Topological and Temporal Data Augmentation for Temporal Graph Networks

Anonymous Author(s)
Affiliation
Address
email

Abstract

Temporal graphs are extensively employed to represent evolving networks, finding applications across diverse fields such as transportation systems, social networks, and biological networks. Temporal Graph Networks (TGNs) build upon these graphs to model and learn from temporal dependencies in dynamic networks. A significant aspect of enhancing the performance of TGNs lies in effective data augmentation, which helps in better capturing the underlying patterns within temporal graphs while ensuring robustness to variations. However, existing data augmentation strategies for temporal graphs are largely heuristic and hand-crafted, which may alter the inherent semantics of temporal graphs, thereby degrading the performance of downstream tasks. To address this, we propose two simple yet effective data augmentation strategies, specifically tailored within the representation space of TGNs, targeting both the graph topology and the temporal axis. Through experiments on future link prediction and node classification tasks, we demonstrate that the integration of our proposed augmentation methods significantly amplifies the performance of TGNs, outperforming state-of-the-art methods.

1 Introduction

Temporal graphs have gained increasing attention due to their wide application in diverse fields, from social networks [11, 16] and transportation systems [1, 24] to biomedical applications [9, 28] and financial markets [14, 15]. These graphs capture not only static connections between entities but also the *dynamic evolution* of these relationships over time, where every edge (or interaction) has a timestamp to denote its occurrence time. Therefore, the temporal nature of such graphs presents unique challenges for modeling and learning. In this paper, we focus on continuous-time dynamic graphs [11], where timestamps associated with interactions can span any continuous value across the entire time range. This characteristic often results in data redundancy [20, 21] along the temporal axis, which, in turn, could engender overfitting in temporal graph learning models, thereby highlighting the need for data augmentation methods.

Data augmentation [3, 10, 17, 27] is a crucial technique for training neural networks, diversifying training data and enhancing model generalization across many domains. For example, image processing functions like randomized cropping and horizontal flipping are widely adopted in image recognition models. Existing research on temporal graph augmentation [19] predominately relies on hand-crafted augmentation strategies such as removing edges, perturbing timestamps, or adding edges with perturbed timestamps – to modify either the graph structure or temporal features. However, our empirical results (as detailed in Section 3.2) reveal that these hand-crafted augmentations might change the semantics of the original temporal graphs and harm downstream task performance. For instance, alterations in timestamps could lead to a change in the chronological order of events, which in turn misguide the model's comprehension of causal relationships between these events. Given

Submitted to Temporal Graph Learning Workshop @ NeurIPS 2023, New Orleans. Do not distribute.

these insights, there emerges a clear need for more refined augmentation approaches for temporal graphs that can mitigate such adverse effects while enhancing model performance.

Motivated by these revelations, our objective is to identify data augmentation strategies that can benefit the performance of TGNs. Drawing inspiration from the achievements of latent augmentation techniques across diverse modalities such as text, tabular, time-series, and image data [3], we propose two straightforward yet effective data augmentation strategies. Specifically, these strategies are designed to operate within the representation space of temporal graph networks, targeting enhancements along both the topological and temporal axes of temporal graphs. The topological strategy aims to enhance the robustness of aggregated messages, which often encapsulate temporal interactions. The temporal strategy refines temporal graphs by introducing temporal smoothness along the temporal axis, aiding in better understanding the temporal evolution of the graph. We conduct experiments on both link prediction and node classification tasks. The results show that by combining these two augmentation strategies, our method has significant improvement built on backbone models and outperforms or performs comparably to the SOTA methods on widely-used datasets, which supports the efficacy of our method.

2 Related Works

Representation Learning for Temporal Graphs. While most graph neural networks are designed for representation learning on static graphs, there is relatively less work focused on temporal graph learning, where the graph evolves over time. Temporal graphs can generally be classified into two categories: discrete-time dynamic graphs (DTDG) [12] and continuous-time dynamic graphs (CTDG) [11]. DTDGs consist of sequences of static graph snapshots taken at specific time intervals, whereas CTDGs are more versatile and can be represented as timed lists of events. Initially, early models for temporal graph learning predominantly concentrated on DTDGs [6, 8, 12, 25], until recent developments introduced methods tailored for CTDGs [4, 11, 16, 18, 22]. For example, DyRep [18] and JODIE [11] employ RNNs to propagate messages across interactions to update node representations. Additionally, TGAT [22] employs self-attention to aggregate messages from the central node's neighbors before event timestamps. Building upon these methodologies, TGN [16] provide a generic framework combining memory module (*e.g.* RNNs) and graph-based operator. In parallel with these efforts, we propose data augmentation strategies designed to be integrated with popular TGN backbones, effectively enhancing their performance.

