Structural Embedding Pre-Training for Deep Temporal Graph Learning

Meng Liu, Wenxuan Tu, Ke Liang, Xinwang Liu*

（刘猛，涂文轩，梁科，刘新旺）

National University of Defense Technology

Oct. 2023

CCHI 2023

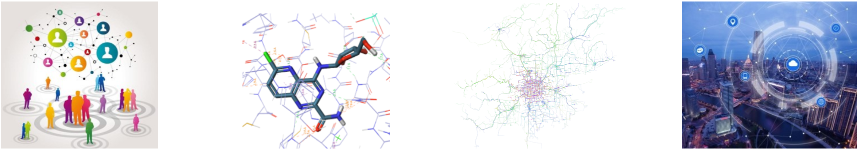
Research Group

Supervisor: Prof. Xinwang Liu has been awarded the NSFC for Distinguished Young Scholar in 2023 / Excellent Youth in 2019 (国家杰青/优青), and the World's Top 2% Scientists. He has made significant contributions on fusion mechanism of multi-view clustering, incomplete multi-view clustering and deep clustering analysis, etc. Based on his research works, he has published more than 100 top papers such as TPAMI, TKDE, ICML and NeurIPS, etc. He has been cited more than 12000 citations, and 12 papers are ESI Highly Cited Papers. He is also on the editorial board of TNNLS, TCYB, and Information Fusion, etc. He served as Area Chair or Senior PC member for top conferences such as ICML and NeurIPS for more than 20 times. He has also received the First-Grade Natural Science Award of Hunan Province twice. Homepage: https://xinwangliu.github.io/

About Me: Meng Liu is a Ph.D. student at NUDT, was advised by Prof. Xinwang Liu. His research interests include Temporal Graph Learning and Deep Clustering. He has published several papers including SIGIR, ACM MM, CIKM, CCHI and BIB, which have received more than 150 citations. He serves as the reviewer for TNNLS, PR, TOMM, NeurIPS, AAAI, and ACM MM, etc.

Contact me: mengliuedu@163.com (Email) / 15138964863 (Wechat).

Graph Deep Learning



(a) Social Network (b) Protein Graph (c) Smart City (d) Internet of Things

Figure 1: Real-Life Application Scenarios of Graph Learning.

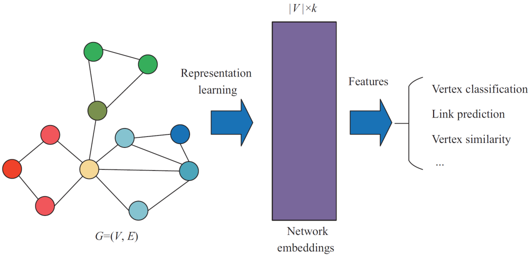


Figure 2: Converting Graph Structure to Node Embeddings.

Temporal Graph

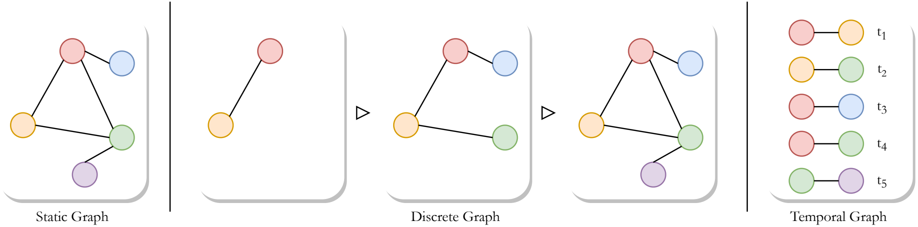
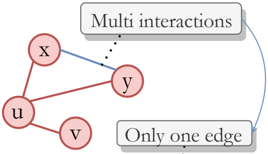


Figure 3: Static Graph, Discrete Dynamic Graph, and Temporal Graph.

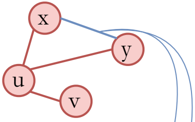
* Static Graphs record graph structure mainly in adjacency matrices without node and edge changes.
* Discrete Dynamic Graphs intercept graph changes at fixed intervals, and each intercepted static snapshot can be regarded as a static graph, with a temporal order between multiple snapshots.
* Temporal Graphs, also known as continuous-time dynamic graphs, discard the adjacency matrix and instead use interaction sequence to record node interactions.

Temporal Graph



	x	v	y	u
x	0	0	1	1
v	0	0	0	1
y	1	0	0	1
u	1	1	1	0

Adjacency Matrix in Static Graph



Node	Node	Time
x	y	Mon
x	u	Mon
x	y	Wen
u	v	Thu
x	y	Sat
y	u	Sun

Interaction Sequence in Temporal Graph

Figure 4: Adjacency Matrix and Interaction Sequence.

Strengths

* More flexible batch training pattern.
* More complete information storage.
* Stricter chronological input order.

