A DYNAMIC GRAPH LEARNING METHOD WITH BACKWARD KNOWLEDGE FEEDBACK VIA PARAMETER EXPANSION

Anonymous authors
Paper under double-blind review

ABSTRACT

Dynamic graphs play a crucial role in modeling temporal interaction systems in the real world. Current dynamic graph learning methods suffer from limited representation capability due to the following issues: (1) fixed model parameter capacity leads to knowledge squeezing when continuously learning new graph knowledge; (2) such methods adhere to the fundamental paradigm of leveraging historical graph knowledge to learn new graph knowledge, neglecting the potential of backward knowledge feedback for enhancing the learning of historical graph knowledge. To address these issues, we propose a novel **D**ynamic graph learning method with **B**ackward **K**nowledge **F**eedback via **P**arameter **E**xpansion (**DBKF-PE**), which achieves effective representation of dynamic graphs through a bidirectional knowledge circulation structure. The circulation structure consists of forward knowledge injection and backward knowledge feedback. Specifically, the forward knowledge injection incorporates a new graph-aware parameter expansion strategy. This strategy expands new parameters to learn new graph knowledge while freezing old parameters, thereby preventing conflicts between new and historical graph knowledge in the parameter space. Then, the backward knowledge feedback is modeled as a meta-learning optimization objective, which not only alleviates catastrophic forgetting through the proposed backward knowledge feedback mechanism but also further enhances the performance of the new model on historical graphs. Finally, extensive experiments on six datasets validate that DBKF-PE outperforms the state-of-the-art baselines.

1 INTRODUCTION

Dynamic graph learning (DGL), as a vital branch of graph representation learning, aims to model and analyze graph data with evolving topological structures and node attributes over time (Sankar et al., 2020; Hong et al., 2026; Qi et al., 2025; Yang et al., 2025). As it's difficult to apply traditional static graph learning methods to dynamic scenarios, researchers have recently begun to focus on DGL methods that can handle temporal dynamics.

DGL methods can be categorized into Continuous-Time Dynamic Graph (CTDG) learning methods and Discrete-Time Dynamic Graph (DTDG) learning methods. CTDG learning methods (Trivedi et al., 2019; Poursafaei et al., 2022; Farokhi et al., 2026; Pang et al., 2025) model dynamic graphs as continuously evolving processes over time, capturing real-time changes in nodes and edges through differential equations or temporal point processes. However, such methods suffer from high computational complexity and sensitivity to noise, rendering them challenging to handle large-scale sparse graphs. In contrast, DTDG learning methods (Pareja et al., 2020; He et al., 2025; Yuan et al., 2025) discretize the dynamic graph into a sequence of temporal snapshots. Such methods exhibit high computational efficiency and are easily scalable to large-scale sparse graphs.

Existing DGL methods typically employ models with fixed parameter capacity to uniformly represent the temporal evolution of dynamic graphs (Marisca et al., 2022; Kim et al., 2022; Liu et al., 2025; Xia et al., 2025). However, due to the fixed parameter capacity, the incorporation of new graph knowledge leads to the overwriting of historical graph knowledge, resulting in the issue of knowledge squeezing. Knowledge squeezing is essentially a manifestation of catastrophic forget-

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

ting in DGL, rooted in the fixed model parameter capacity, which leads to conflicts between new and historical graph knowledge within the limited parameter space. Meanwhile, events in dynamic graphs, such as node additions or deletions, exhibit strong temporal relationships. Although the strategy of directly freezing old parameters and expanding new parameters can address the issue of knowledge squeezing, it neglects the temporal logic among these events, thereby preventing the model from capturing temporal relationships.

Additionally, existing DGL methods primarily focus on unidirectional knowledge flow, which leverages historical graph knowledge to enhance the model's performance on new graphs (Huang et al., 2024; Gravina & Bacciu, 2024; Feng et al., 2025), as shown in Fig. 1(a). However, such methods overlook the potential of backward knowledge feedback, that is, the enhancement of historical graphs through knowledge from new graphs, resulting in unidirectional knowledge flow and potential performance degradation of the new model on historical graphs, as shown in Fig. 1(b). Unlike the goal of continual learning (Wang et al., 2020), which is to avoid catastrophic forgetting (i.e., ensuring that a new model does not lose previously learned graph knowledge, thereby retaining its performance on historical graphs), backward knowledge feedback not only requires the new model to retain the already acquired graph knowledge but further emphasizes leveraging newly learned graph knowledge to enhance its performance on historical graphs.

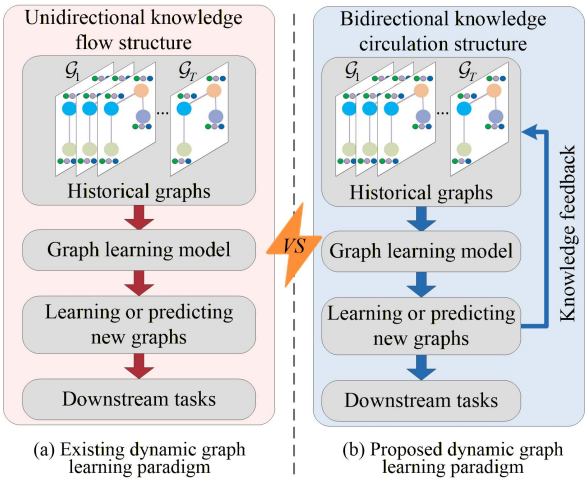


Figure 1: Comparison of existing and proposed dynamic graph learning paradigms. The existing paradigm (a) adopts a unidirectional knowledge flow structure without knowledge feedback, resulting in suboptimal model performance on historical graphs. In contrast, the proposed paradigm (b) constructs a bidirectional knowledge circulation structure through knowledge feedback, continuously improving model performance on historical graphs and enhancing model reusability.

To address the above issues, we propose a novel Dynamic graph learning method with Backward Knowledge Feedback via Parameter Expansion (DBKF-PE) for DTDG learning. Specifically, DBKF-PE constructs a bidirectional knowledge circulation structure. Through the forward knowledge injection, we first propose a new graph-aware parameter expansion strategy, constructing a new set of parameters to learn the new graph knowledge. Meanwhile, we freeze the parameters from historical graphs to ensure that historical graph knowledge is not overwritten. Building upon this, we construct a temporal auxiliary task that requires the model to consider the temporal relationship between the new graph and the graph at the previous moment while learning the new graph. Then, we propose a new backward knowledge feedback mechanism (BKFM) based on the parameters from the forward knowledge injection. BKFM constructs a meta-learning optimization objective by evaluating the performance difference between new and old parameters on historical graphs, and determines the optimization direction of the new parameters by computing meta-gradients. Our contributions are shown as follows:

- We propose a novel graph-aware parameter expansion strategy to address the issue of knowledge squeezing. Furthermore, to overcome the limitation that direct parameter expansion fails to capture temporal relationships, a new temporal guidance mechanism is proposed.
- We propose a new BKFM that transforms DGL into a meta-learning optimization objective to enhance the new model's performance on historical graphs.
- We conduct node classification and community detection tasks on six benchmark datasets. By comparing with the existing state-of-the-art baselines, the effectiveness of DBKF-PE is validated.

2 Related Work

CTDG Learning. The core idea of CTDG learning is to model the evolution of graph as a continuous-time dynamic process. For example, DyGRAIN (Kim et al., 2022) captured changes in the receptive fields of existing nodes to learn the temporal evolution information of CTDG. CPDG (Bei et al., 2024) addressed the long-term modeling dependencies in dynamic graphs through a structure-temporal subgraph sampler. However, these methods exhibited limited interpretability regarding the evolutionary characteristics of the graph. To address this, EasyDGL (Chen et al., 2024) utilized perturbation-based quantitative analysis to interpret model outputs. CT-VAE (Cheng et al., 2025) employed Hawkes processes to capture temporal dynamics and causal relationships. However, both methods suffer from low training efficiency. Recent work to improve training efficiency on dynamic graphs includes batch optimization (Feng et al., 2025), incremental learning (Xu et al., 2026), and graph pattern identification (Wu et al., 2025). However, all the aforementioned methods overlook the potential of backward knowledge feedback, failing to improve the performance of the new model on historical graphs.

DTDG Learning. The core idea of DTDG learning is to capture structural changes of snapshots at discrete timestamps. For example, DySAT (Sankar et al., 2020) employed self-attention layers to learn node representations that incorporated both structural attributes and temporal evolution. EvolveGCN (Pareja et al., 2020) utilized recurrent neural networks to evolve the parameters of graph convolutional networks, aiming to learn the dynamic characteristics. DGCN (Gao et al., 2023) employed a long short-term memory network to capture temporal evolutionary information for learning snapshot representations. However, they still failed to address the issue of long-term dependency in snapshot sequences. To this end, PI-GNN (Zhang et al., 2023) adopted parameter isolation to mitigate the issue of long-term dependency. DG-Mamba (Yuan et al., 2025) employed selective snapshot scanning to capture long-range dependencies. However, they overlooked the temporal relationships among nodes within each snapshot. To this end, both DyAug (Zhang et al., 2025) and FSTG (Liu et al., 2026) utilized causal mechanisms to capture the temporal relations between snapshots. However, all the aforementioned methods overlook the potential of backward knowledge feedback, failing to improve the performance of the new model on historical graphs.