Data Augmentation on Temporal Graphs. Data augmentation is significant for training neural networks, and its effectiveness has been validated on image data [10, 17, 27]. While several data augmentation methods [5, 7, 23, 26] have been developed for static graphs, they cannot be directly applied to temporal graphs due to their lack of consideration for the temporal axis. Specifically, these methods primarily focus on edge addition or deletion without accounting for the timestamps, which are integral to capturing the dynamic evolution in temporal graphs. Furthermore, there have been limited efforts in augmenting CTDGs. Among them, the most related work is MeTA [19], which adaptively combines heuristic augmentations like edge removal, timestamp perturbation, and edge addition with perturbed timestamps. MeTA employs varying magnitudes for these augmentations—applying lower magnitudes for nodes that are closer in time or topology, and larger magnitudes for those that are more distant. In contrast to MeTA, our approach does not apply transformations to the temporal graph structure or node features but rather focuses on augmenting the feature space of messages for TGNs.

3 Topological and Temporal Data Augmentation

3.1 Preliminaries

Dynamic Graphs. Following [13], we represent a dynamic graph as a sequence of interaction events – triplets of source, destination, timestamp, *i.e.* $\mathcal{G} = \{(s_1, v_1, t_1), (s_2, v_2, t_2), \cdots, (s_n, v_n, t_n)\}$, where $0 \leq t_1 \leq t_2 \leq \cdots \leq t_n$. In this paper, we focus on continuous time dynamic graphs (CTDGs), in which the time t associate with each interaction is not restricted to specific timestamps with fixed intervals but can be any continuous value within the entire time span $[0, t_n]$.

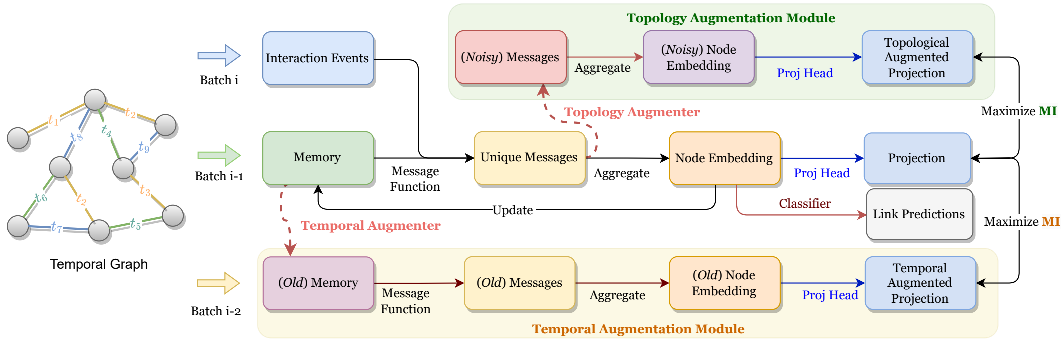

Figure 1: An Illustration of the proposed TTDA framework. Our proposed topological and temporal augmentation strategies are shown in the green and yellow boxes, respectively.

Temporal Graph Networks. Advanced temporal graph networks generally consists of five key modules [16], namely memory module, message function, message aggregator, memory updater, and embedding module. An RNN-based memory module is used to store the state of each node at current timestamp. Specifically, the state $\mathbf{s}_i(t^-)$ of node i is expected to represent i's history before time t in a compressed format. When a new event involving node i is observed, a message function is used to computed a new message $\mathbf{m}_i(t)$. Then all messages involving node i are combined with a message aggregator into a final message $\bar{\mathbf{m}}_i(t)$. A memory updater updates the memory of a node based on the aggregated messages and previous state $\mathbf{s}_i(t^-)$,

$$\mathbf{s}_i(t) = \mathrm{mem}(\bar{\mathbf{m}}_i(t), \mathbf{s}_i(t^-)). \tag{1}$$

Finally, the embedding of node i at timestamp t is derived by considering its state in the memory, associated edge, and node feature. And the node embeddings subsequently serve as input to generate predictions for specific downstream tasks, such as link prediction and node classification. Note that different models can have different implementations of these five models, tailored to address specific downstream tasks and objectives.

