BENCHTEMP: A General Benchmark for Evaluating Temporal Graph Neural Networks

Authors' Response to Reviewer VXfQ

Opportunities For Improvement:

W1. My major concern is that the adopted datasets are relatively small-scaled, with a maximum number of nodes no more than 100,000. This limitation may impact the potential impact of the benchmark.
W2. The paper claims to contribute to the unification of pipelines for existing temporal GNNs, but it is worth noting that there are already libraries available, such as PyG Temporal and TGL (VLDB'22), which address this aspect, albeit without providing benchmarks. It would be helpful to clarify how the proposed method differs from these libraries or whether they can be used together.
W3. The benchmark code page could benefit from improved documentation. For example, it is not immediately clear how users can utilize the benchmark to test their own temporal GNNs.
W4. While the main focus of the paper is on temporal GNNs, it would be valuable to discuss whether the proposed benchmark can be used for discrete-time dynamic GNNs as well.

General Response:

We appreciate your great feedback! We have included new datasets with up to several million edges and nodes. We have carefully through your comments and added *six* datasets (eBay-Small, eBay-Large, Taobal-Large, DGraphFin, YouTubeReddit-Small, YouTubeReddit-Large), including *four* **large-scale** datasets (eBay-Large, Taobao-Large, DGraphFin, YouTubeReddit-Large). We have reported the corresponding experiments and detailed discussions in the updated paper. The eBay datasets are a collection of the user transactions on **eBay's e-commerce platform**. We thank our industrial collaborator for sharing their datasets in our research. Considering user privacy and security, eBay datasets could only be shared among collaborators. Any researchers who are interested in the eBay datasets, please email our team (jonnyhuanghnu@gmail.com).

For easy access, all datasets have been hosted on the open-source platform zenodo (`https://zenodo.org/`) with a Digital Object Identifier (DOI) 10.5281/zenodo.8267846 (`https://zenodo.org/record/8267846`).

We have clarified that the difference between BenchTeMP and existing libraries. Furthermore, We illustrate how users can utilize BenchTeMP to test their own temporal GNNs. BenchTeMP can be used for discrete-time dynamic GNNs as well.

We provide our response to each individual comment below:

Submitted to the 37th Conference on Neural Information Processing Systems (NeurIPS 2023) Track on Datasets and Benchmarks. Do not distribute.

Comment 1

W1. My major concern is that the adopted datasets are relatively small-scaled, with a maximum number of nodes no more than 100,000. This limitation may impact the potential impact of the benchmark.

Response:

We appreciate the reviewer for the suggestions! We have included new datasets with up to several million edges and nodes. We have added *six* datasets (eBay-Small, eBay-Large, Taobal-Large, DGraphFin, YouTubeReddit-Small, YouTubeReddit-Large), including *four* **large-scale** datasets (eBay-Large, Taobao-Large, DGraphFin, YouTubeReddit-Large). The statistics of the new datasets are shown in Table 1. For easy access, all datasets have been hosted on the open-source platform zenodo (`https://zenodo.org/`) with a Digital Object Identifier (DOI) 10.5281/zenodo.8267846 (`https://zenodo.org/record/8267846`). In our paper, we present BenchTemp, a general benchmark for evaluating temporal graph neural network (TGNN) models over a wide range of tasks and settings. We extensively compare representative TGNN models on the benchmark datasets, regarding different tasks, settings, metrics, and especially model efficiency - **inference time.**