3 Methodology

3.1 Preliminaries and Overall Structure

Preliminaries In this study, we focus on DTDG learning. A DTDG can be represented as $\mathcal{G}_{1:T} = (\mathcal{G}_1, \mathcal{G}_2, ..., \mathcal{G}_t, ..., \mathcal{G}_T)$, where $\mathcal{G}_t = (V_t, \mathbf{A}_t, \mathbf{X}_t)$ denotes a snapshot (graph) at time t, T denotes the total number of snapshots, $\mathbf{A}_t \in \mathbb{R}^{n \times n}$ denotes the adjacency matrix at time t (n denotes the number of nodes), and $\mathbf{X}_t \in \mathbb{R}^{n \times f}$ denotes the attribute matrix at time t (f denotes the number of attributes).

Overall Structure The overall structure of DBKF-PE is shown in Fig. 2. DBKF-PE primarily consists of forward knowledge injection and backward knowledge feedback. The forward knowledge injection is mainly responsible for learning new graphs, while the backward knowledge feedback is mainly responsible for feeding the knowledge of new graphs back to historical graphs to improve the overall performance of the model.

3.2 Forward Knowledge Injection

Graph Partitioning In dynamic graphs, subgraphs in $\mathcal{G}_{t+1}$ can be influenced by subgraphs in $\mathcal{G}_t$, leading to changes in $\mathcal{G}_{t+1}$ such as the addition of edges. From $\mathcal{G}_t$ to $\mathcal{G}_{t+1}$, some subgraphs of $\mathcal{G}_t$ remain unchanged. Therefore, the model does not need to repeatedly learn the information from unchanged subgraphs during message passing across snapshots, thereby improving the efficiency of parameter utilization. The changed subgraphs reflect the dynamic nature of the snapshots. We partition a snapshot at time t into a rational subgraph $\mathcal{G}_t^r$ (corresponding to the unchanged subgraph) and an environmental subgraph $\mathcal{G}_t^e$ (corresponding to the changed subgraph), as shown in Eq. (1) and the graph partitioning shown in Fig. 2.

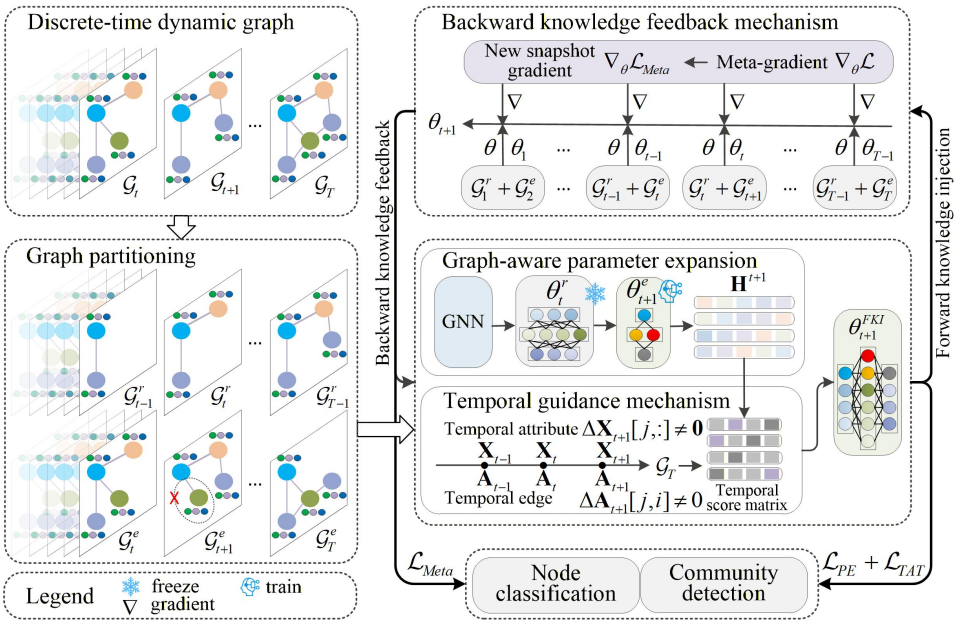


Figure 2: The overall structure of DBKF-PE. Specifically, each snapshot is partitioned into a rational subgraph and an environmental subgraph. Next, through the forward knowledge injection, a graph neural network (GNN) model is employed to learn the topological structure and attribute information of the rational subgraph. The parameters of GNN are expanded to learn the topological structure and attribute information of the environmental subgraph, while the old parameters remain frozen. Furthermore, a temporal guidance mechanism is proposed to capture the temporal relationships within the dynamic graph. Based on the proposed backward knowledge feedback mechanism (BKFM), the learning process for new snapshots not only preserves the historical graph knowledge but also enhances the new model's performance on historical graphs.

$$\mathcal{G}_t = \mathcal{G}_{t-1}^r + \Delta\mathcal{G}_t = \mathcal{G}_t^r + \mathcal{G}_t^e, \tag{1}$$

where $\Delta\mathcal{G}_t$ is a subgraph of changes derived from the evolution from $\mathcal{G}_{t-1}$ to $\mathcal{G}_t$. Note that when performing graph partitioning on $\mathcal{G}_t$, $\mathcal{G}_t^r$ and $\mathcal{G}_{t-1}^r$ are identical, both representing the subgraph where $\mathcal{G}_t$ and $\mathcal{G}_{t-1}$ remain unchanged. However, when performing graph partitioning on $\mathcal{G}_{t+1}$, $\mathcal{G}_t^r$ at time $t+1$ may differ from $\mathcal{G}_t^r$ at time t, because the unchanged subgraph from $\mathcal{G}_t$ to $\mathcal{G}_{t+1}$ may not be identical to that from $\mathcal{G}_{t-1}$ to $\mathcal{G}_t$. Additionally, to ensure that the contextual information of the environmental subgraph is not lost during the graph partitioning stage, the environmental subgraph includes not only the changed nodes and edges but also the 2-hop nodes and edges of the changed nodes. This operation also facilitates the preservation of potential relationships that may exist between nodes of the rational subgraph and those of the environmental subgraph (Zhang et al., 2026), such as temporal relationships.

Graph-aware Parameter Expansion Strategy To address the issue of knowledge squeezing, we propose a new graph-aware parameter expansion strategy, as shown in Fig. 3. Specifically, we train a new set of parameters to exclusively learn the environmental subgraph knowledge of the new snapshot. Concurrently, we freeze the parameters from historical snapshots to prevent the overwriting of historical graph knowledge. The strategy ensures that the model's parameter capacity can grow over time (i.e., parameter expansion), thereby avoiding conflicts between new and historical graph knowledge in the parameter space and addressing the issue of knowledge squeezing. The parameters θ_t^r of the rational subgraph $\mathcal{G}_t^r$ of $\mathcal{G}_t$ can be obtained through Eq. (2).

$$\theta_t^r = \arg\min_{\theta_t} \mathcal{L}(F_{\theta_t}(\mathcal{G}_t)) - \mu\mathcal{L}(F_{\theta_t}(\mathcal{G}_t^e)) \tag{2}$$

where $\mathcal{L}(\bullet)$ denotes the cross-entropy loss, μ is the balance factor, $\mathcal{L}(F_{\theta_t}(\mathcal{G}_t^e))$ / $\mathcal{L}(F_{\theta_t}(\mathcal{G}_t))$ denotes the loss value of the GNN in learning $\mathcal{G}_t^e$ / $\mathcal{G}_t$, and θ_t denotes the parameters of $\mathcal{G}_t$ at time t. Since

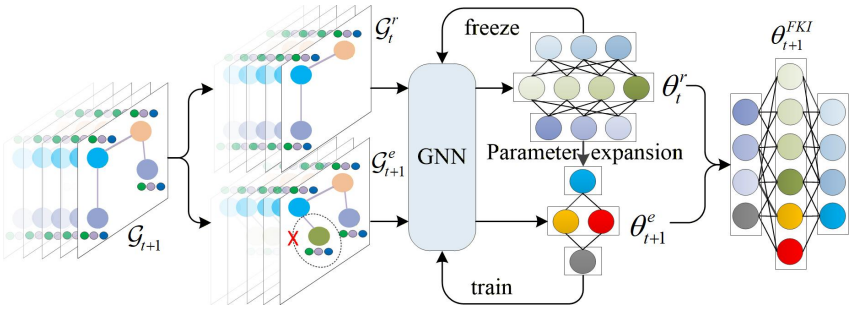


Figure 3: Graph-aware parameter expansion strategy.

the training process of DBKF-PE performs loss computation on a per-node basis, the loss terms in Eq. (2) can be directly added or subtracted (Zhang et al., 2023).