3.2 Motivation

For temporal graphs, Wang et al. [19] proposed three heuristic data augmentation strategies, namely (i) removing edges, (ii) perturbing timestamps, and (iii) adding edges with perturbed timestamps. While the proposed method in that paper [19] *adaptively* combine these augmentations and achieve encouraging results, we notice that performing the augmentation *directly* on the input temporal graph structure does not bring improvement on the TGN performance. A plausible explanation for this could be that such augmentations could potentially disrupt the temporal coherency and the structural integrity of the graph, which in turn could mislead the learning process, rendering a model less effective in capturing the underlying temporal dynamics. To illustrate the impact of augmentation strategies and their combinations, we conduct experiments on a Wikipedia dataset with magnitude of 0.1 on a widely-used TGN model [16]. Figure 2 shows the performance gains (%) in terms of test accuracy for the transductive link prediction task under different augmentation combinations compared to no augmentation. It is evident from the results that all augmentation combinations yield negative effects compared to the absence of augmentation. This observation indicates the detrimental impact of these augmentations on the model training process. Motivated by this, we want to undercover data augmentation strategies that have positive impact on the training process and can be further used in contrastive learning or label-invariant learning for temporal graphs. We propose two simple but effective data augmentation strategies, topological and temporal data augmentation (TTDA) techniques, on the representation

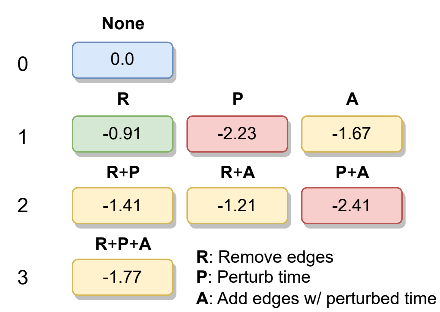

Figure 2: Performance gains (%) of TGN [16] under different augmentation settings on Wikipedia dataset [11] compared to no augmentation setting.

space of messages functions instead of altering original temporal graph structure or timestamps. An overview of our model is shown in Figure 1.

3.3 Message Topological Augmentation

Inspired by latent augmentation methods [3, 26], we carefully design a message topology augmentation to avoid *refeeding* edges for each batch. During the training phase of temporal graph, when processing a batch, the memory module is updated with messages coming from previous batches, and then predicts the interactions. This mechanism enables gradient back-propagation through the memory-related modules while mitigating information leakage. Consequently, in the design of augmentations for temporal graphs, we maintain this efficient training mechanism and perturb messages directly. An illustration of the proposed message topology augmentation module is shown in the green box in Figure 1. Specifically, when the raw messages $\mathbf{rm}_i(t^-)$ from previous batches are retrieved, we augment $\mathbf{rm}_i(t^-)$ by adding Gaussian noise with with zero mean and a standard deviation of $\mathbf{I}$. Let $\epsilon \sim \mathcal{N}(0, \mathbf{I})$ be a random unit vector and λ_{topo} be the scaling factor for message topological augmentation, we have the noisy messages:

$$\hat{\mathbf{m}}_i(t) = \mathbf{rm}_i(t) + \lambda_{topo}\epsilon. \tag{2}$$

Then we apply the message aggregator and embedding module to obtain the topology augmented node embedding $\mathbf{z}_i(t)$ at current timestamp t, while the memory is updated with original node embedding. In this way, we augment the messages without refeeding all edges for each batch.

To enhance the robustness of TGNs against noise, we maximize the mutual information between $\hat{\mathbf{z}}_i(t)$ and original node embedding $\mathbf{z}_i(t)$. To achieve this, we employ a projection head consist of fully-connected layers to get projections $\hat{\mathbf{p}}_i(t)$ and $\mathbf{p}_i(t)$ from $\hat{\mathbf{z}}_i(t)$ and $\mathbf{z}_i(t)$, respectively. To maximize the mutual information, we apply a normalized temperature-scaled cross-entropy loss (NT-Xent) [2] but only keep the positive-pair part as follows:

$$\mathcal{L}_{topo} = \frac{-(\mathbf{p}_i(t))^\top \cdot \hat{\mathbf{p}}_i(t)}{\|\mathbf{p}_i(t)\| \cdot \|\hat{\mathbf{p}}_i(t)\|}. \tag{3}$$

Diverging from existing methods, our strategy augments the representation space to diversify aggregated messages, thereby bolstering the robustness of TGNs. Additionally, our approach maintains the efficient training scheme of TGNs, eliminating the need for refeeding during training.