- **eBay-Small** is a subset of the eBay-Large dataset. We sample 38,427 nodes and 384,677 edges from eBay-Large graph according to edge timestamps.
- **YouTubeReddit-Small** is a collection of massive visual contents on YouTube and long-term community activity on Reddit. This dataset covers a **3**-month period from January to March 2020. Each row in the dataset represents a YouTube video v_i being shared in a subreddit s_j by some user u_k at time t [1]. Nodes are YouTube videos and subreddits, edges are the users' interactions between videos and subreddits. This dynamic graph has 264,443 nodes and 297,732 edges.
- **eBay-Large** is a million-scale dataset consisting of 1.3 million nodes and 1.1 million edges, which comprises the selected transaction records from the eBay e-commerce platform over a two-month period. eBay-Large is modeled as a user-item graph, where items are heterogeneous entities which include information such as phone numbers, addresses, and email addresses associated with a transaction. We selecte one month of transactions as seed nodes and then expand each seed node two hops back in time to enrich the topology while maintaining consistency in the distribution of seed nodes.
- **DGraphFin** is a collection of large-scale dynamic graph datasets, consisting of interactive objects, events and labels that evolve with time.It is a directed, unweighted dynamic graph consisting of millions of nodes and edges, representing a realistic user-to-user social network in financial industry. Nodes are users, and an edge from one user to another means that the user regards the other user as the emergency contact person [2].
- **Youtube-Reddit-Large** dataset covers **54** months of YouTube video propagation history from January 2018 to June 2022 [1]. This dataset has 5,724,111 nodes and 4,228,523 edges.

Table 1: Dataset statistics of the new datasets.

	Domain	*# Nodes*	*# Edges*
eBay-Small	E-commerce	38,427	384,677
YouTubeReddit-Small [1]	Social	264,443	297,732
eBay-Large	E-commerce	1,333,594	1,119,454
DGraphFin [2]	E-commerce	3,700,550	4,300,999
Youtube-Reddit-Large [1]	Social	5,724,111	4,228,523
Taobao-Large [3, 4]	E-commerce	1,630,453	5,008,745

Table 2: ROC AUC results of new datasets on the *dynamic link prediction task*. The best and second-best results are highlighted as **bold red** and underlined blue. We do not highlight the second-best if the gap is > 0.05 compared with the best result.

