BenchTemp: A General Benchmark for Evaluating Temporal Graph Neural Networks

Authors' Response to Reviewer RjkE

Opportunities For Improvement:

O1. The discussion regarding the results of the evaluation experiments is shallow. In particular, the high accuracy and speed of NAT for link prediction, as already reported in NAT's paper, do not provide novelty. Also, there is a limited specific discussion on how dataset differences impact accuracy. It would be more interesting if the authors could demonstrate new insights that are not easily inferred from existing papers.
O2. The discussion of the experimental results is mainly limited to the aspects listed in Table 1 (Memory, Attention, RNN, TempWalk, Scalability), so they lack detailed analysis. For example, CAWN and NeurTW both leverage TempWalk, but the experiments in Table 3 demonstrate that these two have different strengths and weaknesses on different datasets. Therefore, there should be more discussion on what specific differences in the characteristics between CAWN and NeurTW contribute to these strengths and weaknesses. Similar discussions should be made for other methods to show the correlations between the design of existing methods and the datasets. Regarding the statement "NeurTW introduces a continuous-time operation that can depict evolution trajectory, which is potentially suitable for CanParl with a large time granularity", the meaning is unclear due to the lack of a sufficient description of the correlation between the evolution trajectory and large time granularity.
O3. While the difficulty of the inductive New-New setting task is understandable, however, to say "Nevertheless, CAWN, NeurTW, and NAT still perform well due to their structure-aware techniques" is not sufficient. More detailed explanations of the structure-aware techniques should be provided.
O4. Regarding the efficiency evaluation, it may be desirable to discuss the elapsed time for users (i.e., per epoch time \ average number of epochs). Then, it would be beneficial to break down the analysis using the per epoch time and average number of epochs, respectively. Also, the paper only reports the execution time and memory usage without conducting in-depth analyses of their correlations with the design of each existing method.
O5. Node classification becomes more challenging as the number of labels increases, but the paper only handles binary classification, limiting the benchmark's generality. Performance benchmarks for tasks with multiple label numbers are desired.
O6. The statement "MOOC is relatively denser, and the temporal walk mechanism can effectively perceive local structures" is interesting, however, there is insufficient evidence to support the claim. Can the paper demonstrate that the effectiveness of the temporal walk mechanism changes in response to changes in graph density?
O7. The proposed technique, TeMP, is only briefly described in seven lines, making it difficult to fully understand its features.

General Response:

Thanks for the valuable suggestion! We have updated Section 4.4 in the paper (`https://openreview.net/pdf?id=rnZm2vQq31`). We provide summaries of the state-of-the-art results, the limitations of existing methods, and discuss future directions of TGNN research.

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

We have updated Section 4.4 in the paper (`https://openreview.net/pdf?id=rnZm2vQq31`) for the discussion differences between CAWN [1] and NeurTW [2]. We have updated the sentence in Section 4.2 of the paper (`https://openreview.net/pdf?id=rnZm2vQq31`) and added more detailes of the structure-aware techniques.

We have added **the inference time** metric to evaluate the efficiency of methods. We have added experiments of the node classification task with multiple label numbers. We have demonstrated that the effectiveness of the CAWN based on temporal walk mechanism changes in response to changes in graph density. We have added the details of TeMP in Appendix.

We provide our response to each individual comment below:

> **Comment 1**
>
> **O1.** The discussion regarding the results of the evaluation experiments is shallow. In particular, the high accuracy and speed of NAT for link prediction, as already reported in NAT's paper, do not provide novelty. Also, there is a limited specific discussion on how dataset differences impact accuracy. It would be more interesting if the authors could demonstrate new insights that are not easily inferred from existing papers.

Response:

We thank the reviewer for the suggestions! We have updated Section 4.4 in the paper (`https://openreview.net/pdf?id=rnZm2vQq31`). We provide summaries of the state-of-the-art results, the limitations of existing methods, and discuss future directions of TGNN research.