3.4 Message Temporal Augmentation

In temporal graphs, it is often observed that more recent edges on the temporal axis typically carry higher predictive value for the target node's states [16, 19]. Therefore, we impose a smoothness constraint over time in temporal graph learning to encourage the model to capture and exploit these informative temporal patterns effectively. We design a message temporal augmentation to achieve this. An illustration of the proposed message topology augmentation module is shown in the yellow box in Figure 1. Before each update of the memory, we store a copy of node states. Given a batch of interacting events, instead of using the raw messages in previous batch, we use the stored, obsolete node states to compute message $\tilde{\mathbf{m}}_i(t)$ and node embedding $\tilde{\mathbf{z}}_i(t)$. Then we use the projection head to obtain projection $\tilde{\mathbf{p}}_i(t)$. To enforce the temporal smoothness, similar to the process in message topological augmentation module, we maximize the mutual information between temporal augmented projection $\tilde{\mathbf{p}}_i(t)$ and the original projection $\mathbf{p}_i(t)$ as positive pairs with loss $\mathcal{L}_{temp}$.

To summarize, this augmentation strategy maximizes the mutual information between neighboring batches, imposing smoothness along the temporal axis, thereby enhancing the robustness of modeling timestamps and inherent dynamics of the graph in TGNs.

3.5 Overall Loss Function

Combining the message topological and temporal augmentation, as introduced in Sec. 3.3 and Sec. 3.4, respectively, our TTDA framework is trained by minimizing the combined loss

$$\mathcal{L} = \mathcal{L}_{CE} + \alpha\mathcal{L}_{topo} + \beta\mathcal{L}_{temp}. \tag{4}$$

where $\mathcal{L}_{CE}$ is the cross-entropy loss for downstream tasks, α and β are predefined hyper-parameters. Our proposed method can adapt to memory-based TGNs in an efficient way.

Table 1: Statistics of the datasets used in the experiments.

Data	#Users	#Items	#Interactions	#State Changes	Action Repetition
Reddit	10,000	984	672,447	366	79%
Wikipedia	8,227	1,000	157,474	217	61%
MOOC	7,047	97	411,749	4,066	–

Table 2: Test accuracy and average precision (AP) of transductive edge prediction. Mean (%) and standard deviations are reported. The best results are highlighted in **bold**.

Method	MOOC		Reddit		Wikipedia	
	Accuracy	AP	Accuracy	AP	Accuracy	AP
JODIE [11]	76.45 ± 0.6	83.87 ± 0.4	90.91 ± 0.3	97.11 ± 0.3	87.04 ± 0.4	94.62 ± 0.5
TGAT [22]	75.20 ± 0.5	82.66 ± 0.4	92.92 ± 0.3	98.12 ± 0.2	88.14 ± 0.2	95.34 ± 0.1
DyRep [18]	73.36 ± 0.4	81.75 ± 0.3	92.11 ± 0.2	97.98 ± 0.1	87.77 ± 0.2	94.59 ± 0.2
TGN [16]	81.38 ± 0.6	89.79 ± 0.5	92.56 ± 0.2	98.70 ± 0.1	89.51 ± 0.4	98.46 ± 0.1
DyRep + MeTA	76.21 ± 0.4	84.18 ± 0.3	93.04 ± 0.3	$\mathbf{98.62 \pm 0.1}$	88.92 ± 0.2	95.63 ± 0.2
DyRep + TTDA (Ours)	$\mathbf{84.12 \pm 0.7}$	$\mathbf{89.17 \pm 0.3}$	$\mathbf{93.54 \pm 0.2}$	98.39 ± 0.1	$\mathbf{91.87 \pm 0.2}$	$\mathbf{98.10 \pm 0.1}$
TGN + MeTA	83.84 ± 0.5	92.03 ± 0.3	94.19 ± 0.2	$\mathbf{99.08 \pm 0.1}$	91.34 ± 0.3	$\mathbf{98.87 \pm 0.1}$
TGN + TTDA (Ours)	$\mathbf{86.58 \pm 0.1}$	$\mathbf{92.03 \pm 0.7}$	$\mathbf{94.53 \pm 0.1}$	98.76 ± 0.1	$\mathbf{93.67 \pm 0.2}$	98.65 ± 0.1

4 Experiments

4.1 Experiment Setup

Dataset. We conduct experiments on three widely-used temporal graph datasets, namely Wikipedia, Reddit, and MOOC [11]. Following Xu et al. [22], we use a chronological train/validation/test split with a ratio of 70%-15%-15%. Table 1 shows the statistics of these datasets. The downstream tasks we use are future link prediction and dynamic node classification.

Baselines. We take the state-of-the-art approaches for continuous time dynamic graph learning, namely JODIE [11], TGAT [22], DyRep [18], and TGN [16] as well as the state-of-the-art temporal graph augmentation method MeTA [19]. We implement our TTDA augmentation strategies on top of the baseline models DyRep and TGN.