Dataset \ Model	JODIE	DyRep	TGN	TGAT	CAWN	NeurTW	NAT
				Transductive			
eBay-Small	0.9946 ± 0.0002	0.9941 ± 0.0006	0.9984 ± 0.0003	0.9838 ± 0.0006	0.9985 ± 0.0	**0.9991 ± 0.0**	0.9978 ± 0.0003
YouTubeReddit-Small	0.8519 ± 0.0007	0.8499 ± 0.0012	0.8432 ± 0.0032	0.8441 ± 0.0014	0.7586 ± 0.0031	**0.9003 ± 0.0031**	0.8259 ± 0.005
eBay-Large	0.9614 ± 0.0	0.9619 ± 0.0001	0.9642 ± 0.0003	0.5311 ± 0.0003	0.9442 ± 0.0003	0.9608 ± 0.0	**0.9658 ± 0.0002**
DGraphFin	0.8165 ± 0.0024	0.8171 ± 0.0016	**0.8683 ± 0.0023**	0.6112 ± 0.0165	0.5466 ± 0.0103	0.8611 ± 0.0035	0.8258 ± 0.0001
Youtube-Reddit-Large	0.8532 ± 0.0003	0.8529 ± 0.0006	0.8458 ± 0.0025	0.8536 ± 0.0026	0.7466 ± 0.0012	**0.916 ± 0.0025**	0.8605 ± 0.0009
Taobao-Large	0.7726 ± 0.0005	0.7724 ± 0.001	0.8464 ± 0.0008	0.5567 ± 0.0047	0.7771 ± 0.0068	**0.859 ± 0.0091**	0.8188 ± 0.001
				Inductive			
eBay-Small	0.9696 ± 0.0007	0.9674 ± 0.0018	0.9913 ± 0.0004	0.9698 ± 0.0006	0.9964 ± 0.0001	0.9982 ± 0.0	**0.9998 ± 0.0001**
YouTubeReddit-Small	0.7582 ± 0.0003	0.7545 ± 0.0009	0.7276 ± 0.0033	0.7436 ± 0.0006	0.7533 ± 0.0016	0.8978 ± 0.0032	**0.9876 ± 0.0049**
eBay-Large	0.7536 ± 0.0014	0.7515 ± 0.0006	0.7657 ± 0.0026	0.5224 ± 0.0003	0.9459 ± 0.0001	0.9608 ± 0.0	**0.9999 ± 0.0001**
DGraphFin	0.6884 ± 0.0051	0.6876 ± 0.001	0.6439 ± 0.0089	0.5677 ± 0.0184	0.5479 ± 0.009	**0.8635 ± 0.0021**	0.7955 ± 0.0201
Youtube-Reddit-Large	0.7539 ± 0.0005	0.7554 ± 0.0003	0.7243 ± 0.0016	0.7501 ± 0.0019	0.7327 ± 0.0016	0.9128 ± 0.0031	**0.9863 ± 0.006**
Taobao-Large	0.7075 ± 0.0009	0.7042 ± 0.0006	0.6812 ± 0.0032	0.5222 ± 0.0041	0.7787 ± 0.0103	0.869 ± 0.010	**0.9933 ± 0.0008**
				Inductive New-Old			
eBay-Small	0.9862 ± 0.0003	0.9836 ± 0.0016	0.9947 ± 0.0009	0.9712 ± 0.002	0.9985 ± 0.0	0.9988 ± 0.0	**0.9999 ± 0.0**
YouTubeReddit-Small	0.7695 ± 0.001	0.7655 ± 0.0018	0.7396 ± 0.0034	0.7242 ± 0.0004	0.7573 ± 0.0022	0.922 ± 0.0002	**0.9967 ± 0.0014**
eBay-Large	0.6109 ± 0.0244	0.5906 ± 0.0087	0.8134 ± 0.0105	0.6363 ± 0.0605	0.9569 ± 0.0007	0.8973 ± 0.0	**1.0 ± 0.0**
DGraphFin	0.5768 ± 0.0071	0.5735 ± 0.0007	0.5564 ± 0.0021	0.5742 ± 0.013	0.5646 ± 0.0244	0.7702 ± 0.0043	**0.8693 ± 0.0066**
Youtube-Reddit-Large	0.7844 ± 0.0015	0.7894 ± 0.0017	0.7623 ± 0.0031	0.7457 ± 0.0062	0.7511 ± 0.0022	0.9356 ± 0.0004	**0.9958 ± 0.0025**
Taobao-Large	0.7023 ± 0.0015	0.6953 ± 0.0022	0.6771 ± 0.0055	0.5104 ± 0.0106	0.7674 ± 0.005	0.8458 ± 0.0043	**0.9965 ± 0.0005**
				Inductive New-New			
eBay-Small	0.9388 ± 0.0009	0.9366 ± 0.0037	0.9838 ± 0.0007	0.9556 ± 0.0007	0.9937 ± 0.0	0.9975 ± 0.0	**0.9997 ± 0.0004**
YouTubeReddit-Small	0.7436 ± 0.0015	0.7436 ± 0.0018	0.7265 ± 0.0055	0.749 ± 0.0011	0.7479 ± 0.004	0.864 ± 0.0071	**0.9868 ± 0.0049**
eBay-Large	0.7526 ± 0.0013	0.7500 ± 0.0005	0.7639 ± 0.0027	0.5196 ± 0.0002	0.9542 ± 0.0003	0.9615 ± 0.0	**0.9999 ± 0.0001**
DGraphFin	0.7307 ± 0.0007	0.7323 ± 0.0002	0.6843 ± 0.0131	0.5649 ± 0.0248	0.5417 ± 0.0099	**0.9051 ± 0.0028**	0.7584 ± 0.0323
Youtube-Reddit-Large	0.6932 ± 0.0026	0.7022 ± 0.0007	0.6703 ± 0.0024	0.7269 ± 0.0	0.6942 ± 0.0028	0.8716 ± 0.0077	**0.9796 ± 0.0103**
Taobao-Large	0.7243 ± 0.0001	0.7247 ± 0.0001	0.6885 ± 0.0024	0.5256 ± 0.0054	0.7922 ± 0.0118	0.8906 ± 0.0088	**0.9969 ± 0.0002**