For example, the high accuracy and speed of NAT [3] for dynamic link prediction task are already reported in NAT's paper. However, NAT's original paper did not perform experiments for dynamic node classification task. We implement the dynamic node classification task of NAT, the experimental results are shown in Table 5 in the paper (`https://openreview.net/pdf?id=rnZm2vQq31`) and reveal that *NAT performs poorly on the node classification task. The node classification task does not rely on structural features as much as the link prediction task, so that the joint neighborhood mechanism of NAT may be less effective.*

Besides, In the original paper of NeurTW [2], the efficiency of NeurTW is not discussed. In BenchTeMP proposed by us, we evaluated NeurTW with diverse workloads, including performances and efficiency (**runtime** in the paper, **running memory** in the paper, **inference time** shown in Figure 1 of this response file). We reveal that *NeurTW performs poorly on efficiency.*

Furthermore, previous works conduct experiments on datasets with a small number of nodes and edges. Thus, in this response file, 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. 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.

After a discussion with our industrial partner eBay, we are working on sharing the **eBay-Small** and **eBay-Large** datasets in a way that ensures availability and justifies the research purpose. We provide a Google form for the applicants: `https://forms.gle/bP1RmyVJ1C6pgyS66` (**the applicants can remain anonymous**).

The experimental results on large-scale datasets (eBay-Large, Taobao-Large, DGraphFin, YouTubeReddit-Large) may be more convincing. Furthermore, we have added **Average Rank** metric for ranking model performances on the newly added large-scale datasets (eBay-Large, Taobao-Large, DGraphFin, YouTubeReddit-Large) to evaluate TGNN models on dynamic link prediction task and node classification task shown in Table 2 and Table 4.

- **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 [4]. 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 [5].

- **Youtube-Reddit-Large** dataset covers **54** months of YouTube video propagation history from January 2018 to June 2022 [4]. This dataset has 5,724,111 nodes and 4,228,523 edges.

- **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 [6]. 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 `https://openreview.net/pdf?id=rnZm2vQq31`.

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.

Table 1: Dataset statistics of the newly added datasets.

	Domain	*# Nodes*	*# Edges*
eBay-Small	E-commerce	38,427	384,677
YouTubeReddit-Small [4]	Social	264,443	297,732
eBay-Large	E-commerce	1,333,594	1,119,454
DGraphFin [5]	E-commerce	3,700,550	4,300,999
Youtube-Reddit-Large [4]	Social	5,724,111	4,228,523
Taobao-Large [2, 6]	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. **Average Rank** are computed by the experimental results of models on four large-scale datasets (eBay-Large, Taobao-Large, DGraphFin, YouTubeReddit-Large). 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
Average Rank	4.5	4.5	2.75	5.75	6	2.25	2.25
	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**
Average Rank	4	4.5	5.5	6.25	4.75	1.75	1.25
	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**
Average Rank	4.25	5	5.5	5.75	4.25	2.25	1
	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**
Average Rank	5	4.25	5.5	5.75	4.5	1.75	1.25
	JODIE	DyRep	TGN	TGAT	CAWN	NeurTW	NAT
Total Rank	4.44	4.56	4.81	5.88	4.88	2.00	**1.44**

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.

A.3 Efficiency - the inference time

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 have almost the same efficiency, while NeurTW are much slower. TGAT achieves the second-best efficiency. NAT is relatively faster than temporal walk-based methods through caching and parallelism optimizations, *achieving a good trade-off between model quality and efficiency*.

A.4 Efficiency - Runtime, RAM, GPU

We have added model efficiency results for the newly added datasets as follows. We will add all these results to the Appendix (`https://openreview.net/attachment?id=rnZm2vQq31&name=supplementary_material`).