Experimental settings. Our experimental settings closely follow those of the previous work [22, 16] to a ensure fair comparison. For the all datasets, we use the Adam optimizer with a learning rate of 0.0001, a batch size of 200 for both training, validation and testing, and early stopping with a patience of 5. We sample an equal amount of negatives to the positive interactions, and use average precision as reference metric. All experiments and timings are conducted on an NVIDIA RTX A5000 machine and the results are averaged over 10 runs. The code will be made available for reproduction.

4.2 Link Prediction

For the link prediction task, we study both the transductive and inductive setting. We conduct experiments and report the average precision and accuracy on the test set over 10 runs. For the transductive setting, the nodes for link prediction are observed during training. The results are shown in Table 6. The numbers for baselines are taken from Wang et al. [19]. We observe that by adding our TTDA strategies, the test accuracy of DyRep improves by 7.91% on MOOC, 0.5% on Reddit,

Table 3: Test accuracy and average precision (AP) of inductive edge prediction. Mean (%) and standard deviations are reported. The best results are highlighted in **bold**.

Method	MOOC		Reddit		Wikipedia	
	Accuracy	AP	Accuracy	AP	Accuracy	AP
JODIE [11]	75.79 ± 0.5	83.44 ± 0.6	88.34 ± 0.9	94.36 ± 1.1	84.32 ± 0.4	93.11 ± 0.4
TGAT [22]	74.02 ± 0.3	80.84 ± 0.5	90.73 ± 0.2	96.62 ± 0.3	85.35 ± 0.2	93.99 ± 0.3
DyRep [18]	72.92 ± 0.4	80.36 ± 0.4	89.60 ± 0.2	95.68 ± 0.2	83.46 ± 0.3	92.05 ± 0.3
TGN [16]	80.73 ± 0.2	89.21 ± 0.3	91.62 ± 0.1	97.55 ± 0.1	88.60 ± 0.2	97.81 ± 0.1
DyRep + MeTA	75.89 ± 0.4	82.56 ± 0.3	90.52 ± 0.2	96.59 ± 0.2	85.67 ± 0.3	94.13 ± 0.2
$\mathbf{DyRep + TTDA(Ours)}$	$\mathbf{82.27 \pm 0.4}$	$\mathbf{88.04 \pm 0.2}$	$\mathbf{91.38 \pm 0.4}$	$\mathbf{97.32 \pm 0.2}$	$\mathbf{90.47 \pm 0.2}$	$\mathbf{97.60 \pm 0.1}$
TGN + MeTA	83.47 ± 0.2	90.85 ± 0.2	$\mathbf{92.96 \pm 0.1}$	$\mathbf{98.17 \pm 0.1}$	90.82 ± 0.2	$\mathbf{98.26 \pm 0.1}$
$\mathbf{TGN + TTDA(Ours)}$	$\mathbf{85.49 \pm 0.1}$	$\mathbf{91.45 \pm 0.5}$	92.43 ± 0.1	97.86 ± 0.1	$\mathbf{91.89 \pm 0.2}$	98.10 ± 0.1

Table 4: ROC AUC for the dynamic node classification. Mean (%) and standard deviations are reported. The best results are highlighted in **bold**.

Method	Reddit	Wikipedia
JODIE [11]	61.83 ± 2.7	84.84 ± 1.2
TGAT [22]	65.56 ± 0.7	83.69 ± 0.7
DyRep [18]	62.91 ± 2.4	84.59 ± 2.2
TGN [16]	67.06 ± 0.9	87.81 ± 0.3
DyRep + MeTA	64.36 ± 2.0	86.65 ± 1.9
DyRep + TTDA(Ours)	$\mathbf{68.38 \pm 0.9}$	$\mathbf{87.90 \pm 1.3}$
TGN + MeTA	68.37 ± 0.9	90.03 ± 0.3
TGN + TTDA(Ours)	$\mathbf{69.75 \pm 1.3}$	$\mathbf{90.08 \pm 0.6}$

and 2.95% on Wikipedia, comparing with SOTA data augmentation method MeTA [19]. Besides, the test accuracy of TGN improves 2.74%, 0.34%, 2.33% on MOOC, Reddit, and Wikipedia dataset, respectively, comparing with the previous SOTA temporal graph augmentation method MeTA [19].