- **Taobao-Large** is a collection of the Taobao user behavior dataset intercepted based on the period 8:00 to 18:00 on 26 November 2017 [4]. Nodes are users and items, and edges are behaviors between users and items, such as favor, click, purchase, and add an item to shopping cart. This public dataset has 1,630,453 nodes and 5,008,74 user-item interaction edges.

A Experiments

We conduct extensive experiments on the tasks of *dynamic link prediction* and *dynamic node classification.* The experimental setup is the same as in the paper.

A.1 Link Prediction Task

We run the link prediction task on 7 TGNN models and the new datasets under different settings (Transductive, Inductive, Inductive New-Old, and Inductive New-New). The AUC and AP results for each new datasets are shown in Table 2 and Table 3, respectively. For the four large-scale datasets (eBay-Large, Taobao-Large, DGraphFin, YouTubeReddit-Large), we observe the similar results as in the paper. Specifically, NAT and NeurTW achieve the top-2 performance on almost all datasets under transductive and inductive settings.

A.2 Node Classification Task

The eBay-Small and eBay-Large datasets have node labels, so we conduct dynamic node classification experiments on both the eBay-Small and eBay-Large datasets. The AUC results are shown in Table 4. We can observe the similar results as in the paper. NeurTW achieves the best performance on both eBay-Small and eBay-Large datasets. NAT performs poorly on the node classification task.

Table 3: AP results of new datasets on the *dynamic link prediction task*. The best and second-best results are highlighted as **bold red** and underlined blue. We do not highlight the second-best if the gap is > 0.05 compared with the best result.