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
Average Rank	4	4.5	5.5	3.5	2.5	**1**	7

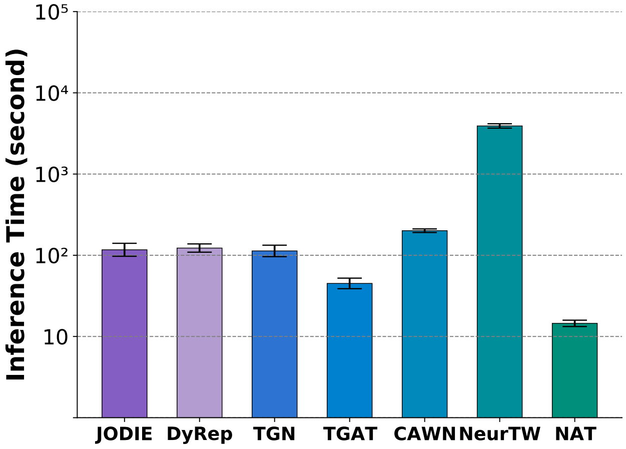

Figure 1: Inference time comparison per 100,000 edges.

Since many real-world graphs are extremely large, we believe efficiency is a vital issue for TGNNs in practice. We thereby compare the efficiency of the evaluated models on the newly added datasets (eBay-Small, eBay-Large, Taobal-Large, DGraphFin, YouTubeReddit-Small, YouTubeReddit-Large), and present the results for dynamic link prediction task in Table 5, while dynamic node classification task Table 6.

The Runtime in Table 5 and Table 6 shows that NAT is always trained much faster than the others and need a low RAM and GPU Memory. TGAT obtains the second-best efficiency performance on the newly added datasets. JODIE, DyRep, TGN achieve similar efficiency performance. We observe similar results as the main paper, NeurTW performs poorly on model efficiency.

Table 5: Model efficiency for the newly added datasets on *the link prediction task*. We report seconds per epoch as **Runtime**, the maximum RAM usage as **RAM**, and the maximum GPU memory usage as **GPU Memory**, respectively. The best and second-best results are highlighted as **bold red** and underlined blue.

	Runtime (second)						
Model Dataset	JODIE	DyRep	TGN	TGAT	CAWN	NeurTW	NAT
eBay-Small	749.80	801.58	905.19	61.05	1,385.54	1,556.32	**25.12**
YouTubeReddit-Small	213.92	227.99	214.17	85.59	378.94	7,459.92	**29.51**
eBay-Large	28,203.53	30,151.18	30,286.88	791.86	52,116.62	58,540.48	**117.38**
DGraphFin	4,579.52	4,210.48	4,397.32	1,708.71	30,144.25	81,653.89	**904.38**
Youtube-Reddit-Large	4,630.49	4,935.05	4,635.91	1,852.67	8,202.50	161,476.80	**638.77**
Taobao-Large	3,108.45	2,931.87	2,860.83	2,658.34	12,143.02	148,922.55	**6654.56**
	RAM (GB)						
eBay-Small	7.8	6.2	6.8	**4.3**	9.1	7.8	**4.3**
YouTubeReddit-Small	6.8	7.2	6.6	5.3	13.1	8.1	**4.5**
eBay-Large	20.2	18.3	19.1	5.2	17.1	10.1	**5.5**
DGraphFin	17.5	15.3	17.5	8.3	23.2	24.3	**6.9**
Youtube-Reddit-Large	26.3	16.6	18.9	7.9	18.5	21.3	**6.3**
Taobao-Large	14.3	12.1	13.4	7.5	18.1	20.7	**6.2**
	GPU Memory (GB)						
eBay-Small	2.0	1.9	2.0	1.9	1.8	**1.6**	2.2
YouTubeReddit-Small	1.3	1.4	2.1	1.3	1.8	**1.1**	**1.1**
eBay-Large	29.7	24.6	30.9	5.8	5.7	**3.0**	5.9
DGraphFin	19.3	18.5	16.1	6.3	6.9	6.1	**6.0**
Youtube-Reddit-Large	22.1	23.0	23.4	7.8	**6.3**	7.2	7.1
Taobao-Large	20.3	21.8	19.6	7.7	7.3	6.8	**5.6**

Table 6: Model efficiency for the newly added datasets on *the node classification task*. We report seconds per epoch as **Runtime**, the maximum RAM usage as **RAM**, and the maximum GPU memory usage as **GPU Memory**, respectively. The best and second-best results are highlighted as **bold red** and underlined blue.