Subsequently, we expand the parameters θ_t^r to $\theta_{t+1}^G = \theta_t^r \oplus \theta_{t+1}^e$ (where θ_{t+1}^G denotes the parameters of $\mathcal{G}_{t+1}$ at time $t+1$, and θ_{t+1}^e denotes the parameters of $\mathcal{G}_{t+1}^e$ at time $t+1$), thereby enhancing the capability of DBKF-PE to learn new snapshot $\mathcal{G}_{t+1}$. Next, we freeze θ_t^r to prevent the overwriting of historical graph knowledge. Note that θ_t^r is not updated but is used to compute the loss during the forward propagation process (Kag & Saligrama, 2021). Based on this, we can calculate the loss $\mathcal{L}_{PE}$ of the graph-aware parameter expansion strategy according to Eq. (3).

$$\mathcal{L}_{PE} = \sum_{t=1}^{T} \mathcal{L}(F_{\theta_{t+1}^{FKI}}(\mathcal{G}_t^r) + F_{\theta_{t+1}^{FKI}}(\mathcal{G}_{t+1}^e)). \tag{3}$$

where θ_{t+1}^{FKI} denotes the parameters of the forward knowledge injection at time $t+1$. We optimize θ_{t+1}^{FKI} by minimizing the loss $\mathcal{L}_{PE}$ to enhance the model's learning capability for both historical and new snapshots.

Temporal Guidance Mechanism To capture the temporal relationships between snapshots, we propose a new temporal guidance mechanism that takes into account the state changes among nodes (e.g., addition of attributes or edges). In dynamic graphs, the state changes in a node are typically caused by changes in its neighbors at the previous snapshot (Kemertas et al., 2020). Specifically, for any node v_i at time $t+1$, we define its neighbor set at time t as $N_t(i) = \{v_j | \mathbf{A}_t[j,i] = 1\}$. We then define an edge change matrix $\Delta\mathbf{A}_{t+1}$ and an attribute change matrix $\Delta\mathbf{X}_{t+1}$ to capture temporal relationships between snapshots, as shown in Eq. (4):

$$\Delta\mathbf{A}_{t+1} = \mathbf{A}_{t+1} - \mathbf{A}_t, \Delta\mathbf{X}_{t+1} = \mathbf{X}_{t+1} - \mathbf{X}_t. \tag{4}$$

To identify the cause of the state change in node v_i at time $t+1$, we attribute it to changes in any neighbor node $v_j \in N_t(i)$ that satisfies one of the following conditions at time t. Temporal edge: The connection state between v_j and v_i changes, denoted as $\Delta\mathbf{A}_{t+1}[j,i] \neq 0$. Temporal attribute: The attribute of v_j itself changes, denoted as $\Delta\mathbf{X}_{t+1}[j,:] \neq \mathbf{0}$. Based on this, we can construct a temporal label vector $\mathbf{y}_{t+1}^{(c)} \in \{0,1\}^n$ for each v_i at time $t+1$, as shown in Eq. (5).

$$\mathbf{y}_{t+1}^{(c)}[j] = \begin{cases} 1, & \text{if } v_j \in N_t(i) \text{ and } (\Delta\mathbf{A}_{t+1}[j,i] \neq 0 \text{ or } \Delta\mathbf{X}_{t+1}[j,:] \neq \mathbf{0}) \\ 0, & \text{otherwise} \end{cases}, \tag{5}$$

where $\mathbf{y}_{t+1}^{(c)}$ explicitly indicates which changes in neighbor nodes ($N_t(i)$) at time t are potential causes for the state changes in node v_i at time $t+1$. We stack the temporal label vectors of all nodes into a temporal label matrix $\mathbf{Y}_{t+1}^{(c)} \in \{0,1\}^{n \times n}$ at time $t+1$. Subsequently, we design a temporal auxiliary task: given the node representations at the new snapshot, predict the temporal neighbors from the previous snapshot. Specifically, for a target node v_i, the model needs to predict the temporal score $\mathbf{S}_{i,j}^{t+1}$ of its temporal neighbor v_j at time $t+1$. We compute the temporal score matrix $\mathbf{S}$ using the representation $\mathbf{H}_i^{t+1}$ of v_i at time $t+1$ and the representation $\mathbf{H}_j^t$ of v_j at time t, as shown in Eq. (6):

$$\mathbf{S}_{i,j}^{t+1} = \sigma\left((\mathbf{H}_i^{t+1})^{\mathrm{T}}\mathbf{H}_j^t\right), \tag{6}$$

where σ denotes the Sigmoid. Subsequently, we employ the binary cross-entropy (Sankar et al., 2020) loss to measure the discrepancy between $\mathbf{Y}_{t+1}^{(c)}$ and $\mathbf{S}_{i,j}^{t+1}$, as shown in Eq. (7):

$$\mathcal{L}_{TAT} = -\frac{1}{n^2}\sum_{i=1}^{n}\sum_{j=1}^{n}\ [\mathbf{Y}_{t+1}^{(c)}[i,j]\log(\mathbf{S}_{i,j}^{t+1}) + (1-\mathbf{Y}_{t+1}^{(c)}[i,j])\log(1-\mathbf{S}_{i,j}^{t+1})], \tag{7}$$

where $\mathcal{L}_{TAT}$ denotes the temporal loss of the temporal auxiliary task. By minimizing this loss, DBKF-PE is compelled to learn the state changes among nodes to capture temporal relationships between snapshots.

3.3 Backward Knowledge Feedback

The core of the proposed BKFM is to construct a meta-learning optimization objective. This objective requires that the direction of parameter updates should enhance the new model's performance on all historical snapshots. By optimizing this objective, we obtain the final updated parameters θ_{t+1}. The objective is shown in Eq. (8):

$$\theta_{t+1} = \arg\min_{\theta} \mathcal{L}(F_\theta(\mathcal{G}_t^r) + F_\theta(\mathcal{G}_{t+1}^e)) + \lambda\mathcal{L}_{Meta}(\theta), \tag{8}$$

where $\mathcal{L}_{Meta}(\theta)$ denotes the meta-loss, λ denotes the meta-loss weight, and θ denotes the dummy variable parameter. We expect that the new model with θ_{t+1} can achieve superior performance over the model with the old parameters on historical snapshots. Therefore, we employ Eq. (9) to maximize the performance gain.

$$\mathcal{L}_{Meta}(\theta) = \sum_{i=1}^{t}\max(0, \mathcal{L}_{old_i}(\theta) - \mathcal{L}_{old_i}(\theta_t) + \gamma), \tag{9}$$

where $\mathcal{L}_{old_i}(\bullet)$ denotes the loss value computed by the model on historical snapshots, θ_t denotes the parameters of the model at time t, and γ denotes a margin hyperparameter. Our objective is to drive $\mathcal{L}_{old_i}(\theta) - \mathcal{L}_{old_i}(\theta_i) + \gamma$ to be negative, thereby achieving the performance gain. To better enable backward knowledge feedback, we allow the parameters of frozen snapshots to be synchronously updated when the new model achieves improved performance on historical snapshots. Otherwise, they remain frozen. The optimization steps are shown as follows: (1) Initialization: θ is initialized as θ_{t+1}^{FKI}. (2) Gradient computation: The gradient of the BKFM consists of two components: New snapshot gradient ($\nabla_\theta\mathcal{L} = \nabla_\theta\mathcal{L}(F_\theta(\mathcal{G}_t^r) + F_\theta(\mathcal{G}_{t+1}^e))|_{\theta=\theta_{t+1}^{FKI}}$) and Meta-gradient ($\nabla_\theta\mathcal{L}_{Meta} = \lambda\nabla_\theta\mathcal{L}_{Meta}(\theta)|_{\theta=\theta_{t+1}^{FKI}}$). (3) Parameter update: Perform the final update using the combined gradient, as shown in Eq. (10):

$$\theta_{t+1} = \theta_{t+1}^{FKI} - \beta[\nabla_\theta\mathcal{L}(F_\theta(\mathcal{G}_t^r) + F_\theta(\mathcal{G}_{t+1}^e)) + \lambda\nabla_\theta\mathcal{L}_{Meta}(\theta)]_{\theta=\theta_{t+1}^{FKI}}, \tag{10}$$

where β denotes the learning rate of the backward knowledge feedback. BKFM enables the model to not only mitigate catastrophic forgetting but also continuously improve its performance on historical snapshots, thereby enhancing its representation capabilities.