Dataset \ Model	JODIE	DyRep	TGN	TGAT	CAWN	NeurTW	NAT
Transductive							
eBay-Small	0.9938 ± 0.0004	0.9936 ± 0.0006	0.9983 ± 0.0003	0.9819 ± 0.0009	0.9981 ± 0.0	**0.9991 ± 0.0**	0.9975 ± 0.0002
YouTubeReddit-Small	0.8612 ± 0.0009	0.8594 ± 0.0012	0.8421 ± 0.0041	0.8515 ± 0.0012	0.7625 ± 0.0042	**0.9112 ± 0.0021**	0.8325 ± 0.0068
eBay-Large	0.9318 ± 0.0002	0.9322 ± 0.0002	0.9357 ± 0.0006	0.5239 ± 0.0002	0.9144 ± 0.0004	0.9307 ± 0.0	**0.9398 ± 0.0004**
DGraphFin	0.7705 ± 0.0009	0.7705 ± 0.0024	0.8571 ± 0.0009	0.6441 ± 0.0123	0.5431 ± 0.0095	**0.8637 ± 0.0014**	0.7956 ± 0.0012
Youtube-Reddit-Large	0.8622 ± 0.0007	0.8632 ± 0.0004	0.8476 ± 0.0022	0.8591 ± 0.0026	0.7475 ± 0.0017	**0.9222 ± 0.0013**	0.8628 ± 0.0015
Taobao-Large	0.7164 ± 0.0003	0.7142 ± 0.0008	0.844 ± 0.0011	0.5761 ± 0.0023	0.7616 ± 0.0069	**0.8568 ± 0.016**	0.7904 ± 0.0008
Inductive							
eBay-Small	0.9638 ± 0.0007	0.9619 ± 0.0017	0.9898 ± 0.0005	0.9675 ± 0.0007	0.9953 ± 0.0002	0.9982 ± 0.0	**0.9998 ± 0.0001**
YouTubeReddit-Small	0.7866 ± 0.0007	0.7833 ± 0.0009	0.7387 ± 0.0069	0.7551 ± 0.0002	0.7568 ± 0.0031	0.9086 ± 0.0022	**0.9872 ± 0.0056**
eBay-Large	0.6989 ± 0.0018	0.6973 ± 0.0007	0.7096 ± 0.0030	0.518 ± 0.0002	0.9174 ± 0.0001	0.9308 ± 0.0	**0.9999 ± 0.0001**
DGraphFin	0.6563 ± 0.002	0.6567 ± 0.0009	0.624 ± 0.006	0.5866 ± 0.0123	0.5428 ± 0.0082	**0.8626 ± 0.0012**	0.7053 ± 0.0185
Youtube-Reddit-Large	0.7796 ± 0.0009	0.7818 ± 0.0009	0.73 ± 0.0029	0.7587 ± 0.0025	0.7353 ± 0.0022	0.9192 ± 0.0022	**0.9849 ± 0.0071**
Taobao-Large	0.6763 ± 0.0011	0.6746 ± 0.0011	0.6664 ± 0.0012	0.5315 ± 0.0027	0.7533 ± 0.011	0.8596 ± 0.0205	**0.9941 ± 0.0007**
Inductive New-Old							
eBay-Small	0.9849 ± 0.0007	0.9836 ± 0.0013	0.9931 ± 0.0008	0.9682 ± 0.0028	0.9985 ± 0.0001	0.999 ± 0.0	**0.9999 ± 0.0**
YouTubeReddit-Small	0.7963 ± 0.0013	0.7937 ± 0.0014	0.729 ± 0.0086	0.7296 ± 0.0013	0.762 ± 0.0041	0.9244 ± 0.0015	**0.9966 ± 0.0016**
eBay-Large	0.5670 ± 0.0186	0.5870 ± 0.0074	0.8024 ± 0.0060	0.6504 ± 0.0385	0.9592 ± 0.0008	0.8458 ± 0.0	**1.0 ± 0.0**
DGraphFin	0.6005 ± 0.0048	0.5872 ± 0.0059	0.5753 ± 0.0062	0.5927 ± 0.0058	0.5669 ± 0.0269	0.7572 ± 0.0025	**0.8184 ± 0.0088**
Youtube-Reddit-Large	0.808 ± 0.0014	0.8142 ± 0.0019	0.7472 ± 0.0043	0.7526 ± 0.0097	0.7553 ± 0.0025	0.9368 ± 0.0009	**0.9953 ± 0.0028**
Taobao-Large	0.7009 ± 0.0013	0.698 ± 0.0014	0.6879 ± 0.0008	0.5254 ± 0.0074	0.7597 ± 0.0053	0.8459 ± 0.0103	**0.9969 ± 0.0004**
Inductive New-New							
eBay-Small	0.923 ± 0.001	0.9226 ± 0.0024	0.98 ± 0.0007	0.9505 ± 0.0009	0.991 ± 0.0001	0.9973 ± 0.0	**0.9997 ± 0.0004**
YouTubeReddit-Small	0.7578 ± 0.0015	0.7582 ± 0.0021	0.7564 ± 0.0043	0.7718 ± 0.0023	0.7498 ± 0.004	0.8868 ± 0.0034	**0.9861 ± 0.0063**
eBay-Large	0.6976 ± 0.0016	0.6957 ± 0.0007	0.7078 ± 0.0031	0.5154 ± 0.0001	0.93 ± 0.0003	0.9318 ± 0.0	**0.9999 ± 0.0001**
DGraphFin	0.6802 ± 0.0005	0.6811 ± 0.0002	0.6526 ± 0.0098	0.5831 ± 0.0184	0.5379 ± 0.0071	**0.8977 ± 0.0014**	0.6529 ± 0.0249
Youtube-Reddit-Large	0.7038 ± 0.0024	0.7115 ± 0.0007	0.6979 ± 0.002	0.7414 ± 0.0012	0.6965 ± 0.004	0.8848 ± 0.0023	**0.9761 ± 0.0134**
Taobao-Large	0.6738 ± 0.0005	0.6742 ± 0.0005	0.6611 ± 0.0011	0.53 ± 0.0023	0.7521 ± 0.0127	0.8738 ± 0.0145	**0.9973 ± 0.0001**

Table 4: ROC AUC results for the *dynamic node classification task* on the eBay datasets. The top-2 results are highlighted as **bold red** and underlined blue.

