

Retrieval Augmented Generation for Dynamic Graph Modeling

Yuxia Wu
Singapore Management University
Singapore
yieshah2017@gmail.com

Lizi Liao
Singapore Management University
Singapore
lzliao@smu.edu.sg

Yuan Fang*
Singapore Management University
Singapore
yfang@smu.edu.sg

Abstract

Modeling dynamic graphs, such as those found in social networks, recommendation systems, and e-commerce platforms, is crucial for capturing evolving relationships and delivering relevant insights over time. Traditional approaches primarily rely on graph neural networks with temporal components or sequence generation models, which often focus narrowly on the historical context of target nodes. This limitation restricts the ability to adapt to new and emerging patterns in dynamic graphs. To address this challenge, we propose a novel framework, **Retrieval-Augmented Generation for Dynamic Graph modeling (RAG4DyG)**, which enhances dynamic graph predictions by incorporating contextually and temporally relevant examples from broader graph structures. Our approach includes a time- and context-aware contrastive learning module to identify high-quality demonstrations and a graph fusion strategy to effectively integrate these examples with historical contexts. The proposed framework is designed to be effective in both transductive and inductive scenarios, ensuring adaptability to previously unseen nodes and evolving graph structures. Extensive experiments across multiple real-world datasets demonstrate the effectiveness of RAG4DyG in improving predictive accuracy and adaptability for dynamic graph modeling. The code and datasets are publicly available at <https://github.com/YuxiaWu/RAG4DyG>.

CCS Concepts

• Information systems → Information retrieval; • Computing methodologies → Artificial intelligence.

Keywords

Retrieval-augmented generation, graph neural networks, dynamic graph modeling

ACM Reference Format:

Yuxia Wu, Lizi Liao, and Yuan Fang. 2025. Retrieval Augmented Generation for Dynamic Graph Modeling. In *Proceedings of the 48th International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR'25)*, July 13–18, 2025, Padua, Italy. ACM, New York, NY, USA, 10 pages. <https://doi.org/10.1145/3726302.3729958>

1 Introduction

Dynamic graph modeling is essential for understanding and predicting evolving interactions across various applications such as

social networks [5, 34], personalized recommendations [33, 35, 41], and Web-based services [7, 11]. These applications require effective handling of the temporal and contextual dynamics inherent in evolving graph structures to provide accurate and timely insights. For example, in a recommendation system [35], capturing user behavioral shifts over time can enhance the personalization of content, while in social networks [34], understanding interaction patterns can improve engagement and fraud detection mechanisms.

Existing dynamic graph modeling methods generally fall into discrete-time and continuous-time approaches [9]. Discrete-time models capture graph snapshots at specific intervals, providing a simplified representation of the graph's evolution but often failing to account for fine-grained temporal dynamics [29, 32]. Continuous-time models, such as DyRep [36], TGAT [43], and TREND [39], model events as they occur, offering a more granular perspective and better capturing event-driven dynamics. These models commonly rely on graph neural networks (GNNs) integrated with temporal mechanisms such as recurrent neural networks [29], self-attention [32], and temporal point processes [39] to update graph representations over time. Despite their strengths, GNN-based methods struggle with long-term dependencies and issues such as over-smoothing and over-squashing [1, 2]. Recently, SimpleDyG [40] redefined dynamic graph modeling as a generative sequence modeling task, leveraging Transformers to effectively capture long-range dependencies within temporal sequences. However, the reliance on localized historical interactions still limits the ability to generalize across different contexts and adapt to emerging patterns.

To address these limitations, the Retrieval-Augmented Generation (RAG) framework from the Natural Language Processing (NLP) domain [10] offers a promising approach. RAG has the potential to broaden the contextual understanding of dynamic graphs by retrieving and incorporating relevant examples from across the graph's temporal and contextual space, as illustrated in Figure 1. However, adopting RAG for dynamic graph modeling presents two major challenges: (1) *Selecting high-quality demonstrations*, and (2) *effectively integrating the retrieved demonstrations*. Identifying contextually and temporally relevant demonstrations is challenging because existing retrieval methods, such as BM25 and other matching-based schemes [30] primarily rely on historical interaction similarities and struggle with inductive scenarios where nodes lack historical interactions. Moreover, effective integration of retrieved demonstrations is another challenge, as simply concatenating them with the query sequence can lead to overly lengthy inputs and overlook underlying structural patterns.