As for the inductive setting, we predict edges for unseen nodes. We keep the same experiment settings and baselines with transductive setting. Table 7 shows the results. We observe that by adding our TTDA strategies, the test accuracy improves by 6.38% on MOOC, 5.48% on Reddit, and 4.8% on Wikipedia by using DyRep as the backbone model, and 1.02% on MOOC and 1.07% on Wikipedia with TGN, comparing with MeTA [19]. To summarize, our TTDA method enhances DyRep and TGN to outperform the baseline methods in both transductive and inductive link prediction tasks.

4.3 Node Classification

In this experiment, the goal is to predict the time-varying labels of nodes following an interaction event [11]. For node classification, the transductive setting is used. Following previous works [16], we initiate the model with pre-training via the link prediction task. Subsequently, we fixed the parameters of the TGN models and introduced a one-layer MLP as the decoder for the node classification task. The decoder was supervisedly trained using the node state labels. The results are shown in Table 4 where we use the area under the receiver operating characteristic curve (AUC-ROC) metrics for evaluation. We observe that TTDA achieves SOTA results on both datasets and improves ROC AUC of DyRep by 2.1% on MOOC, 2.3% on Reddit, 2.4% on Wikipedia, and TGN by 2.2% on MOOC, 2.0% on Reddit, 2.5% on Wikipedia. The results indicate the effectiveness of our proposed augmentation strategies in enhancing the performance of TGNs in the node classification tasks.

4.4 Ablation Studies

In this section, we investigate the contributions of our data augmentation strategies. We apply the topology augmentation and topology augmentation mechanisms in our TTDA separately on the TGN model. The experiments are conducted under the inductive link prediction setting, and the results are shown in Table 5. We observe that both topology and temporal augmentation strategies yield positive effects on the TGN training. Combining them results in achieving SOTA performance.

Table 5: Ablation study on the impact of proposed augmentations in inductive link prediction tasks.

Method	MOOC		Reddit		Wikipedia	
	Accuracy	AP	Accuracy	AP	Accuracy	AP
No Aug.	80.73 ± 0.2	89.21 ± 0.3	91.62 ± 0.1	97.55 ± 0.1	88.60 ± 0.2	97.81 ± 0.1
Topo. Aug.	85.06 ± 0.8	91.08 ± 0.5	92.27 ± 0.3	97.76 ± 0.1	91.56 ± 0.2	97.92 ± 0.2
Temp. Aug.	$\mathbf{85.51 \pm 0.1}$	91.39 ± 0.5	91.94 ± 0.1	97.63 ± 0.1	91.80 ± 0.4	98.06 ± 0.2
TTDA	85.49 ± 0.1	$\mathbf{91.45 \pm 0.5}$	$\mathbf{92.43 \pm 0.1}$	$\mathbf{97.86 \pm 0.1}$	$\mathbf{91.89 \pm 0.2}$	$\mathbf{98.10 \pm 0.1}$

5 Conclusions

In this paper, we identify the limitations of current data augmentation techniques for temporal graphs and introduce two novel strategies for temporal graph networks. Targeting both graph topology and temporal axis, we aim to enhance robustness and performance of downstream tasks without directly altering graph structure and features. Through experiments, we showcase how our strategies significantly improve TGN performance. Our method proves valuable for training temporal graph networks and holds promise for application in domains like contrastive and label-invariant learning.

References

[1] Khac-Hoai Nam Bui, Jiho Cho, and Hongsuk Yi. Spatial-temporal graph neural network for traffic forecasting: An overview and open research issues. *Applied Intelligence*, 52(3): 2763–2774, 2022.

[2] Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. A simple framework for contrastive learning of visual representations. In *International conference on machine learning*, pages 1597–1607. PMLR, 2020.

[3] Tsz-Him Cheung and Dit-Yan Yeung. Modals: Modality-agnostic automated data augmentation in the latent space. In *International Conference on Learning Representations*, 2020.

[4] Weilin Cong, Si Zhang, Jian Kang, Baichuan Yuan, Hao Wu, Xin Zhou, Hanghang Tong, and Mehrdad Mahdavi. Do we really need complicated model architectures for temporal networks? In *The Eleventh International Conference on Learning Representations*, 2023. URL `https://openreview.net/forum?id=ayPPc0SyLv1`.

[5] Kaize Ding, Zhe Xu, Hanghang Tong, and Huan Liu. Data augmentation for deep graph learning: A survey. *ACM SIGKDD Explorations Newsletter*, 24(2):61–77, 2022.

[6] Daniel M Dunlavy, Tamara G Kolda, and Evrim Acar. Temporal link prediction using matrix and tensor factorizations. *ACM Transactions on Knowledge Discovery from Data (TKDD)*, 5 (2):1–27, 2011.