	Runtime (second)						
Model Dataset	JODIE	DyRep	TGN	TGAT	CAWN	NeurTW	NAT
eBay-Small	765.05	794.03	718.56	55.05	226.56	583.08	**13.05**
eBay-Large	29,153.28	29,867.17	27,028.53	629.52	8,522.04	25,693.71	**97.54**
	RAM (GB)						
eBay-Small	6.5	6.8	6.7	4.2	6.9	7.2	**4.1**
eBay-Large	41.8	39.2	20.5	**5.2**	15.1	7.4	5.8
	GPU Memory (GB)						
eBay-Small	1.8	**1.2**	1.5	1.8	1.9	1.8	2.3
eBay-Large	31.7	31	31.4	5.8	5.8	**2.9**	5.9

Comment 2

O2. The discussion of the experimental results is mainly limited to the aspects listed in Table 1 (Memory, Attention, RNN, TempWalk, Scalability), so they lack detailed analysis. For example, CAWN and NeurTW both leverage TempWalk, but the experiments in Table 3 demonstrate that these two have different strengths and weaknesses on different datasets. Therefore, there should be more discussion on what specific differences in the characteristics between CAWN and NeurTW contribute to these strengths and weaknesses. Similar discussions should be made for other methods to show the correlations between the design of existing methods and the datasets. Regarding the statement "NeurTW introduces a continuous-time operation that can depict evolution trajectory, which is potentially suitable for CanParl with a large time granularity", the meaning is unclear due to the lack of a sufficient description of the correlation between the evolution trajectory and large time granularity.

Comment 2

Response:

Thanks for the valuable suggestion! We have updated Section 4.4 in the paper (`https://openreview.net/pdf?id=rnZm2vQq31`) for discussing the differences between CAWN [1] and NeurTW [2].

CAWN [1] and NeurTW [2] perform well on the link prediction task and are both based on motifs and index anonymization operation. However, NeurTW [2] additionally constructs neural ordinary differential equations (NODEs). With a component based on neural ordinary differential equations, the extracted motifs allow for irregularly-sampled temporal nodes to be embedded explicitly over *multiple different interaction time intervals*, enabling the effective capture of the underlying spatiotemporal dynamics.

In NeurTW [2], The *Continuous Evolution* operation is illustrated in Equation 8 in the original paper (`https://openreview.net/pdf?id=NqbktPUkZf7`).

$$h_i' = h_{i-1} + \int_{t_{i-1}}^{t_i} f\left(h_t, \theta\right) dt, \tag{1}$$

where $f\left(h_t, \theta\right)$ is the ODE function, implemented by an autoregressive gated recurrent unit with a parameter σ. The corresponding ODE function $\tilde{f}\left(\tilde{h}_s, s, \theta\right)$ follows:

$$\begin{aligned}
\tilde{f}\left(\tilde{h}_s, s, \theta\right) := \frac{d\tilde{h}_s}{ds} &= \left.\frac{dh_t}{dt}\right|_{t=s\left(t^c_{\text{end}} - t^c_{\text{start}}\right)+t^c_{\text{start}}} \frac{dt}{ds} \\
&= \left. f\left(h_t, t, \theta\right)\right|_{t=s\left(t^c_{\text{end}} - t^c_{\text{start}}\right)+t^c_{\text{start}}} \left(t^c_{\text{end}} - t^c_{\text{start}}\right) \\
&= f\left(\tilde{h}_s, s\left(t^c_{\text{end}} - t^c_{\text{start}}\right) + t^c_{\text{start}}, \theta\right)\left(t^c_{\text{end}} - t^c_{\text{start}}\right)
\end{aligned} \tag{2}$$

Thus, Due to the neural ordinary differential equations (NODEs) , NeurTW [2] performs better on datasets with a large time granularity.

CanParl is a Canadian parliament bill voting network extracted from open website [7]. Nodes are members of parliament (MPs), and edges are the interactions between MPs from 2006 to 2019.