3.4 Overall Loss

The overall loss $\mathcal{L}_{DBKF-PE}$ of the DBKF-PE is shown in Eq. (11).

$$\mathcal{L}_{DBKF-PE} = \mathcal{L}_{PE} + \mathcal{L}_{TAT} + \lambda\mathcal{L}_{Meta}. \tag{11}$$

DBKF-PE achieves effective dynamic graph learning by minimizing $\mathcal{L}_{DBKF-PE}$. Furthermore, we equate the minimization of overall loss with the maximization of the joint information objective to demonstrate the solid theoretical foundation of proposed temporal guidance mechanism, as detailed in the Appendix A.1.

Table 1: Performance comparison of models for node classification. The best performance is shown in **bold**. The second-best performance is underlined.

Models	High School		Cell Phone		Cora		Citeseer		Photo		DBLP	
	PM	FM	PM	FM	PM	FM	PM	FM	PM	FM	PM	FM
EvolveGCN	78.85	-0.32	74.85	1.22	62.22	-3.14	67.12	-2.74	77.72	-4.52	61.52	-4.91
DyGNN	83.91	-0.11	78.89	2.34	55.37	-7.20	66.28	-6.50	68.34	-7.57	54.32	-8.62
DNE	84.56	0.93	79.93	2.34	50.50	-18.47	51.95	-16.72	40.34	-13.98	32.44	-15.62
ContinualGNN	90.48	0.73	80.39	2.33	74.65	-1.51	68.92	0.10	83.66	-5.93	62.67	-0.13
DiCGRL	93.02	1.03	81.39	1.09	62.93	-0.84	66.60	-1.15	71.23	-4.03	57.09	-3.24
TWP	92.32	0.32	83.72	2.09	77.05	-1.31	69.24	-1.53	86.76	-5.93	55.82	-5.24
PI-GNN	95.89	0.07	84.32	2.88	77.63	1.66	70.66	**0.49**	91.55	-3.66	64.76	-0.64
DGCN	94.59	1.03	85.52	1.98	75.94	1.92	70.55	-0.93	89.23	-0.35	62.93	-3.48
PiECL	95.03	1.56	86.79	3.98	77.04	0.87	66.62	-1.39	88.67	-1.73	62.93	-2.84
DG-Mamba	96.84	0.93	90.72	4.72	78.49	0.23	66.84	-5.69	89.94	-2.83	64.39	1.88
MSGCL	95.59	1.02	90.89	5.43	76.48	1.83	67.95	0.11	90.62	0.82	60.12	-1.95
GDDSG	94.31	1.32	88.98	6.34	76.87	1.67	68.82	0.11	91.17	0.34	64.78	1.66
FSTG	94.48	1.20	89.59	4.92	74.57	1.28	67.58	-0.22	91.74	1.03	65.59	1.67
SFDyG	95.68	1.08	88.59	6.44	76.69	1.66	69.94	-1.67	**93.04**	0.46	63.89	1.56
DBKF-PE	**97.08**	**2.01**	**92.80**	**7.61**	**79.41**	**2.36**	**71.93**	0.32	92.23	**1.59**	**67.68**	**2.13**

Table 2: Performance comparison of models for community detection. The best performance is shown in **bold**. The second-best performance is underlined.

Models	High School		Cell Phone		Cora		Citeseer		Photo		DBLP	
	NMI	ARI	NMI	ARI	NMI	ARI	NMI	ARI	NMI	ARI	NMI	ARI
EvolveGCN	67.27	56.19	37.45	11.81	15.61	10.58	23.97	8.98	34.32	12.49	11.08	3.04
DyGNN	66.49	53.71	37.18	13.39	27.79	11.13	26.45	6.00	23.77	11.55	9.03	2.88
DNE	68.52	47.67	34.13	28.42	20.78	12.89	22.07	7.69	33.23	13.72	11.94	2.83
ContinualGNN	73.93	72.33	53.03	38.29	28.75	11.25	29.97	11.51	38.04	17.22	13.74	3.37
DiCGRL	76.22	56.93	46.20	37.43	30.08	9.08	23.56	10.04	36.68	17.72	11.79	4.45
TWP	76.43	78.93	52.90	42.89	33.20	10.80	29.99	12.09	37.84	18.86	12.39	3.57
PI-GNN	80.93	78.07	46.67	32.46	18.28	13.28	15.42	4.78	31.74	14.67	14.32	4.89
DGCN	71.77	58.56	29.73	9.84	24.16	15.11	10.07	4.45	11.78	3.96	15.78	6.69
PiECL	76.58	77.93	51.02	41.03	33.78	10.92	31.88	11.03	38.90	15.65	15.50	6.04
DG-Mamba	81.40	79.23	58.03	46.23	35.89	**20.03**	33.43	12.49	48.33	20.23	16.44	5.99
MSGCL	70.14	56.56	39.05	15.21	30.01	15.35	24.64	**15.29**	45.77	19.88	10.41	2.48
GDDSG	77.63	74.34	53.84	42.73	31.82	16.29	32.55	11.99	46.84	22.73	14.63	5.83
FSTG	73.64	80.05	56.37	46.29	33.72	17.39	32.07	11.81	38.21	21.55	14.42	6.65
SFDyG	78.62	76.22	57.18	43.62	30.39	15.62	33.45	12.13	46.79	23.78	13.56	5.03
DBKF-PE	**84.50**	**82.51**	**59.89**	**47.25**	**36.15**	19.69	**35.33**	14.78	**50.24**	**26.95**	**19.01**	**7.48**

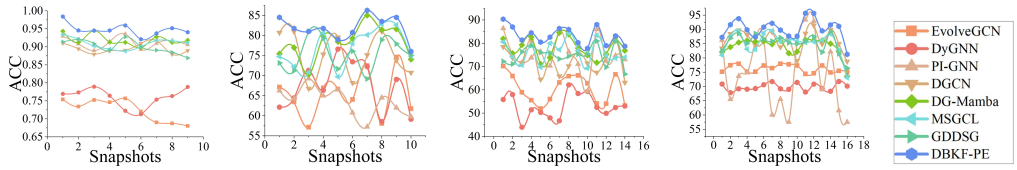
4 EXPERIMENTS

4.1 EXPERIMENTAL ENVIRONMENT

Benchmark Datasets The datasets include High School (Gao et al., 2023), Cell Phone (Gao et al., 2023), Cora (McCallum et al., 2000), Citeseer (Sen et al., 2008), Photo (Shirzad et al., 2023), and DBLP (Tang et al., 2008). These datasets are all benchmark datasets in the field of DGL and can effectively evaluate model performance. The relevant information of the benchmark datasets is listed in the Appendix A.2.1.

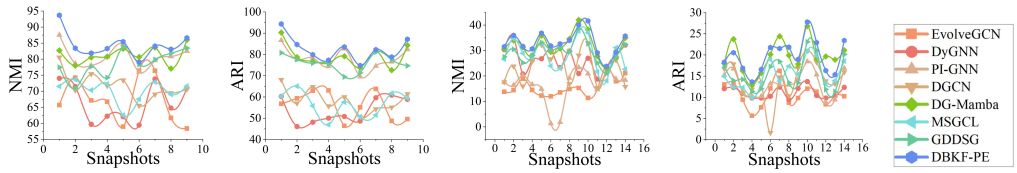
Baselines We evaluated the performance of DBKF-PE by comparing it with the current mainstream continual learning methods and DGL methods. (1) Continual learning methods: ContinualGNN (Wang et al., 2020), DiCGRL (Kou et al., 2020), TWP (Liu et al., 2021), PI-GNN (Zhang et al., 2023), PiECL (Peng et al., 2024), and GDDSG (Lai et al., 2025). (2) DGL methods: EvolveGCN (Pareja et al., 2020), DyGNN (Ma et al., 2020), DNE (Du et al., 2018), DGCN (Gao et al., 2023), DG-Mamba (Yuan et al., 2025), MSGCL (Teng et al., 2026), FSTG (Liu et al., 2026), and SFDyG (Qi et al., 2025).

Implementation Details The code environment for this experiment was Python 3.8. DBKF-PE was trained on a server with 512GB of memory, an X86_64 architecture processor, and the Ubuntu 20.04.06 LTS operating system. The GPU used was NVIDIA A800. The GNN model employed



(a) High School (b) Cell Phone (c) Cora (d) Photo

Figure 4: The experiment results of backward knowledge feedback for node classification. ACC denotes accuracy.