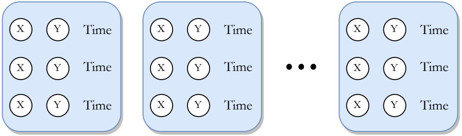
Weaknesses

* Classical adjacency matrix-based module is difficult to adapt.
* Higher-order structure information is difficult to capture.

Motivation

Temporal methods can hardly obtain higher-order structural information, which is limited by the interaction sequence-based structure and the batch processing-based pattern.

The absence of higher-order structural information limits the receptive field of the models.



Interaction Sequence-Based Batch Processing

Thus we ask: can this hard-to-get information be fed into the model in advance by means of pre-training?

To mitigate the computational complexity, we propose a novel method SET, which can "set up" a pre-training process for higher-order structural information to achieve data augmentation for temporal graph methods.

Problem Definition

Temporal Graphs

Given a temporal graph $G = (V, E, T, X)$, we define V as the node set, E as the interaction set, T as the timestamp set, and X as the initial feature set.

One interaction can be formulated as (x, y, t), where x denotes the source node, y denotes the target node, and t denotes the interaction time.

Learning Objective

Temporal graph learning aims to capture important information in the temporal graph to generate embedding for each node.

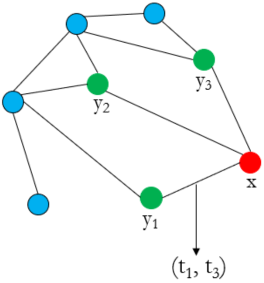
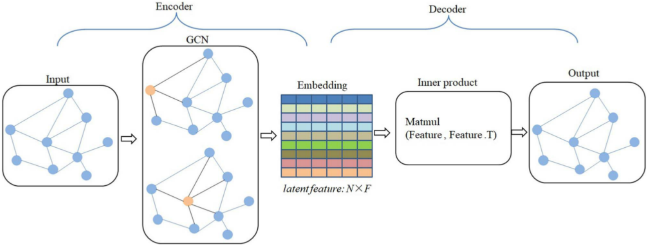


Figure 5: One Edge, Multiple Interactions.

Pre-Training with GAE



We introduce the Graph AutoEncoder (GAE) as the pre-training module to generate structural embeddings as initial features, which can be divided into encoder and decoder.

$$H = f_E(A, X, W^E), \qquad Z = f_D(H, W^D) \tag{1}$$

After obtain the final node embeddings Z, GAE utilizes them to reconstruct the adjacency matrix $\hat{A}$ by the sigmoid function σ.

$$\hat{A} = \sigma(Z, Z^T, W^A), \qquad L_{rec} = \min_{W^E, W^D, W^A} MSE(\hat{A}, A) \tag{2}$$

Training with HTNE

For temporal embedding training, we utilize the classic temporal graph method HTNE as our baseline model. Given an interaction (x, y, t), HTNE calculate their conditional intensity $\lambda_{x,y,t}$ as follows.

$$\lambda_{x,y,t} = \mu_{x,y} + h_{x,y,t} \tag{3}$$

$$h_{x,y,t} = \sum_{i \in N_x} \mu_{i,y} \cdot a_{i,x} \cdot f(t - t_i) \tag{4}$$

Here $\mu_{x,y} = -||z_x - z_y||^2$ denotes the basic intensity, $h_{x,y,t}$ denotes the Hawkes increment intensity. The loss function of HTNE can be formulated as follows.

$$L_{tem} = -\log \sigma(\lambda_{x,y,t}) - \sum_{k \sim P_x} \log \sigma(1 - \lambda_{x,k,t}) \tag{5}$$

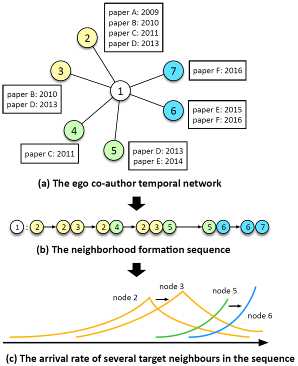


Figure 6: Concept of HTNE.

Enhanced Loss Function

During model training, the embeddings of nodes are updated, resulting in deviations from structural embeddings.

$$L_{align} = -||z_x^0 - z_x||^2 - ||z_y^0 - z_y||^2 \tag{6}$$

Therefore, we set up additional loss functions that encourage constantly updated time embeddings Z to align with the initialized structural embeddings Z^0.

We also introduce the power scaling error for SET. The intuition of the power scaling error is that for dimensions with small differences, they do not require much optimization, but should focus on optimizing those dimensions with large differences.

$$L = \sum_E (L_{tem} + L_{align})^\gamma \tag{7}$$

By magnifying these differences to power levels, the model is more likely to optimize where it is not already optimized. γ is a hyper-parameter to adjust the power level.

Experimental Setting

Table 1: Description of The Datasets.

Datasets	DBLP	Brain	BITotc	AMms
# Nodes	28,085	5,000	5,881	74,526
# Interactions	236,894	1,955,488	35,592	89,689
# Timestamps	27	12	27,487	5,082
# Class	10	10	21	5
# Type	Academic	Bioinformatic	Financial	E-commerce

Datasets

We construct experiments on several datasets from different areas. DBLP is a co-author graph, which contains different computer science domains. Brain is a connectivity graph of brain tissue in humans. BITotc is the transaction records of the Bitcoin exchange platform OTC. AMms is magazine subscription graph on Amazon website.