We illustrate the distribution of temporal edge count for the CanParl dataset in Figure 2. As shown in Figure 2, CanParl dataset has a large time granularity and NeurTW [2] achieves the best performance on CanParl dataset. Inspired by the above analysis, we could infer that NeurTW is potentially suitable for datasets with a large time granularity and time intervals, such as CanParl.

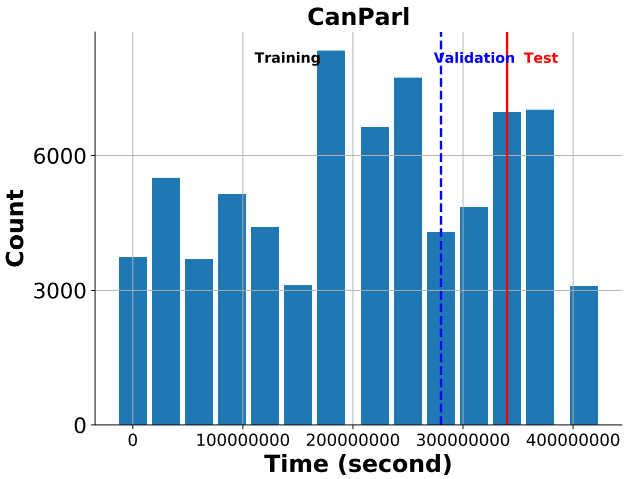

Figure 2: The distribution of temporal edge count for the CanParl dataset, and the illustration on the train-validation-test splitting.

We further conduct ablation studies to verify the effectiveness of neural ordinary differential equations (NODEs) of NeurTW on datasets with a large time granularity and time intervals. The experimental results are detailed in Table 7.

Table 7: Ablation studies on neural ordinary differential equations (NODEs) of NeurTW. "– NODEs" means remove NODEs module.

Ablation	Datasets	AUC				AP			
		Transductive	Inductive	New-Old	New-New	Transductive	Inductive	New-Old	New-New
original	CanParl	0.8920 ± 0.0173	0.8871 ± 0.0139	0.8847 ± 0.0102	0.8882 ± 0.0045	0.8528 ± 0.0213	0.8469 ± 0.0161	0.8417 ± 0.0132	0.8511 ± 0.0079
	USLegis	0.9715 ± 0.0009	0.9708 ± 0.0009	0.9682 ± 0.0018	0.9787 ± 0.0004	0.9713 ± 0.0013	0.971 ± 0.0009	0.9671 ± 0.0027	0.9803 ± 0.0005
– NODEs	CanParl	0.5 ± 0.0	0.5001 ± 0.0	0.5001 ± 0.0	0.5 ± 0.0	0.5 ± 0.0	0.5001 ± 0.0	0.5001 ± 0.0	0.5 ± 0.0
	USLegis	0.898 ± 0.004	0.9186 ± 0.0018	0.9026 ± 0.0025	0.9474 ± 0.0	0.8721 ± 0.0047	0.9037 ± 0.0034	0.8651 ± 0.0029	0.9458 ± 0.0004

NeurTW without differential equations (NODEs) module performs much poorly on datasets with a large time granularity and time intervals (such as, CanParl). However, on a tiny time granularity and time intervals (such as, USLegis, the timestamp of USLegis is only from 0 to 11.), thus, removing the differential equations (NODEs) module has relatively little negative impact on the performance of the NeurTW.

Ablation studies on neural ordinary differential equations (NODEs) of NeurTW verify that "NeurTW introduces a continuous-time operation that can depict evolution trajectory, which is potentially suitable for CanParl with a large time granularity and time intervals".

Comment 3

O3. While the difficulty of the inductive New-New setting task is understandable, however, to say "Nevertheless, CAWN, NeurTW, and NAT still perform well due to their structure-aware techniques" is not sufficient. More detailed explanations of the structure-aware techniques should be provided.

Response:

We appreciate this suggestion! We have updated the sentence in Section 4.2 of the paper (`https://openreview.net/pdf?id=rnZm2vQq31`) and added more detailes of the structure-aware techniques.