(a) High School-NMI (b) High School-ARI (c) Cora-NMI (d) Cora-ARI

Figure 5: The experiment results of backward knowledge feedback for community detection.

was a two-layer GraphSAGE (SAmple and aggreGatE) (Hamilton et al., 2017). μ, γ and β were set to 0.01, 0.05 and 0.001, respectively. Next, for node classification, we strictly adhered to the experimental protocols established by PI-GNN (Parameter Isolation GNN) (Zhang et al., 2023). For community detection, the Louvain (Blondel et al., 2008) was employed as the downstream model, following the experimental protocols established by Louvain. For the parameter experiments on the expansion units (the number of parameter expansion) and the meta-loss parameter λ, please refer to Appendix A.2.2. In addition, we adopted model distillation (Zhou et al., 2021) to address the parameter surge problem of DBKF-PE. Experimental results demonstrate that the distilled model exhibits a performance degradation of less than 2% at most, please refer to Appendix A.2.3. For node classification, we utilized Performance Mean (PM) and Forgetting Measure (FM) (Chaudhry et al., 2018) to evaluate the model's capability in mitigating catastrophic forgetting. $\mathrm{FM} < 0$ indicates forgetting, whereas $\mathrm{FM} > 0$ indicates no forgetting. For community detection, we utilized Normalized Mutual Information (NMI) and Adjusted Rand Index (ARI) (Chakraborty et al., 2017) to evaluate the detection performance of the model.

4.2 Experimental Analysis for Downstream Tasks

Experimental Analysis for Node Classification Table 1 presents the performance comparison of DBKF-PE and the baselines for node classification. As listed in Table 1, DBKF-PE outperforms the baselines in overall performance. Specifically, DBKF-PE achieves the best performance on High School, Cell Phone, Cora, and DBLP. The main reason is that DBKF-PE employs the proposed graph-aware parameter expansion strategy to learn rational subgraphs and environmental subgraphs between different snapshots, effectively enhancing the model's ability to learn various graph patterns. In addition, DBKF-PE employs the proposed temporal guidance mechanism to capture the temporal evolution information of dynamic graphs, learning the temporal relationships between nodes and attributes in a fine-grained manner. It is worth noting that DBKF-PE has a lower PM value on Photo than SFDyG. The main reason is that SFDyG integrates multiple snapshots into a single temporal graph, reducing the sparsity of the dynamic graph. DBKF-PE also has a lower FM value on Citeseer than PI-GNN. The main reason is that PI-GNN utilizes the continuous learning to mitigate the catastrophic forgetting, decreasing the model's probability of forgetting historical information.

Experimental Analysis for Community Detection Table 2 presents the performance comparison of DBKF-PE and the baselines for community detection. As listed in Table 2, DBKF-PE outperforms the baselines in overall performance. Specifically, DBKF-PE achieves the best performance on High School, Cell Phone, Photo, and DBLP. The reason is similar to that for node classification,

Table 3: Ablation study for node classification. The best performance is shown in **bold**. The second-best performance is underlined. PE denotes Parameter Expansion. TG denotes Temporal Guidance mechanism. BK denotes BKFM.

Models	High School		Cell Phone		Cora		Citeseer		Photo		DBLP	
	PM	FM	PM	FM	PM	FM	PM	FM	PM	FM	PM	FM
DBKF-PE w/o PE	85.69	0.08	65.27	2.11	66.01	1.16	59.60	-2.51	74.15	0.47	51.95	-2.28
DBKF-PE w/o TG	94.02	1.33	88.18	6.72	73.20	1.33	62.12	0.06	81.27	1.23	58.32	-1.59
DBKF-PE w/o BK	96.04	1.41	89.80	5.44	75.03	1.63	67.33	0.14	91.83	1.27	57.95	1.23
DBKF-PE	**97.08**	**2.01**	**92.80**	**7.61**	**79.41**	**2.36**	**71.93**	**0.32**	**92.23**	**1.59**	**67.68**	**2.13**

which involves leveraging the graph-aware parameter expansion strategy and the temporal guidance mechanism to enhance the accuracy of node representation. This allows similar nodes to be clustered more effectively, forming stable communities. It is worth noting that the DBKF-PE has a lower ARI value on Cora than DG-Mamba. The main reason is that DG-Mamba leverages Mamba's temporal capture capability to enhance the consistency of nodes across all snapshots, thereby improving the accuracy of community detection. Additionally, DBKF-PE has a lower ARI value on Citeseer than MSGCL. The main reason is that MSGCL utilizes graph contrastive learning to strengthen the dependency between nodes and communities, reducing the error in calculating node membership.

4.3 The Experiment of Backward Knowledge Feedback

To evaluate the backward knowledge feedback capability of model, we separately used models trained on all snapshots of High School, Cell Phone, Cora, and Photo to learn their corresponding historical snapshots and record their experimental results, as shown in Figures 4 and 5. The community detection results of the model on the Cell Phone and Photo are shown in Appendix A.2.4. From Figs. 4 and 5, the performance of DBKF-PE is overall superior to that of the comparative baselines. The main reason is that DBKF-PE employs the BKFM to enhance historical graph knowledge with knowledge from new graphs. This enables the new model to recognize graph patterns that the old model could not recognize, thereby improving learning performance of the new model on historical graphs. Notably, unlike continual learning models that focus solely on mitigating catastrophic forgetting (e.g., PI-GNN and GDDSG), DBKF-PE not only avoids performance degradation on historical snapshots but also achieves performance improvement. For instance, as listed in Table 1, the FM value of GDDSG on Cora is greater than 0, indicating that no forgetting occurs. However, Figs. 4 and 5 show that the performance of GDDSG on both node classification and community detection tasks is inferior to that of DBKF-PE. This indicates that BKFM not only achieves knowledge retention but also facilitates knowledge enhancement.

4.4 Ablation Study

To evaluate the contribution of each component to the overall performance of DBKF-PE, we conducted ablation studies on node classification and community detection. The ablation results for node classification are listed in Table 3. The ablation results for community detection are shown in Appendix A.2.5. As listed in Table 3, compared to the performance of DBKF-PE, the performance of DBKF-PE w/o PE degrades the most, followed by DBKF-PE w/o TG, and then DBKF-PE w/o BKFM. This indicates that the graph-aware parameter expansion strategy has the greatest impact on DBKF-PE, while also demonstrating that each component is an indispensable part of DBKF-PE.

5 Conclusion

The proposed graph-aware parameter expansion strategy expands new parameters to learn environmental subgraph knowledge from the new graph while freezing old parameters to enhance the model's representation capability for dynamic graphs. Meanwhile, we employ the proposed temporal guidance mechanism to capture the temporal relationships of dynamic graphs during the process of parameter expansion. More importantly, our proposed BKFM propagates newly acquired graph knowledge back to historical graphs, breaking the unidirectional knowledge flow paradigm in traditional DGL and providing a novel theoretical perspective for DGL.

AI USE STATEMENT

In this work, we did not use generative AI tools for any tasks with required disclosure. We have not used generative AI tools for generating synthetic datasets, developing theoretical models or conceptual frameworks, formulating mathematical claims, providing critical ingredients for proving mathematical claims, assisting in the writing of proofs, proposing or refining hypotheses, designing or providing feedback on research methodology or experiments, implementing methods, assisting with translation, cleaning and reformatting datasets, supporting qualitative and thematic data analysis, or interpreting results. No other required-disclosure tasks are applicable to this work.

Additionally, we used generative AI tools for a task with recommended disclosure: editing the research paper to improve readability, i.e., language polishing. The tools were used only to suggest improvements to grammar, wording, clarity, and style. They were not used to generate or alter the scientific content, research ideas, hypotheses, data, code, analyses, figures, references, or conclusions. We have reviewed all AI-assisted work. All AI-assisted edits were checked and revised by the authors to ensure accuracy, originality, and fidelity to the authors' intended meaning. We take responsibility for the final content of this work, including text, claims, or artifacts produced with the aid of generative AI.

REPRODUCIBILITY STATEMENT

To ensure reproducibility, we provide an anonymous repository at `https://anonymous.4open.science/r/DBKF-PE-Y-BD07A`. This repository contains our code, datasets, data preprocessing scripts, model configurations, and detailed instructions for reproducing the main experimental results. All necessary implementation details, hyperparameters, and evaluation procedures are included in the repository.

REFERENCES

Yuanchen Bei, Hao Xu, Sheng Zhou, Huixuan Chi, Haishuai Wang, Mengdi Zhang, Zhao Li, and Jiajun Bu. Cpdg: A contrastive pre-training method for dynamic graph neural networks. In *Proceedings of the 40th IEEE International Conference on Data Engineering*, pp. 1199–1212, 2024.

Vincent D Blondel, Jean-Loup Guillaume, Renaud Lambiotte, and Etienne Lefebvre. Fast unfolding of communities in large networks. *Journal of Statistical Mechanics: Theory and Experiment*, 2008(10):P10008, 2008.

Tanmoy Chakraborty, Ayushi Dalmia, Animesh Mukherjee, and Niloy Ganguly. Metrics for community analysis: A survey. *ACM Computing Surveys*, 50(4):1–37, 2017.