Dataset \ Model	JODIE	DyRep	TGN	TGAT	CAWN	NeurTW	NAT
eBay-Small	0.9274 ± 0.0017	0.8677 ± 0.0356	0.913 ± 0.0025	0.9342 ± 0.0002	0.9305 ± 0.0001	**0.9529 ± 0.0002**	0.6797 ± 0.0115
eBay-Large	0.7244 ± 0.0002	0.7246 ± 0.0	0.6586 ± 0.0129	0.672 ± 0.0016	0.7710 ± 0.0002	**0.7859 ± 0.0**	0.5304 ± 0.0011

A.3 Efficiency

Considering many real world applications and , we add **the inference time** metric to evaluate the efficiency of models. The inference time comparison per 100,000 edges is shown in Figure 1. According to the figure, we can observe the similar model efficiency results as in the paper. In terms of the inference time, JODIE, DyRep, TGN and TGAT are faster, while CAWN and NeurTW are much slower. NAT is relatively faster than temporal walk-based methods through caching and parallelism optimizations, *achieving a good trade-off between model quality and efficiency*.

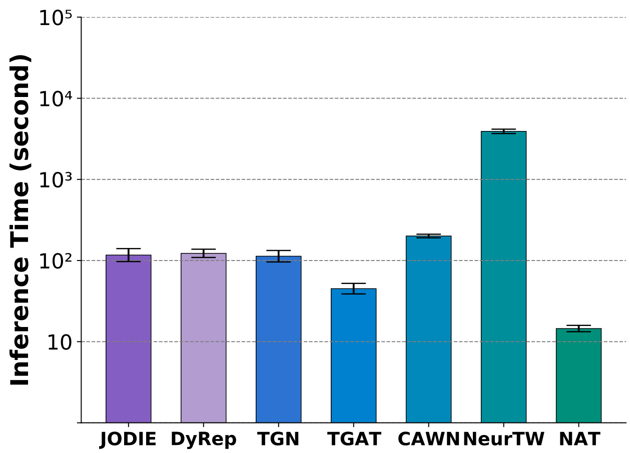

Figure 1: Inference time comparison per 100,000 edges.

Comment 2

W2. The paper claims to contribute to the unification of pipelines for existing temporal GNNs, but it is worth noting that there are already libraries available, such as PyG Temporal and TGL (VLDB'22), which address this aspect, albeit without providing benchmarks. It would be helpful to clarify how the proposed method differs from these libraries or whether they can be used together.

Response:

We appreciate this suggestion! We have carefully through your comments and compare BenchTeMP with existing libraries or methods, such as PyG Temporal [5] and TGL [6].

PyG Temporal [5] is a temporal graph neural network extension library for PyTorch Geometric. PyG Temporal consists of state-of-the-art deep learning and parametric learning methods to process spatio-temporal signals. PyG Temporal provides constant time difference graph neural networks on dynamic and static graphs. Users can build their own models based on PyG Temporal. BenchTeMP proposed in this paper is a benchmark library, which consists of *benchmark dataset*, *DataLoader*, *EdgeSampler*, *Evaluator*, *EarlyStopMonitor*, and *Leaderboard*. For evaluating TGNN models onto the same ground and compares them comprehensively, BenchTeMP provided a unified benchmark pipeline shown in Figure 4 in the paper for users to evaluate their own models constructed by PyG Temporal [5]. Of course, BenchTeMP and PyG Temporal can be used together. For example, users can construct their own models by PyG Temporal framework, and evaluate the performance of models by BenchTeMP.