CAWN, NeurTW, and NAT still perform well under the inductive **New-New** setting due to their structure-aware techniques. CAWN and NeurTW are both based on motifs and index anonymization operation [1, 2]. NeurTW additionally constructs neural ordinary differential equations (NODEs). NAT relies on joint neighborhood features based on a dedicated data structure termed *N-caches* [3].

Comment 4

O4. Regarding the efficiency evaluation, it may be desirable to discuss the elapsed time for users (i.e., per epoch time \ average number of epochs). Then, it would be beneficial to break down the analysis using the per epoch time and average number of epochs, respectively. Also, the paper only reports the execution time and memory usage without conducting in-depth analyses of their correlations with the design of each existing method.

Response:

We appreciate the suggestion and totally agree!

In many real world applications, ***the inference time*** (***the elapsed time***) of methods might be more important as they are deployed in the real world. Thus, we have added **the inference time** metric to evaluate the efficiency of TGNN models. See Section A.3 for details.

Comment 5

O5. Node classification becomes more challenging as the number of labels increases, but the paper only handles binary classification, limiting the benchmark's generality. Performance benchmarks for tasks with multiple label numbers are desired.

Response:

Thanks for this valuable suggestion! We have added experiments of the node classification task with multiple label numbers.

In the previous works, only Reddit, Wikipedia, and MOOC datasets have node labels (two labels: 0 and 1) and are used for binary node classification task. Through unremitting efforts, we have finded a large-scale temporal dataset - DGraphFin [5] with multiple node labels. DGraphFin consists of 3,700,550 nodes and 4,300,999 edges. 4,300,999 edges.

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 [5].

There four classes. Below are the nodes counts of each class.

- 0: 1210092
- 1: 15509
- 2: 1620851
- 3: 854098

Nodes of Class 1 are fraud users and nodes of 0 are normal users, and they the two classes to be predicted. Nodes of Class 2 and Class 3 are background users.

We preprocess DGraphFin as the format of temporal graph. We open source the code of preprocessing DGraphFin dataset at `https://github.com/qianghuangwhu/benchtemp/blob/master/DGraphFin/DGraphFin.py`.

DGraphFin dataset has been hosted on the open-source platform zenodo (https://zenodo.org/) with a Digital Object Identifier (DOI) 10.5281/zenodo.8267771 (`https://zenodo.org/record/8267846`).

The experimental results for dynamic node classification task on DGraphFin dataset are shown in Table 8. Different evaluation metrics are available, including Accuracy, Precision, Recall and F1.

$$
\begin{aligned}
\text{Precision }_{\text{weighted}} &= \frac{\sum_{i=1}^{N} \text{ Support }_{i} \times \text{ Precision }_{i}}{\sum_{i=1}^{N} \text{ Support }_{i}} \\
\text{Recall }_{\text{weighted}} &= \frac{\sum_{i=1}^{N} \text{ Support }_{i} \times \text{ Recall }_{i}}{\sum_{i=1}^{N} \text{ Support }_{i}} \\
\text{F1}_{\text{weighted}} &= \frac{2 \times \text{ Precision }_{\text{weighted}} \times \text{ Recall }_{\text{weighted}}}{\text{Precision }_{\text{weighted}} + \text{ Recall }_{\text{weighted}}}
\end{aligned}
\tag{3}
$$

where Support $_i$ is the number of supports for the i-th class.

Table 8: The experimental results for dynamic node classification task with multiple labels on DGraphFin dataset.