[7] Xiaotian Han, Zhimeng Jiang, Ninghao Liu, and Xia Hu. G-mixup: Graph data augmentation for graph classification. In *International Conference on Machine Learning*, pages 8230–8248. PMLR, 2022.

[8] Ryohei Hisano. Semi-supervised graph embedding approach to dynamic link prediction. In *Complex Networks IX: Proceedings of the 9th Conference on Complex Networks CompleNet 2018 9*, pages 109–121. Springer, 2018.

[9] Mansu Kim, Jaesik Kim, Jeffrey Qu, Heng Huang, Qi Long, Kyung-Ah Sohn, Dokyoon Kim, and Li Shen. Interpretable temporal graph neural network for prognostic prediction of alzheimer's disease using longitudinal neuroimaging data. In *2021 IEEE International Conference on Bioinformatics and Biomedicine (BIBM)*, pages 1381–1384. IEEE, 2021.

[10] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. *Advances in neural information processing systems*, 25, 2012.

[11] 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 & data mining*, pages 1269–1278, 2019.

[12] David Liben-Nowell and Jon Kleinberg. The link prediction problem for social networks. In *Proceedings of the twelfth international conference on Information and knowledge management*, pages 556–559, 2003.

[13] Farimah Poursafaei, Shenyang Huang, Kellin Pelrine, and Reihaneh Rabbany. Towards better evaluation for dynamic link prediction. *Advances in Neural Information Processing Systems*, 35:32928–32941, 2022.

[14] Nitendra Rajput and Karamjit Singh. Temporal graph learning for financial world: Algorithms, scalability, explainability & fairness. In *Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pages 4818–4819, 2022.

[15] Shivshankar Reddy, Pranav Poduval, Anand Vir Singh Chauhan, Maneet Singh, Sangam Verma, Karamjit Singh, and Tanmoy Bhowmik. Tegraf: temporal and graph based fraudulent transaction detection framework. In *Proceedings of the Second ACM International Conference on AI in Finance*, pages 1–8, 2021.

[16] 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.

[17] Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Dropout: a simple way to prevent neural networks from overfitting. *The journal of machine learning research*, 15(1):1929–1958, 2014.

[18] Rakshit Trivedi, Mehrdad Farajtabar, Prasenjeet Biswal, and Hongyuan Zha. Dyrep: Learning representations over dynamic graphs. In *International Conference on Learning Representations*, 2019. URL `https://openreview.net/forum?id=HyePrhR5KX`.

[19] Yiwei Wang, Yujun Cai, Yuxuan Liang, Henghui Ding, Changhu Wang, Siddharth Bhatia, and Bryan Hooi. Adaptive data augmentation on temporal graphs. *Advances in Neural Information Processing Systems*, 34:1440–1452, 2021.

[20] Yiwei Wang, Yujun Cai, Yuxuan Liang, Henghui Ding, Changhu Wang, and Bryan Hooi. Time-aware neighbor sampling for temporal graph networks. *arXiv preprint arXiv:2112.09845*, 2021.

[21] Yufeng Wang and Charith Mendis. Tgopt: Redundancy-aware optimizations for temporal graph attention networks. In *Proceedings of the 28th ACM SIGPLAN Annual Symposium on Principles and Practice of Parallel Programming*, pages 354–368, 2023.

[22] Da Xu, Chuanwei Ruan, Evren Korpeoglu, Sushant Kumar, and Kannan Achan. Inductive representation learning on temporal graphs. *arXiv preprint arXiv:2002.07962*, 2020.

[23] Yuning You, Tianlong Chen, Yongduo Sui, Ting Chen, Zhangyang Wang, and Yang Shen. Graph contrastive learning with augmentations. *Advances in neural information processing systems*, 33:5812–5823, 2020.

[24] Bing Yu, Haoteng Yin, and Zhanxing Zhu. Spatio-temporal graph convolutional networks: A deep learning framework for traffic forecasting. *arXiv preprint arXiv:1709.04875*, 2017.

[25] Wenchao Yu, Wei Cheng, Charu C Aggarwal, Haifeng Chen, and Wei Wang. Link prediction with spatial and temporal consistency in dynamic networks. In *IJCAI*, pages 3343–3349, 2017.

[26] Han Yue, Chunhui Zhang, Chuxu Zhang, and Hongfu Liu. Label-invariant augmentation for semi-supervised graph classification. *Advances in Neural Information Processing Systems*, 35: 29350–29361, 2022.

[27] Hongyi Zhang, Moustapha Cisse, Yann N Dauphin, and David Lopez-Paz. mixup: Beyond empirical risk minimization. *arXiv preprint arXiv:1710.09412*, 2017.