Arslan Chaudhry, Puneet Kumar Dokania, Thalaiyasingam Ajanthan, and Philip H S Torr. Riemannian walk for incremental learning: Understanding forgetting and intransigence. In *Proceedings of the 15th European Conference on Computer Vision*, pp. 556–572, 2018.

Chao Chen, Haoyu Geng, Nianzu Yang, Xiaokang Yang, and Junchi Yan. Easydgl: Encode, train and interpret for continuous-time dynamic graph learning. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 46(12):10845–10862, 2024.

Junwei Cheng, Chaobo He, Pengxing Feng, Weixiong Liu, Kunlin Han, and Yong Tang. Community-aware variational autoencoder for continuous dynamic networks. In *Proceedings of the 39th AAAI Conference on Artificial Intelligence*, volume 39, pp. 16001–16009, 2025.

Lun Du, Yun Wang, Guojie Song, Zhicong Lu, and Junshan Wang. Dynamic network embedding: An extended approach for skip-gram based network embedding. In *Proceedings of the 27th International Joint Conference on Artificial Intelligence*, pp. 2086–2092, 2018.

Soheila Farokhi, Xiaojun Qi, and Hamid Karimi. Tawrmac: A novel dynamic graph representation learning method. In *Proceedings of the 35th ACM Web Conference*, pp. 684–695, 2026.

ZhengZhao Feng, Rui Wang, Longjiao Zhang, Tongya Zheng, Ziqi Huang, and Mingli Song. Efficient dynamic graph learning with refined batch parallel training. In *Proceedings of the 34th International Joint Conference on Artificial Intelligence*, pp. 2802–2810, 2025.

Chao Gao, Junyou Zhu, Fan Zhang, Zhen Wang, and Xuelong Li. A novel representation learning for dynamic graphs based on graph convolutional networks. *IEEE Transactions on Cybernetics*, 53(6):3599–3612, 2023.

Alessio Gravina and Davide Bacciu. Deep learning for dynamic graphs: models and benchmarks. *IEEE Transactions on Neural Networks and Learning Systems*, 35(9):11788–11801, 2024.

William Hamilton, Zhitao Ying, and Jure Leskovec. Inductive representation learning on large graphs. In *Proceedings of the 31st Annual Conference on Neural Information Processing Systems*, pp. 1024–1034, 2017.

Yiming He, Xiang Li, Zhongying Zhao, Haobing Liu, Peilan He, and Yanwei Yu. Maskdgnn: Self-supervised dynamic graph neural networks with activeness-aware temporal masking. In *Proceedings of the 34th International Joint Conference on Artificial Intelligence*, pp. 2892–2900, 2025.

Xiaobin Hong, Mingkai Lin, Xiaoli Wang, Chaoqun Wang, and Wenzhong Li. Multimodal graph representation learning with dynamic information pathways. In *Proceedings of the 40th AAAI Conference on Artificial Intelligence*, volume 40, pp. 14829–14837, 2026.

Xing Huang, Dandan Lin, Weiyi Huang, Shijie Sun, Jie Wen, and Chuan Chen. Platod2gl: An efficient dynamic deep graph learning system for graph neural network training on billion-scale graphs. In *Proceedings of the 40th IEEE International Conference on Data Engineering*, pp. 2421—-2434, 2024.

Anil Kag and Venkatesh Saligrama. Training recurrent neural networks via forward propagation through time. In *Proceedings of the 38th International Conference on Machine Learning*, pp. 5189–5200, 2021.

Mete Kemertas, Leila Pishdad, Konstantinos G Derpanis, and Afsaneh Fazly. Rankmi: A mutual information maximizing ranking loss. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 14362–14371, 2020.

Seoyoon Kim, Seongjun Yun, and Jaewoo Kang. Dygrain: An incremental learning framework for dynamic graphs. In *Proceedings of the 31st International Joint Conference on Artificial Intelligence*, pp. 3157–3163, 2022.

Xiaoyu Kou, Yankai Lin, Shaobo Liu, Peng Li, Jie Zhou, and Yan Zhang. Disentangle-based continual graph representation learning. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing*, pp. 2961–2972, 2020.

Guannan Lai, Yujie Li, Xiangkun Wang, Junbo Zhang, Tianrui Li, and Xin Yang. Order-robust class incremental learning: Graph-driven dynamic similarity grouping. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Conference*, pp. 4894–4904, 2025.

Huihui Liu, Yiding Yang, and Xinchao Wang. Overcoming catastrophic forgetting in graph neural networks. In *Proceedings of the 35th AAAI Conference on Artificial Intelligence*, pp. 8653—-8661, 2021.

Yezi Liu, Jiaxuan Xie, and Yanning Shen. Dgexplainer: Explaining dynamic graph neural networks via relevance back-propagation. In *Proceedings of the 34th International Joint Conference on Artificial Intelligence*, pp. 3135–3143, 2025.

Yuxuan Liu, Wenchao Xu, Haozhao Wang, Zhiming He, Zhaofeng Shi, Chongyang Xu, Peichao Wang, and Boyuan Zhang. Causality-inspired federated learning for dynamic spatio-temporal graphs. In *Proceedings of the 40th AAAI Conference on Artificial Intelligence*, pp. 23926–23934, 2026.

Yao Ma, Ziyi Guo, Zhaochun Ren, Jiliang Tang, and Dawei Yin. Streaming graph neural networks. In *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval*, pp. 719–728, 2020.

Ivan Marisca, Andrea Cini, and Cesare Alippi. Learning to reconstruct missing data from spatiotemporal graphs with sparse observations. In *Proceedings of the 36th Annual Conference on Neural Information Processing Systems*, pp. 32069–32082, 2022.

Andrew Kachites McCallum, Kamal Nigam, Jason Rennie, and Kristie Seymore. Automating the construction of internet portals with machine learning. *Information Retrieval*, 3(2):127–163, 2000.

Jianye Pang, Xinjie Zhu, and Xiaofei Xiong. Dyfmvp: Say goodbye to staleness! fresh memory vigorous preserver for continuous-time dynamic graph. In *Proceedings of the 41st International Conference on Data Engineering*, pp. 1622—-1635, 2025.

Aldo Pareja, Giacomo Domeniconi, Jie Chen, Tengfei Ma, Toyotaro Suzumura, Hiroki Kanezashi, Tim Kaler, Tao Schardl, and Charles Leiserson. Evolvegcn: Evolving graph convolutional networks for dynamic graphs. In *Proceedings of the 34th AAAI Conference on Artificial Intelligence*, volume 34, pp. 5363–5370, 2020.

Ciyuan Peng, Tao Tang, Qiuyang Yin, Xiaomei Bai, Suryani Lim, and Charu C Aggarwal. Physics-informed explainable continual learning on graphs. *IEEE Transactions on Neural Networks and Learning Systems*, 35(9):11761–11772, 2024.

Farimah Poursafaei, Shenyang Huang, Kellin Pelrine, and Reihaneh Rabbany. Towards better evaluation for dynamic link prediction. In *Proceedings of the 36th Annual Conference on Neural Information Processing Systems*, pp. 32928–32941, 2022.

QingGuo Qi, Hongyang Chen, Minhao Cheng, and Han Liu. Input snapshots fusion for scalable discrete-time dynamic graph neural networks. In *Proceedings of the 31st ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pp. 1138–1149, 2025.

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 13rd ACM International Conference on Web Search and Data Mining*, pp. 519–527, 2020.

Prithviraj Sen, Galileo Namata, Mustafa Bilgic, Lise Getoor, Brian Gallagher, and Tina Eliassi-Rad. Collective classification in network data. *AI Magazine*, 29(3):93–93, 2008.

Hamed Shirzad, Ameya Velingker, Balaji Venkatachalam, Danica J Sutherland, and Ali Kemal Sinop. Exphormer: Sparse transformers for graphs. In *Proceedings of the 40th International Conference on Machine Learning*, pp. 31613–31632, 2023.

Jie Tang, Jing Zhang, Limin Yao, Juanzi Li, Li Zhang, and Zhong Su. Arnetminer: extraction and mining of academic social networks. In *Proceedings of the 14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 990–998, 2008.

Min Teng, Chao Gao, Xianghua Li, Zhen Wang, Kefeng Fan, and Vladimir Nekorkin. Multi-scale graph contrastive learning for community detection in dynamic graphs. *Information Processing & Management*, 63(2):104410, 2026.

Rakshit Trivedi, Mehrdad Farajtabar, Prasenjeet Biswal, and Hongyuan Zha. Dyrep: Learning representations over dynamic graphs. In *Proceedings of 7th International Conference on Learning Representation*, 2019.

Junshan Wang, Guojie Song, Yi Wu, and Liang Wang. Streaming graph neural networks via continual learning. In *Proceedings of the 29th ACM International Conference on Information and Knowledge Management*, pp. 1515–1524, 2020.