	JODIE	DyRep	TGN	TGAT	CAWN	NeurTW	NAT
Accuracy	0.4396 ± 0.0070	0.4400 ± 0.0117	**0.5696 ± 0.0056**	0.5529 ± 0.0036	0.4366±0.0068	0.4933±0.0870	0.4131±0.0038
Precision	0.1933 ± 0.0062	0.1937 ± 0.0103	**0.4806 ± 0.0053**	0.4644 ± 0.0118	0.1906±0.0060	0.3879±0.2849	0.3068±0.0044
Recall	0.4396 ± 0.0070	0.4400 ± 0.0117	**0.5696 ± 0.0056**	0.5529 ± 0.0036	0.4366±0.0068	0.4933±0.0870	0.4131±0.0038
F1	0.2685 ± 0.0073	0.2690 ± 0.0121	**0.4905 ± 0.0063**	0.4744 ± 0.0020	0.2654±0.0071	0.3727±0.1588	0.3402±0.0047

As shown in Table 8, TGN achieves the best performance on dynamic node classification task with multiple labels, followed by TGAT. JODIE, DyRep, and CAWN perform poorly.

Comment 6

O6. The statement "MOOC is relatively denser, and the temporal walk mechanism can effectively perceive local structures" is interesting, however, there is insufficient evidence to support the claim. Can the paper demonstrate that the effectiveness of the temporal walk mechanism changes in response to changes in graph density?

Response:

We appreciate your valuable suggestion! We have added experiments to demonstrate that the effectiveness of the temporal walk mechanism changes in response to changes in graph density.

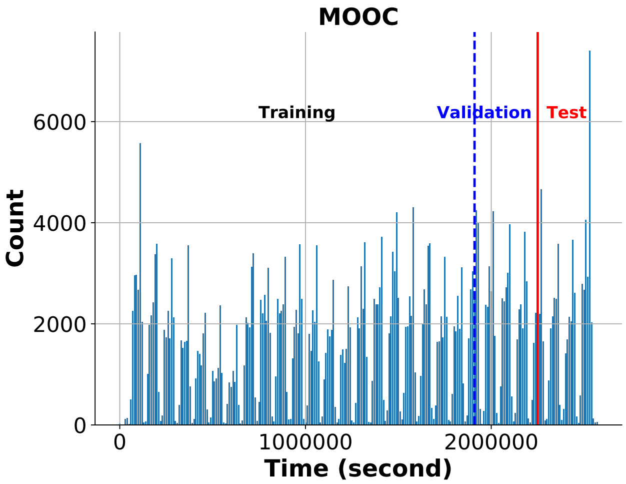

Figure 3: The distribution of temporal edge count for the MOOC dataset, and the illustration on the train-validation-test splitting.

As shown in Figure 3, MOOC is relatively denser and CAWN based on the temporal walk mechanism (motifs) achieves the best performance on this dataset.

To demonstrate that the effectiveness of the temporal walk mechanism changes in response to changes in graph density, we adopt edges sampling strategy. We randomly sample a constant number N_e of temporal edges $\{(u_1, i_1), \ldots, (u_N, i_N)\}$ each time as a subgraph G_S of the original temporal graph. The number N_u of the source nodes is the number of the elements in set $\{u_1, \ldots, u_N\}$. The number N_i of the source nodes is the number of the elements in set $\{i_1, \ldots, i_N\}$.

The graph density σ_{D_S} in our paper is the temoral density of temoral edges (u_t, i_t), the calculation formula is as follows:

$$\sigma_{D_S} = \frac{N_e}{N_u \times N_i}. \tag{4}$$

The number of sampled edges is a constant N_e. Thus, we sampled two subgraphs: G_{S_1} and G_{S_2}. The statistics of sampled subgraphs are shown in Table 9. The temoral graph density of G_{S_1} is 0.6320, while G_{S_2} 0.3127.

Table 9: The parameters of sampled graphs G_{S_1}, G_{S_2}.

	N_e	N_u	N_i	σ_{D_S}
G_{S_1}	100000	4395	36	0.6320
G_{S_2}	100000	4264	75	0.3127

The experimental results of CAWN based on the temporal walk mechanism (motifs) on G_{S_1} and G_{S_2} are shown in Table 10.

Table 10: The experimental results of CAWN on G_{S_1} and G_{S_2}.

