5	81.7	74.1	91.4	74.1	40.2	66.5	40.6
GRU	81.6	98.6	82.2	82.5	93.7	83.2	75.6	91.5	75.2	42.1	67.2	41.9
DynGEM	71.0	97.2	70.2	52.3	59.0	52.8	31.6	54.6	9.9	39.9	66.2	41.5
DynAERNN	46.6	89.0	47.0	50.2	53.5	50.3	36.8	55.9	16.0	28.9	53.6	18.6
STAR-NH	84.7	98.4	86.1	83.1	94.4	83.5	76.6	92.2	75.9	42.3	67.1	42.1
STAR-TA	81.3	93.5	81.7	78.2	86.6	78.3	74.5	91.7	74.7	46.1	71.3	46.2
STAR-SA	79.5	90.2	79.9	78.3	86.5	79.6	72.1	88.5	72.6	44.6	68.0	44.4
STAR	**89.2**	**99.2**	**90.0**	**86.2**	**97.1**	**86.7**	**80.3**	**95.5**	**80.7**	**50.8**	**75.0**	**51.1**

Table 3: Node classification comparison (%)

representations of different time steps into a vector. The node labels are predicted by employing a logistic regression model on the vector. LSTM [Hochreiter and Schmidhuber, 1997], GRU [Cho *et al.*, 2014], DynGEM [Goyal *et al.*, 2018b] and DynAERNN [Goyal *et al.*, 2018a] can model the temporal patterns. LSTM and GRU only consider the evolution of node attributes. DynGEM and DynAERNN can model the neighborhood information.

To evaluate the effectiveness of different components of STAR, we study its variants. STAR-NH is the variant that generates node representations only based on node attributes. The temporal attention is applied. STAR-TA is the variant without applying the temporal attention. STAR-SA is the variant without applying the spatial attention. It generates the aggregation of the neighborhood by the mean pooling.

In our experiments, d_g, d_h and d_α are set to 10. K and m are set to 2 and 3 respectively. λ_1 and λ_2 are set to 10^{-3}. They are determined by grid-search from {1,2,5,10,15}, {1,2,3,4} and $\{0, 10^{-5}, 10^{-4}, 10^{-3}, 10^{-2}\}$ respectively. $\mathcal{N}(\cdot)$ is the uniform sampling function. The number of sampled neighbors is set to 4. LeakyReLU [Maas *et al.*, 2013] is adopted as $F(\cdot)$. We apply 10-fold cross-validation to evaluate all supervised methods. STAR is implemented with Tensorflow [Abadi *et al.*, 2016] and optimized by Adam [Kingma and Ba, 2014]. The code of STAR and the data used are publicly available[4].

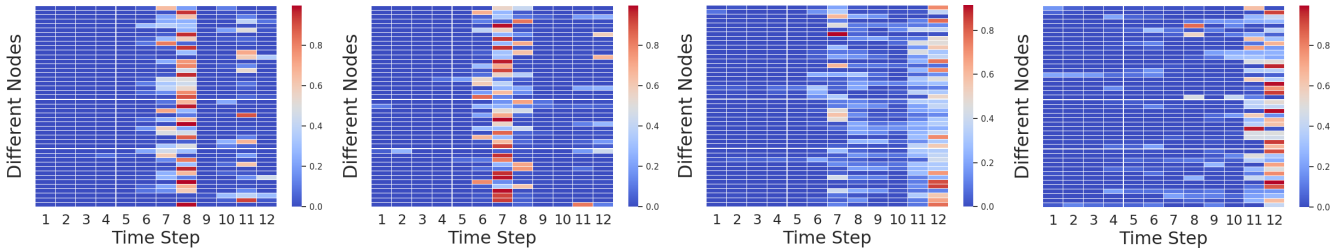
4.2 Experimental Results

Node classification. We evaluate the classification results by accuracy, AUC and F1 [Grover and Leskovec, 2016], which are shown in Table 3. It is observed that STAR achieves the best performance. DeepWalk and node2vec show good performance on the Reddit dataset. This is because the node labels of Reddit are largely dominated by the graph topology information, which is well utilized by DeepWalk and node2vec. LSTM and GRU show good performance on Brain, DBLP-3 and DBLP-5, where the node labels are dominated by node attributes information. STAR shows the best performance on all datasets, which indicates STAR utilizes well both graph topology and node attributes. Moreover, compared to DynGEM and DynAERNN, STAR shows a better ability of modeling the evolution patterns of temporal attributed graphs. STAR outperforms STAR-NH, indicating the importance of the spatial information of neighborhood on extracting better representations. By comparing STAR with STAR-TA and STAR-SA, we see the benefits of applying the dual attention mechanism.