Yuxia Wu, Lizi Liao, and Yuan Fang. Retrieval augmented generation for dynamic graph modeling. In *Proceedings of the 48th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pp. 1434–1443, 2025.

Kaiwen Xia, Li Lin, Shuai Wang, Qi Zhang, Shuai Wang, and Tian He. Prost: Prompt future snapshot on dynamic graphs for spatio-temporal prediction. In *Proceedings of the 31st ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pp. 1645–1656, 2025.

Jiyuan Xu, Wenyu Zhang, Xin Jing, Shuai Chen, Shuai Zhang, and Jiahao Nie. Cpiri: Channel permutation-invariant relational interaction for multivariate time series forecasting. In *arXiv preprint arXiv:2601.20318*, 2026.

Xiao Yang, Xuejiao Zhao, and Zhiqi Shen. A generalizable anomaly detection method in dynamic graphs. In *Proceedings of the 39th AAAI Conference on Artificial Intelligence*, volume 39, pp. 22001–22009, 2025.

Haonan Yuan, Qingyun Sun, Zhaonan Wang, Xingcheng Fu, Cheng Ji, Yongjian Wang, Bo Jin, and Jianxin Li. Dg-mamba: Robust and efficient dynamic graph structure learning with selective state space models. In *Proceedings of the 39th AAAI Conference on Artificial Intelligence*, volume 39, pp. 22272–22280, 2025.

Guibin Zhang, Yiyan Qi, Ziyang Cheng, Yanwei Yue, Dawei Cheng, and Jian Guo. Rationalizing and augmenting dynamic graph neural networks. In *Proceedings of the 13rd International Conference on Learning Representations*, 2025.

Peiyan Zhang, Yuchen Yan, Chaozhuo Li, Senzhang Wang, Xing Xie, Guojie Song, and Sunghun Kim. Continual learning on dynamic graphs via parameter isolation. In *Proceedings of the 46th international ACM SIGIR Conference on Research and Development in Information Retrieval*, pp. 601–611, 2023.

Zhenghao Zhang, Jun Xie, Xingchen Chen, Tao Yu, Hongzhu Yi, Kaixin Xu, Yuanxiang Wang, Tianyu Zong, Xinming Wang, Jiahuan Chen, Guoqing Chao, Feng Chen, Zhepeng Wang, and Jungang Xu. Dynamic deep graph learning for incomplete multi-view clustering with masked graph reconstruction loss. In *Proceedings of the 40th AAAI Conference on Artificial Intelligence*, pp. 28600–28608, 2026.

Sheng Zhou, Yucheng Wang, Defang Chen, Jiawei Chen, Xin Wang, Can Wang, and Jiajun Bu. Distilling holistic knowledge with graph neural networks. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 10367–10376, 2021.

A Appendix

A.1 Theoretical Analysis of Temporal Guidance Mechanism

DBKF-PE achieves effective dynamic graph learning by minimizing $\mathcal{L}_{DBKF-PE}$. Furthermore, we equate the minimization of overall loss with the maximization of the joint information objective to demonstrate the solid theoretical foundation of proposed temporal guidance mechanism, as shown in Eq. (12):

$$\min_{\theta} \mathcal{L}_{DBKF-PE} \Leftrightarrow \max_{\theta} (I(\mathbf{H}^{t+1}; \mathbf{Y}_{t+1}^{(label)}) + I(\mathbf{H}^{t+1}; \mathbf{Y}_{t+1}^{(c)})), \tag{12}$$

where $I(\bullet;\bullet)$ denotes mutual information, $\mathbf{H}^{t+1}$ denotes the node representation at time $t+1$, and $\mathbf{Y}_{t+1}^{(label)}$ denotes the true label matrix at time $t+1$.

Subsequently, we evaluate the primary task loss ($\mathcal{L}_{PE} + \lambda\mathcal{L}_{Meta}$) by taking the cross-entropy loss of node classification as an analytical example. The objective of minimizing the cross-entropy loss is to make the distribution $p(\mathbf{Y}_{t+1}^{(label)}|\mathbf{H}^{t+1})$ predicted by the model approximate the true label distribution. In information theory, this objective is equivalent to maximizing the mutual information $I(\mathbf{H}^{t+1}; \mathbf{Y}_{t+1}^{(label)})$ between $\mathbf{H}^{t+1}$ and $\mathbf{Y}_{t+1}^{(label)}$. The definition of $I(\mathbf{H}^{t+1}; \mathbf{Y}_{t+1}^{(label)})$ is shown in Eq. (13):

$$I(\mathbf{H}^{t+1}; \mathbf{Y}_{t+1}^{(label)}) = \mathbb{E}_{p(\mathbf{h}^{t+1}, \mathbf{y}_{t+1}^{(label)})}[\log \frac{p(\mathbf{y}_{t+1}^{(label)}|\mathbf{h}^{t+1})}{p(\mathbf{y}_{t+1}^{(label)})}], \tag{13}$$

Table A1: Information of the benchmark datasets.

Dataset	Node	Edge	Attribute	Snapshot
High School	327	188508	654	9
Cell Phone	400	9834	800	10
Cora	2708	5429	1433	14
Citeseer	3327	4732	3703	12
Photo	7650	119043	745	16
DBLP	20000	75706	128	26

where $\mathbb{E}$ denotes mathematical expectation, $p(\bullet)$ denotes probability distribution, and $\mathbf{h}^{t+1}$/$\mathbf{y}_{t+1}^{(label)}$ denotes a subvector of $\mathbf{H}^{t+1}$/$\mathbf{Y}_{t+1}^{(label)}$.

Next, we evaluate the temporal loss $\mathcal{L}_{TAT}$, which is equivalent to maximizing the mutual information $I(\mathbf{H}^{t+1}; \mathbf{Y}_{t+1}^{(c)})$. The temporal auxiliary task is formulated as a binary classification problem. Therefore, we employ the binary cross-entropy (BCE) as the loss function. The temporal loss $\mathcal{L}_{TAT}^{(i,j)}$ for a single node pair (i, j) is shown in Eq. (14):

$$\mathcal{L}_{TAT}^{(i,j)} = -[\mathbf{Y}_{t+1}^{(c)}[i,j] \log(\mathbf{S}_{i,j}^{t+1}) + (1 - \mathbf{Y}_{t+1}^{(c)}[i,j]) \log(1 - \mathbf{S}_{i,j}^{t+1})], \tag{14}$$

where $\mathbf{S}_{i,j}^{t+1} = p(\mathbf{Y}_{t+1}^{(c)}[i,j] = 1 | \mathbf{h}_i^{t+1}, \mathbf{h}_j^t)$ denotes the predicted probability (temporal score), and $\mathbf{h}_i^{t+1}$/$\mathbf{h}_j^t$ denotes the representation of v_i/v_j at time $(t+1)$/t. Building on this, we average the temporal loss for n^2 node pairs according to Eq. (15).

$$\mathcal{L}_{TAT} = -\frac{1}{n^2} \sum_{i=1}^{n} \sum_{j=1}^{n} \mathbb{E}_{p(\mathbf{h}_i^{t+1}, \mathbf{h}_j^t)} [\log p(\mathbf{y}_{t+1}^{(c)}[i,j] | \mathbf{h}_i^{t+1}, \mathbf{h}_j^t)]. \tag{15}$$

From Eq. (15), it can be observed that the BCE loss is essentially the negative log-likelihood function under the independent and identically distributed assumption. The mutual information $I(\mathbf{H}^{t+1}; \mathbf{Y}_{t+1}^{(c)})$ of the temporal loss is shown in Eq. (16):

$$\begin{aligned} I(\mathbf{H}^{t+1}; \mathbf{Y}_{t+1}^{(c)}) &= \mathbb{E}_{p(\mathbf{h}^{t+1}, \mathbf{y}_{t+1}^{(c)})} [\log \frac{p(\mathbf{y}_{t+1}^{(c)} | \mathbf{h}^{t+1})}{p(\mathbf{y}_{t+1}^{(c)})}] \\ &= \mathbb{E}_{p(\mathbf{h}^{t+1}, \mathbf{y}_{t+1}^{(c)})} [\log p(\mathbf{y}_{t+1}^{(c)} | \mathbf{h}^{t+1})] \\ &\quad - \mathbb{E}_{p(\mathbf{y}_{t+1}^{(c)})} [\log p(\mathbf{y}_{t+1}^{(c)})], \end{aligned} \tag{16}$$