	AUC				AP			
	Transductive	Inductive	Inductive New-Old	Inductive New-Old	Transductive	Inductive	Inductive New-Old	Inductive New-Old
G_{S_1}	0.886 ± 0.0164	0.883 ± 0.0158	0.8847 ± 0.0166	0.8701 ± 0.0134	0.8651 ± 0.021	0.8601 ± 0.0219	0.8616 ± 0.0225	0.8511 ± 0.0174
G_{S_2}	0.8357 ± 0.0073	0.8353 ± 0.0105	0.8535 ± 0.0087	0.7752 ± 0.0157	0.8172 ± 0.0088	0.8154 ± 0.0128	0.8337 ± 0.0103	0.7535 ± 0.0197

As shown in Table 10, CAWN performs much better on G_{S_1} with a larger graph density, $\sigma_{D_{S_1}} = 0.6320 > \sigma_{D_{S_2}} = 0.3127$. The experimental results demonstrate that the effectiveness of the CAWN based on temporal walk mechanism changes in response to changes in graph density.

Comment 7

O7. The proposed technique, TeMP, is only briefly described in seven lines, making it difficult to fully understand its features.

Response:

We are grateful for this suggestion!

TeMP is a novel approach that incorporates GNN aggregation and temporal structure. TeMP performs node pre-initialization as an alternative approach to global memory, and utilizes message passing on temporal subgraphs. Moreover, TeMP runs label propagation to capture evolving local structures, motivated by the temporal walk technique. We provide detail of TeMP at Section E of Appendix (`https://openreview.net/attachment?id=rnZm2vQq31&name=supplementary_material`).

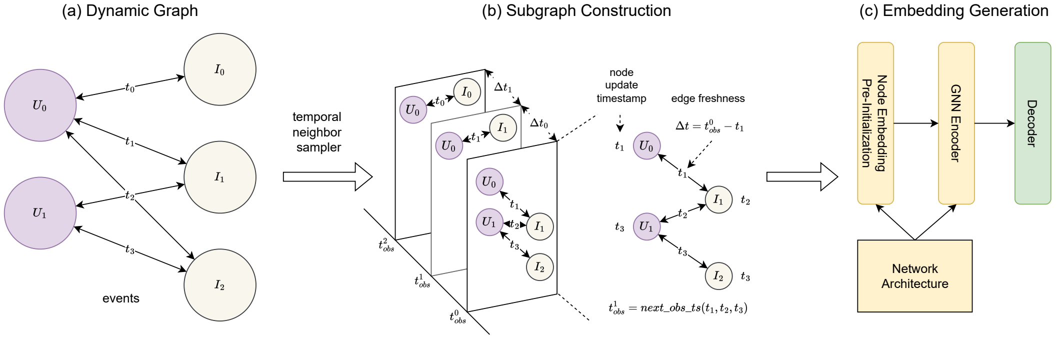

Figure 4: Workflow of TeMP. U denotes user, and I denotes item.

As shown in Figure 4, given a dynamic graph (a), the processing of TeMP is as follows:

- *Subgraph Construction (b).*

  We construct a subgraph with a temporal neighbor sampler with intervals adaptive to data. We try to find a reference timestamp and sample a subgraph before this timestamp. We have conducted experiments at various quantiles, and chosen the mean timestamp since it obtains the overall best performance.

- *Embedding Generation (c).* Upon subgraph construction, TeMP generates temporal embeddings for nodes and edges. The model architecture consists of three main components: temporal label propagation (LPA), message-passing operators, and a sequence updater. The temporal LPA captures the motif pattern, while the message-passing operators aggregate the original edge features. The sequence updater chooses RNN to update the embeddings with a memory module. Furthermore, TeMP uses a pre-initialization strategy to generate initial temporal node embeddings.

We provide detail of TeMP at Section E of Appendix (`https://openreview.net/attachment?id=rnZm2vQq31&name=supplementary_material`).

References

[1] 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.

[2] 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.

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

[4] 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.

[5] 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.

[6] 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.

[7] Shenyang Huang, Yasmeen Hitti, Guillaume Rabusseau, and Reihaneh Rabbany. Laplacian change point detection for dynamic graphs. In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 349–358, 2020.