Visualization of temporal attention. To investigate the interpretability of STAR, we visualize the attention values of different time steps. Figure 2 shows the results on four categories of nodes from the Brain dataset. It is observed that the nodes belonging to different categories show different kinds of attention distributions across time steps. This is because different categories of brain nodes show different degrees of activation when the subject conducts different tasks. For example, during time steps 7, 8 and 11, the subject mainly focuses on processing the audio information. As can be seen in Figure 2(a), the attention values of time steps 7, 8 and 11 are significantly larger for the nodes belonging to the category of auditory processing. Similar observations can be observed in Figures 2(b)-2(d). Larger values indicate that these time steps are more important for the node representations for classification. This demonstrates the effectiveness of the temporal attention mechanism. In addition, for Figure 2(a), the attention values of time steps 7, 8 and 11 are generated by three temporal attention units respectively. The same case exists for Figures 2(b)-2(d). This verifies the effectiveness of the multiple temporal attention mechanism.

Visualization of spatial attention. To gain further insight about the interpretability of STAR, we visualize the attention values of different neighbors. The results of two nodes from the Brain dataset, node 4023 and node 3266, are shown in Figure 3. Different node colors represent different node categories. Solid red and dotted gray lines represent higher and lower attention values respectively. Higher values indicate the neighbors are more important to the target node's representation. As can be seen in Figure 3, the target node is influenced more by the the neighbors from the same category. This is because the nodes from the same category play similar roles in the graph. For example, for node 4023, its immedi-

[4]https://tinyurl.com/y67ywq6j



(a) Auditory Processing (b) Language Processing (c) Emotion Processing (d) Body Movement

Figure 2: Overall attention values of different time steps for different categories of nodes.

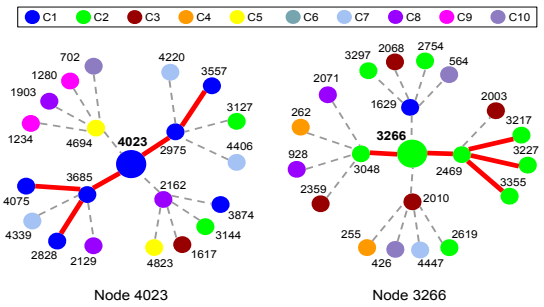
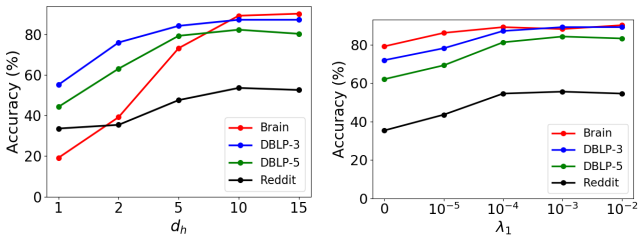


Figure 3: Visualization of spatial attention values. Solid red and dotted gray lines indicate higher and lower values respectively.



(a) Varying d_h (b) Varying λ_1

Figure 4: Parameter sensitive analysis.

ate neighbors with higher attention values are node 3685 and node 2975. The two nodes and node 4023 belong to the category of auditory processing. Similar observations can be observed for node 3266. This verifies that STAR is capable of detecting the important neighbors for node classification based on the spatial attention mechanism.

Parameter sensitivity. We study the sensitivity of STAR with respect to two parameters, the size of state vectors d_h and the parameter for the attention penalization term λ_1. The values of d_h and λ_1 are from $\{1,2,5,10,15\}$ and $\{0, 10^{-5}, 10^{-4}, 10^{-3}, 10^{-2}\}$ respectively. Other settings are the same as Section 4.1. Figs. 4(a)-4(b) summarize the results of d_h and λ_1 respectively. It is observed that as d_h or λ_1 increases, the classification accuracy increases until an optimal value. $d_h \in [10,15]$ and $\lambda_1 \in [10^{-4}, 10^{-2}]$ give the optimal results. Thus, it is reasonable to set them as 10 and 10^{-3} respectively. Moreover, the non-zero choices of λ_1 verify the importance of the attention penalization term in STAR in Eq. (18).