Tasks

We conduct two tasks: node classification and node clustering. We also study the parameter sensitivity of the enhanced loss parameter γ.

Node Classification

Table 2: Node Classification Performance on all datasets. Note that the optimal results are highlighted in bold, and the sub-optimal results are underlined.

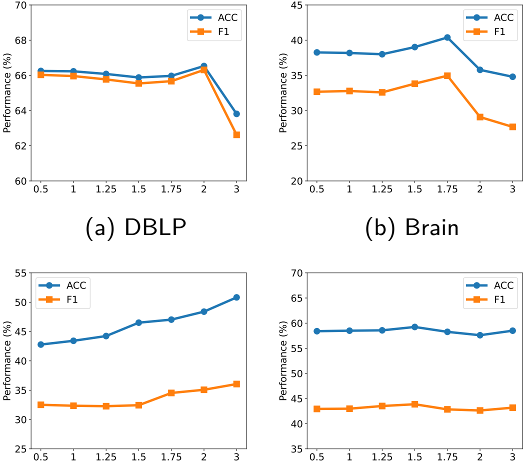
Datasets	DBLP		Brain		BITotc		AMms	
	ACC	F1	ACC	F1	ACC	F1	ACC	F1
AE (2006)	43.17	40.44	25.12	23.15	27.43	28.11	42.41	41.59
DeepWalk (2014)	62.76	62.35	34.54	32.97	42.65	33.81	57.91	43.12
node2vec (2016)	63.87	63.48	38.72	33.17	44.23	32.47	57.93	43.10
GAE (2016)	64.89	<u>65.46</u>	38.16	32.84	41.17	32.59	57.09	42.77
SDCN (2020)	60.56	59.65	37.64	32.60	39.55	30.97	<u>58.51</u>	43.19
HTNE (2018)	<u>65.47</u>	65.32	33.94	29.11	38.84	32.22	57.96	43.04
JODIE (2019)	58.76	55.93	<u>39.93</u>	33.68	<u>47.52</u>	<u>34.41</u>	55.34	42.47
TGN (2020)	55.67	54.52	25.48	24.15	36.27	29.35	58.50	43.19
MNCI (2021)	65.88	65.41	38.54	34.35	44.93	33.61	58.48	<u>43.20</u>
TREND (2022)	61.28	59.88	39.85	<u>34.44</u>	32.14	27.32	58.43	43.16
SET (ours)	**66.53**	**66.31**	**40.38**	**34.95**	**48.39**	**35.07**	**59.24**	**43.87**

Node Clustering

Table 3: Node Clustering Performance on DBLP and Brain datasets. Note that the optimal results are highlighted in bold, and the sub-optimal results are underlined.

Datasets	DBLP				Brain			
	ACC	NMI	ARI	F1	ACC	NMI	ARI	F1
AE	42.16	<u>36.71</u>	22.54	37.84	43.48	<u>50.49</u>	<u>29.78</u>	43.26
DeepWalk	28.95	22.03	13.73	24.79	41.28	49.09	28.40	42.54
node2vec	46.31	34.87	20.40	43.35	<u>43.92</u>	45.96	26.08	<u>46.61</u>
GAE	39.31	29.75	17.17	35.04	31.22	32.23	14.97	34.11
SDCN	46.69	35.07	23.74	40.31	42.62	46.61	27.93	41.42
HTNE	45.74	35.95	22.13	<u>43.98</u>	43.20	50.33	29.26	43.85
JODIE	20.79	11.67	11.32	13.23	19.14	10.50	5.00	11.12
TGN	19.78	9.82	5.46	10.66	17.40	8.04	4.56	13.49
MNCI	<u>46.85</u>	36.28	<u>23.40</u>	42.57	40.42	43.58	26.74	39.63
TREND	25.36	14.25	6.24	19.89	39.83	45.64	22.82	33.67
SET	**48.48**	**39.47**	**24.21**	**44.93**	**44.96**	**50.97**	**30.08**	**47.14**

Parameter Sensitivity



(a) DBLP (b) Brain

(c) BITotc (d) AMms

Figure 7: Parameter sensitivity study of different γ values on all datasets.

Discussion

Contributions

* We propose a temporal graph learning method called SET, which introduces structural embedding pre-training to enhance temporal graph learning.
* To enhance the effectiveness of deep learning training, we impose constraints on the loss function that focus more on optimizing those dimensions with large variance.
* We compare SET with multiple baseline methods on different datasets, and the results validate the effectiveness and performance of our method.

Future Direction

* Large model, large graph.
* Graph for science and engineering.
  - Dynamic social computing in smart city.
  - Open-world multi-agent cooperative control.
  - Real-time multi-sensor information fusion.

Thanks for Listening!