TGL [6] comprises five main components, *a temporal sampler, a mailbox, a node memory module, a memory updater*, and *a message passing engine*. Considering the memory module and memory updater components, TGL [6] is a framework for memory-based TGNN models, such as JODIE [7], TGAT [8], TGN [9], and APAN [10]. However, BenchTeMP is a general benchmark framework for evaluating, no matter memory-based TGNNs (JODIE, DyRep, TGAT, TGN), TGNNs based on motifs (CAWN[11], NeurTW [3]), or even TGNNs based on joint-neighborhood operation (NAT [12]). Furthermore, BenchTeMP provides diverse workloads for evaluating TGNNs, regarding different tasks (dynamic link prediction and dynamic node classification), settings (transductive, inductive, New-Old, and New-New), metrics, and efficiency (runtime, running memory, inference time).

Comment 3

W3. The benchmark code page could benefit from improved documentation. For example, it is not immediately clear how users can utilize the benchmark to test their own temporal GNNs.

Response:

Thanks for your valuable suggestion! We have updated the benchmark code page (`https://github.com/qianghuangwhu/benchtemp`)and add a Section - *Usage Example*. In future work, we will continuously update our benchmark code page for users. Contributions and issues from the community are eagerly welcomed, with which we can together push forward the TGNN research.

Comment 4

W4. While the main focus of the paper is on temporal GNNs, it would be valuable to discuss whether the proposed benchmark can be used for discrete-time dynamic GNNs as well.

Response:

Thanks for this valuable comment! Of course, BenchTeMP proposed by us can can be used for discrete-time dynamic GNNs as well!

For example, DySAT [13] is a discrete-time dynamic graph model [9]. DySAT treats temporal interactions as graph snapshots. The input of DySAT are snapshot at each time step, and the output of DySAT model is node representations at each time step. For link prediction task, Using the node embeddings trained on graph snapshots up to time step t, single-step link prediction predicts the connections between nodes at time step $t + 1$. Exactly the same as the RandEdgeSampler of BenchTeMP, DySAT also performs negative sampling operation.

For example, the code of negative sampling in DySAT as follows, See file `https://github.com/aravindsankar28/DySAT/blob/master/utils/preprocess.py` for code details:

```
# python3
# Create train edges.
train_edges_false = []
while len(train_edges_false) < len(train_edges):
    idx_i = np.random.randint(0, adj.shape[0])
    idx_j = np.random.randint(0, adj.shape[0])
    if idx_i == idx_j:
        continue
    if ismember([idx_i, idx_j], edges_all):
        continue
    if ismember([idx_j, idx_i], edges_all):
        continue
    if train_edges_false:
        if ismember([idx_j, idx_i], np.array(train_edges_false)):
            continue
        if ismember([idx_i, idx_j], np.array(train_edges_false)):
            continue
    train_edges_false.append([idx_i, idx_j])
```

The code of RandEdgeSampler in BenchTeMP below, See file (`https://github.com/qianghuangwhu/benchtemp/blob/master/lp/edgesampler.py`) for code details:

```
# BenchTeMP RandEdgeSampler
class RandEdgeSampler:
    def __init__(self, src_list, dst_list, seed=None):
```

```
        self.seed = None
        self.src_list = np.unique(src_list)
        self.dst_list = np.unique(dst_list)

        if seed is not None:
            self.seed = seed
            self.random_state = np.random.RandomState(self.seed)

    def sample(self, size):
        if self.seed is None:
            src_index = np.random.randint(0, len(self.src_list), size)
            dst_index = np.random.randint(0, len(self.dst_list), size)
        else:

            src_index = self.random_state.randint(0, len(self.src_list
    ), size)
            dst_index = self.random_state.randint(0, len(self.dst_list
    ), size)
        return self.src_list[src_index], self.dst_list[dst_index]

    def reset_random_state(self):
        self.random_state = np.random.RandomState(self.seed)
```

Besides, the data format of datasets and data loading operation of DySAT are exactly the same as BenchTeMP.