5 Related Work

In recent years, node classification has drawn extensive research attention [Perozzi *et al.*, 2014; Grover and Leskovec, 2016; Wang *et al.*, 2016; Kipf and Welling, 2017; Wang *et al.*, 2017b; Hamilton *et al.*, 2017]. Random walks are employed to extract the local information of a node in the graph. Nodes are further classified based on the local information preserved by word embedding techniques [Perozzi *et al.*, 2014]. A biased random walk is proposed to learn richer node representations by exploring diverse local information [Grover and Leskovec, 2016]. Recently, some interest has been focused on the use of attention mechanism for node classification [Veličković *et al.*, 2018]. These approaches are for the static graphs. Graphs in many real-world applications are dynamic. Both graph topology and node attributes evolve over time. Some recent progresses have been made on the node classification in the temporal graphs [Zhu *et al.*, 2018; Du *et al.*, 2018; Goyal *et al.*, 2018a; Ma *et al.*, 2018; Zuo *et al.*, 2018; Yu *et al.*, 2018]. The triadic closure process is used to model the evolution patterns of temporal graphs and a logistic regression model is used to classify nodes [Zhou *et al.*, 2018]. To make the node representations stable over time, the node representation at time t is generated from the one at time t-1 [Goyal *et al.*, 2018b]. SLIDE [Li *et al.*, 2018] uses a low-rank matrix to preserve all observed nodes in the graph where both structure and attributes change. However, none of these approaches can differentiate the relative importance of different factors that influence node representations.

6 Conclusion

In this paper, we propose a novel method, STAR, for node classification in temporal attributed graphs. STAR consists of a spatio-temporal GRU and a dual attention module. By feeding the sequential node attributes and the neighborhood representations into the GRU, the temporal features of attributes evolution and the spatial information of the node's local neighborhood can be effectively modeled respectively. A dual attention mechanism is developed to perform a thorough analysis on the interpretability of STAR. And it helps STAR detect the time steps and the node neighbors that are more important for the classification. Extensive experimental results demonstrate the effectiveness of STAR.

Acknowledgements

This work was partially supported by the National Science Foundation grant IIS-1707548.

References

[Abadi *et al.*, 2016] Martín Abadi, Paul Barham, Jianmin Chen, Zhifeng Chen, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Geoffrey Irving, Michael Isard, et al. Tensorflow: a system for large-scale machine learning. In *OSDI*, volume 16, pages 265–283, 2016.

[Avena-Koenigsberger *et al.*, 2018] Andrea Avena-Koenigsberger, Bratislav Misic, and Olaf Sporns. Communication dynamics in complex brain networks. *Nature Reviews Neuroscience*, 19(1):17, 2018.

[Cho *et al.*, 2014] Kyunghyun Cho, Bart Van Merriënboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. Learning phrase representations using rnn encoder-decoder for statistical machine translation. In *EMNLP*, 2014.

[Du *et al.*, 2018] Lun Du, Yun Wang, Guojie Song, Zhicong Lu, and Junshan Wang. Dynamic network embedding: An extended approach for skip-gram based network embedding. In *IJCAI*, pages 2086–2092, 2018.

[Gonzalez-Castillo *et al.*, 2015] Javier Gonzalez-Castillo, Colin W Hoy, Daniel A Handwerker, Meghan E Robinson, Laura C Buchanan, Ziad S Saad, and Peter A Bandettini. Tracking ongoing cognition in individuals using brief, whole-brain functional connectivity patterns. *PNAS*, 112(28):8762–8767, 2015.

[Goyal *et al.*, 2018a] Palash Goyal, Sujit Rokka Chhetri, and Arquimedes Canedo. dyngraph2vec: Capturing network dynamics using dynamic graph representation learning. *arXiv preprint arXiv:1809.02657*, 2018.

[Goyal *et al.*, 2018b] Palash Goyal, Nitin Kamra, Xinran He, and Yan Liu. Dyngem: Deep embedding method for dynamic graphs. *arXiv preprint arXiv:1805.11273*, 2018.