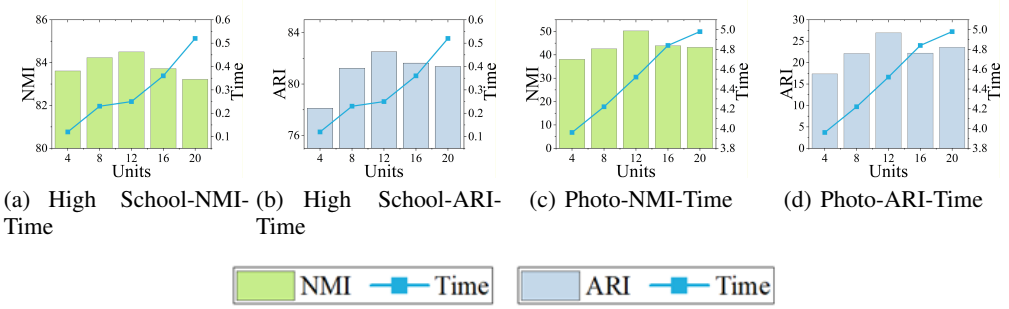
where the second term $H(\mathbf{Y}_{t+1}^{(c)}) = -\mathbb{E}_{p(\mathbf{y}_{t+1}^{(c)})} [\log p(\mathbf{y}_{t+1}^{(c)})]$ denotes the entropy of $\mathbf{Y}_{t+1}^{(c)}$. Since $H(\mathbf{Y}_{t+1}^{(c)})$ is independent of the model parameters, $H(\mathbf{Y}_{t+1}^{(c)})$ is a constant when optimizing the model parameters. Therefore, maximizing $I(\mathbf{H}^{t+1}; \mathbf{Y}_{t+1}^{(c)})$ is equivalent to maximizing $\mathbb{E}_{p(\mathbf{h}^{t+1}, \mathbf{y}_{t+1}^{(c)})} [\log p(\mathbf{y}_{t+1}^{(c)} | \mathbf{h}^{t+1})]$, as shown in Eq. (17):

$$\max_{\theta} I(\mathbf{H}^{t+1}; \mathbf{Y}_{t+1}^{(c)}) \Leftrightarrow \max_{\theta} \mathbb{E}_{p(\mathbf{h}^{t+1}, \mathbf{y}_{t+1}^{(c)})} [\log p(\mathbf{y}_{t+1}^{(c)} | \mathbf{h}^{t+1})]. \tag{17}$$

Given the monotonicity of the logarithmic function, maximizing the expected log-likelihood function is equivalent to minimizing its negative value, as shown in Eq. (18):

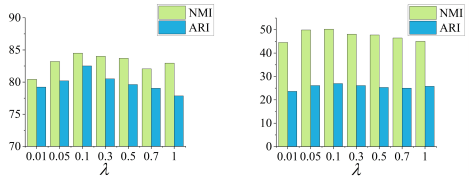
$$\max_{\theta} \mathbb{E}_{p(\mathbf{h}^{t+1}, \mathbf{y}_{t+1}^{(c)})} [\log p(\mathbf{y}_{t+1}^{(c)} | \mathbf{h}^{t+1})] \Leftrightarrow \min_{\theta} -\mathbb{E}_{p(\mathbf{h}^{t+1}, \mathbf{y}_{t+1}^{(c)})} [\log p(\mathbf{y}_{t+1}^{(c)} | \mathbf{h}^{t+1})]. \tag{18}$$

At this point, the right-hand side of the equation represents the expected form of the temporal loss $\mathcal{L}_{TAT}$. Thus, it is proven that minimizing the overall loss is equivalent to maximizing the joint information objective.



(a) High School-NMI-Time (b) High School-ARI-Time (c) Photo-NMI-Time (d) Photo-ARI-Time

Figure A1: Experimental results of expansion units (Units). The unit of Time is seconds.



(a) High School-λ (b) Photo-λ

Figure A2: Experimental results of the meta-loss parameter.

Table A2: Experimental results of model compression for node classification.

Models	High School		Cell Phone		Cora		Citeseer		Photo		DBLP	
	PM	FM	PM	FM	PM	FM	PM	FM	PM	FM	PM	FM
Distilled DBKF-PE	96.63	1.87	92.18	7.01	78.67	1.64	70.47	0.27	91.11	1.23	67.21	1.89
DBKF-PE	97.08	2.01	92.80	7.61	79.41	2.36	71.93	0.32	92.23	1.59	67.68	2.13

A.2 Experimental Analysis

A.2.1 Benchmark Datasets

The relevant information of the benchmark datasets is listed in Table A1.

A.2.2 Parameter Experiments

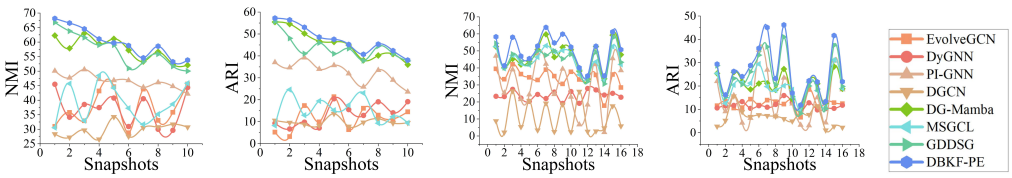
In this section, we took community detection as an example to conduct parameter experiments on High School and Photo regarding expansion units (the number of parameter expansion) and meta-loss parameter λ. The experimental results are shown in Figs. A1 and A2.

From Fig. A1, different numbers of the expansion units have varying impacts on the performance of DBKF-PE. Increasing the number of expansion units does not improve the performance of DBKF-PE. Instead, it continually increases the training time cost. Therefore, we set the number of expansion units to 12 to balance model performance and time cost.

From Fig. A2, when λ is set to 0.1, DBKF-PE achieves the best NMI and ARI on both the High School and Photo. Different values of λ have varying impacts on the performance of DBKF-PE. Consequently, identifying the appropriate value of λ is crucial for optimizing DBKF-PE.

A.2.3 Model Compression Experiments

The parameter count of DBKF-PE was influenced by the number of T. If T increased sharply, it might lead to a sharp increase in the parameter count of DBKF-PE. Therefore, we adopted model distillation to solve the problem of parameter surge. The experimental results are listed in Tables A2 and A3.



(a) Cell Phone-NMI (b) Cell Phone-ARI (c) Photo-NMI (d) Photo-ARI

Figure A3: The experiment results of backward knowledge feedback for community detection.

Table A3: Experimental results of model compression for community detection.

Models	High School		Cell Phone		Cora		Citeseer		Photo		DBLP	
	NMI	ARI	NMI	ARI	NMI	ARI	NMI	ARI	NMI	ARI	NMI	ARI
Distilled DBKF-PE	83.80	81.68	59.25	46.48	35.29	18.83	34.56	14.31	49.50	26.39	18.37	6.73
DBKF-PE	84.50	82.51	59.89	47.25	36.15	19.69	35.33	14.78	50.24	26.95	19.01	7.48

Table A4: Ablation study for community detection. The best performance is shown in **bold**. The second-best performance is underlined. PE denotes Parameter Expansion. TG denotes Temporal Guidance mechanism. BK denotes BKFM.

Models	High School		Cell Phone		Cora		Citeseer		Photo		DBLP	
	NMI	ARI	NMI	ARI	NMI	ARI	NMI	ARI	NMI	ARI	NMI	ARI
DBKF-PE w/o PE	75.03	67.93	52.98	43.50	18.10	3.91	20.17	5.08	32.05	11.04	14.14	3.14
DBKF-PE w/o TG	80.93	77.56	55.03	45.84	33.80	13.15	33.73	13.24	42.57	21.12	17.86	7.17
DBKF-PE w/o BK	83.15	80.08	56.67	46.17	34.16	12.79	35.28	13.19	47.11	26.28	18.64	6.27
DBKF-PE	**84.50**	**82.51**	**59.89**	**47.25**	**36.15**	**19.69**	**35.33**	**14.78**	**50.24**	**26.95**	**19.01**	**7.48**

As listed in Tables A2 and A3, after model distillation, the performance of DBKF-PE does not degrade excessively and remains relatively high, indicating that DBKF-PE is still well-suited for dynamic graphs with a large number of snapshots.

A.2.4 The Experiment Results of Backward Knowledge Feedback for Community Detection

The community detection results of the model on the Cell Phone and Photo are shown in Fig. A3. From Fig. A3, the experimental results in Fig. A3 are similar to those in Figs. 4 and 5 in the main text, and the overall performance of DBKF-PE is superior to that of the other baselines. The primary reason is that DBKF-PE leverages BKFM to achieve backward feedback of new graph knowledge, thereby enabling the new model to effectively learn historical graph knowledge. Furthermore, we can observe that DBKF-PE not only alleviates catastrophic forgetting but also enhances the model's ability to learn both new and historical graph knowledge.

A.2.5 Ablation Study for Community Detection

The ablation study for community detection is listed in Table A4. As listed in Table A4, the ablation results for community detection are similar to those for node classification. Parameter expansion makes the largest contribution to DBKF-PE, followed by the temporal guidance mechanism and then BKFM. This indicates that these components make different contributions within DBKF-PE.