Therefore, by the above analysis, we can conclude that BenchTeMP proposed by us can can be used for discrete-time dynamic graph models as well.

References

[1] Yiqiao Jin, Yeon-Chang Lee, Kartik Sharma, Meng Ye, Karan Sikka, Ajay Divakaran, and Srijan Kumar. Predicting information pathways across online communities. *arXiv preprint arXiv:2306.02259*, 2023.

[2] Xuanwen Huang, Yang Yang, Yang Wang, Chunping Wang, Zhisheng Zhang, Jiarong Xu, Lei Chen, and Michalis Vazirgiannis. Dgraph: A large-scale financial dataset for graph anomaly detection. *Advances in Neural Information Processing Systems*, 35:22765–22777, 2022.

[3] Ming Jin, Yuan-Fang Li, and Shirui Pan. Neural temporal walks: Motif-aware representation learning on continuous-time dynamic graphs. In *Advances in Neural Information Processing Systems*, 2022.

[4] Han Zhu, Xiang Li, Pengye Zhang, Guozheng Li, Jie He, Han Li, and Kun Gai. Learning tree-based deep model for recommender systems. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 1079–1088, 2018.

[5] Benedek Rozemberczki, Paul Scherer, Yixuan He, George Panagopoulos, Alexander Riedel, Maria Astefanoaei, Oliver Kiss, Ferenc Beres, , Guzman Lopez, Nicolas Collignon, and Rik Sarkar. PyTorch Geometric Temporal: Spatiotemporal Signal Processing with Neural Machine Learning Models. In *Proceedings of the 30th ACM International Conference on Information and Knowledge Management*, page 4564–4573, 2021.

[6] Hongkuan Zhou, Da Zheng, Israt Nisa, Vasileios Ioannidis, Xiang Song, and George Karypis. Tgl: a general framework for temporal gnn training on billion-scale graphs. *Proceedings of the VLDB Endowment*, 15(8):1572–1580, 2022.

[7] Srijan Kumar, Xikun Zhang, and Jure Leskovec. Predicting dynamic embedding trajectory in temporal interaction networks. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 1269–1278, 2019.

[8] Da Xu, Chuanwei Ruan, Evren Korpeoglu, Sushant Kumar, and Kannan Achan. Inductive representation learning on temporal graphs. In *International Conference on Learning Representations*, 2020.

[9] Emanuele Rossi, Ben Chamberlain, Fabrizio Frasca, Davide Eynard, Federico Monti, and Michael Bronstein. Temporal graph networks for deep learning on dynamic graphs. *arXiv preprint arXiv:2006.10637*, 2020.

[10] Xuhong Wang, Ding Lyu, Mengjian Li, Yang Xia, Qi Yang, Xinwen Wang, Xinguang Wang, Ping Cui, Yupu Yang, Bowen Sun, et al. Apan: Asynchronous propagation attention network for real-time temporal graph embedding. In *Proceedings of the 2021 international conference on management of data*, pages 2628–2638, 2021.

[11] Yanbang Wang, Yen-Yu Chang, Yunyu Liu, Jure Leskovec, and Pan Li. Inductive representation learning in temporal networks via causal anonymous walks. In *International Conference on Learning Representations*, 2021.

[12] Yuhong Luo and Pan Li. Neighborhood-aware scalable temporal network representation learning. In *The First Learning on Graphs Conference*, 2022.

[13] Aravind Sankar, Yanhong Wu, Liang Gou, Wei Zhang, and Hao Yang. Dysat: Deep neural representation learning on dynamic graphs via self-attention networks. In *Proceedings of the 13th international conference on web search and data mining*, pages 519–527, 2020.