[Grover and Leskovec, 2016] Aditya Grover and Jure Leskovec. node2vec: Scalable feature learning for networks. In *SIGKDD*, pages 855–864. ACM, 2016.

[Hamilton *et al.*, 2017] Will Hamilton, Zhitao Ying, and Jure Leskovec. Inductive representation learning on large graphs. In *NeurIPS*, pages 1024–1034, 2017.

[Hochreiter and Schmidhuber, 1997] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997.

[Kingma and Ba, 2014] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[Kipf and Welling, 2017] Thomas N Kipf and Max Welling. Semi-supervised classification with graph convolutional networks. In *ICLR*, 2017.

[Li *et al.*, 2018] Jundong Li, Kewei Cheng, Liang Wu, and Huan Liu. Streaming link prediction on dynamic attributed networks. In *WSDM*, pages 369–377. ACM, 2018.

[Ma *et al.*, 2018] Yao Ma, Ziyi Guo, Zhaochun Ren, Eric Zhao, Jiliang Tang, and Dawei Yin. Streaming graph neural networks. *arXiv preprint arXiv:1810.10627*, 2018.

[Maas *et al.*, 2013] Andrew L Maas, Awni Y Hannun, and Andrew Y Ng. Rectifier nonlinearities improve neural network acoustic models. In *ICML*, volume 30, page 3, 2013.

[Mikolov *et al.*, 2013] Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*, 2013.

[Ni *et al.*, 2018] Jingchao Ni, Shiyu Chang, Xiao Liu, Wei Cheng, Haifeng Chen, Dongkuan Xu, and Xiang Zhang. Co-regularized deep multi-network embedding. In *WWW*, pages 469–478. International World Wide Web Conferences Steering Committee, 2018.

[Perozzi *et al.*, 2014] Bryan Perozzi, Rami Al-Rfou, and Steven Skiena. Deepwalk: Online learning of social representations. In *SIGKDD*, pages 701–710. ACM, 2014.

[Preti *et al.*, 2017] Maria Giulia Preti, Thomas AW Bolton, and Dimitri Van De Ville. The dynamic functional connectome: State-of-the-art and perspectives. *Neuroimage*, 160:41–54, 2017.

[Shaw *et al.*, 2018] Peter Shaw, Jakob Uszkoreit, and Ashish Vaswani. Self-attention with relative position representations. *arXiv preprint arXiv:1803.02155*, 2018.

[Vaswani *et al.*, 2017] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *NeurIPS*, pages 5998–6008, 2017.

[Veličković *et al.*, 2018] Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Lio, and Yoshua Bengio. Graph attention networks. 2018.

[Wang *et al.*, 2016] Daixin Wang, Peng Cui, and Wenwu Zhu. Structural deep network embedding. In *KDD*, pages 1225–1234. ACM, 2016.

[Wang *et al.*, 2017a] Shen Wang, Lifang He, Bokai Cao, Chun-Ta Lu, Philip S Yu, and Ann B Ragin. Structural deep brain network mining. In *SIGKDD*, pages 475–484. ACM, 2017.

[Wang *et al.*, 2017b] Xiao Wang, Peng Cui, Jing Wang, Jian Pei, Wenwu Zhu, and Shiqiang Yang. Community preserving network embedding. In *AAAI*, 2017.

[Yu *et al.*, 2018] Wenchao Yu, Wei Cheng, Charu C Aggarwal, Kai Zhang, Haifeng Chen, and Wei Wang. Netwalk: A flexible deep embedding approach for anomaly detection in dynamic networks. In *KDD*, pages 2672–2681. ACM, 2018.

[Zhou *et al.*, 2018] Lekui Zhou, Yang Yang, Xiang Ren, Fei Wu, and Yueting Zhuang. Dynamic network embedding by modeling triadic closure process. In *AAAI*, 2018.

[Zhu *et al.*, 2018] Dingyuan Zhu, Peng Cui, Ziwei Zhang, Jian Pei, and Wenwu Zhu. High-order proximity preserved embedding for dynamic networks. *TKDE*, 30(11):2134–2144, 2018.

[Zuo *et al.*, 2018] Yuan Zuo, Guannan Liu, Hao Lin, Jia Guo, Xiaoqian Hu, and Junjie Wu. Embedding temporal network via neighborhood formation. In *KDD*, pages 2857–2866. ACM, 2018.