[28] Minzhang Zheng, Sergii Domanskyi, Carlo Piermarocchi, and George I Mias. Visibility graph based temporal community detection with applications in biological time series. *Scientific reports*, 11(1):5623, 2021.

Table 6: Test accuracy and average precision (AP) of transductive edge prediction. Mean (%) and standard deviations are reported. The best results are highlighted in **bold**.

Method	MOOC		Reddit		Wikipedia	
	Accuracy	AP	Accuracy	AP	Accuracy	AP
JODIE [11]	76.45 ± 0.6	83.87 ± 0.4	90.91 ± 0.3	97.11 ± 0.3	87.04 ± 0.4	94.62 ± 0.5
TGAT [22]	75.20 ± 0.5	82.66 ± 0.4	92.92 ± 0.3	98.12 ± 0.2	88.14 ± 0.2	95.34 ± 0.1
DyRep [18]	73.36 ± 0.4	81.75 ± 0.3	92.11 ± 0.2	97.98 ± 0.1	87.77 ± 0.2	94.59 ± 0.2
TGN [16]	81.38 ± 0.6	89.79 ± 0.5	92.56 ± 0.2	98.70 ± 0.1	89.51 ± 0.4	98.46 ± 0.1
DyRep + MeTA	76.21 ± 0.4	84.18 ± 0.3	93.04 ± 0.3	$\mathbf{98.62 \pm 0.1}$	88.92 ± 0.2	95.63 ± 0.2
DyRep + TTDA (Ours)	$\mathbf{84.12 \pm 0.7}$	$\mathbf{89.17 \pm 0.3}$	$\mathbf{93.54 \pm 0.2}$	98.39 ± 0.1	$\mathbf{91.87 \pm 0.2}$	$\mathbf{98.10 \pm 0.1}$
TGN + MeTA	83.84 ± 0.5	92.03 ± 0.3	94.19 ± 0.2	$\mathbf{99.08 \pm 0.1}$	91.34 ± 0.3	$\mathbf{98.87 \pm 0.1}$
TGN + TTDA (Ours)	$\mathbf{86.58 \pm 0.1}$	$\mathbf{92.03 \pm 0.7}$	$\mathbf{94.53 \pm 0.1}$	98.76 ± 0.1	$\mathbf{93.67 \pm 0.2}$	98.65 ± 0.1

Table 7: Test accuracy and average precision (AP) of inductive edge prediction. Mean (%) and standard deviations are reported. The best results are highlighted in **bold**.

Method	MOOC		Reddit		Wikipedia	
	Accuracy	AP	Accuracy	AP	Accuracy	AP
JODIE [11]	75.79 ± 0.5	83.44 ± 0.6	88.34 ± 0.9	94.36 ± 1.1	84.32 ± 0.4	93.11 ± 0.4
TGAT [22]	74.02 ± 0.3	80.84 ± 0.5	90.73 ± 0.2	96.62 ± 0.3	85.35 ± 0.2	93.99 ± 0.3
DyRep [18]	72.92 ± 0.4	80.36 ± 0.4	89.60 ± 0.2	95.68 ± 0.2	83.46 ± 0.3	92.05 ± 0.3
TGN [16]	80.73 ± 0.2	89.21 ± 0.3	91.62 ± 0.1	97.55 ± 0.1	88.60 ± 0.2	97.81 ± 0.1
DyRep + MeTA	75.89 ± 0.4	82.56 ± 0.3	90.52 ± 0.2	96.59 ± 0.2	85.67 ± 0.3	94.13 ± 0.2
$\mathbf{DyRep + TTDA (Ours)}$	$\mathbf{82.27 \pm 0.4}$	$\mathbf{88.04 \pm 0.2}$	$\mathbf{91.38 \pm 0.4}$	$\mathbf{97.32 \pm 0.2}$	$\mathbf{90.47 \pm 0.2}$	$\mathbf{97.60 \pm 0.1}$
TGN + MeTA	83.47 ± 0.2	90.85 ± 0.2	$\mathbf{92.96 \pm 0.1}$	$\mathbf{98.17 \pm 0.1}$	90.82 ± 0.2	$\mathbf{98.26 \pm 0.1}$
$\mathbf{TGN + TTDA (Ours)}$	$\mathbf{85.49 \pm 0.1}$	$\mathbf{91.45 \pm 0.5}$	92.43 ± 0.1	97.86 ± 0.1	$\mathbf{91.89 \pm 0.2}$	98.10 ± 0.1