To overcome these challenges, we propose a novel **Retrieval-Augmented Generation for Dynamic Graph modeling (RAG4DyG)** framework. It integrates a time- and context-aware contrastive learning strategy that evaluates historical interaction sequences

*Corresponding author



This work is licensed under a Creative Commons Attribution 4.0 International License. *SIGIR'25, Padua, Italy*

© 2025 Copyright held by the owner/author(s).

ACM ISBN 979-8-4007-1592-1/2025/07

<https://doi.org/10.1145/3726302.3729958>

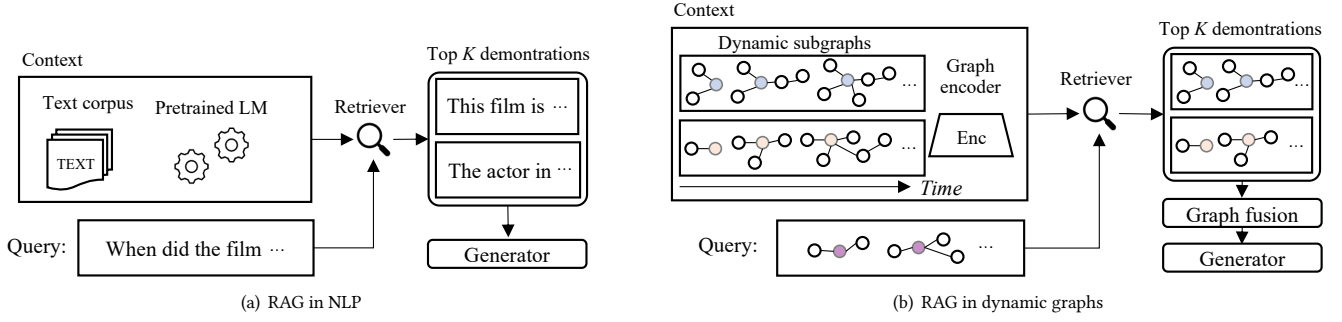


Figure 1: Illustration of RAG in NLP and dynamic graph modeling. (a) In NLP, RAG leverages pre-trained language models to encode text and retrieve semantically similar or related demonstrations, which are further concatenated to enhance the generation task. (b) Our work addresses the challenges of complex temporal and structural characteristics of dynamic graphs, incorporating RAG through time- and context-aware retrieval and graph fusion modules.

to identify relevant examples and a graph fusion module to construct a summary graph from the retrieved samples. The contrastive learning strategy incorporates a time decay function to prioritize temporally relevant samples, while context-aware augmentation techniques such as masking and cropping enhance the model’s ability to capture complex structural patterns. The graph fusion module applies a GNN-based readout mechanism to enrich the representation before feeding it into the sequence generation model. These solutions empower RAG4DyG to effectively leverage retrieved demonstrations to enhance dynamic graph modeling. Through extensive experimentation on various real-world datasets, we demonstrate that RAG4DyG outperforms state-of-the-art methods in both transductive and inductive scenarios, offering improved accuracy and adaptability in dynamic graph scenarios. In transductive settings, where test nodes have appeared during training, our model effectively leverages historical data to refine predictions. In inductive settings, involving previously unseen nodes, the retrieval mechanism enables the model to generalize by providing relevant contextual examples as guidance.

To sum up, our main contributions are as follows. (1) We propose a novel retrieval-augmented generation approach for dynamic graph modeling named RAG4DyG, which employs a retriever to broaden historical interactions with contextually and temporally relevant demonstrations. (2) We introduce a time- and context-aware contrastive learning module that incorporates temporal and structural information for demonstration retrieval and a graph fusion module to effectively integrate retrieved demonstrations. (3) We conduct extensive experiments to validate our approach, demonstrating the effectiveness of RAG4DyG across various domains.

2 Related Work

Dynamic Graph Modeling. Existing approaches for dynamic graphs can be categorized into discrete-time and continuous-time methods. Discrete-time methods regard a dynamic graph as a sequence of static graph snapshots captured at various time steps. Each snapshot represents the graph structure at a specific time step. These methods typically adopt GNNs to model the structural information of each snapshot, and then incorporate a sequence model [29, 32] to capture the changes across snapshots. However, these

approaches neglect fine-grained time information within a snapshot. In contrast, continuous-time methods model graph evolution as a continuous process, capturing all time steps for more precise temporal modeling. These methods often integrate GNNs with specially designed temporal modules, such as temporal random walk [38], temporal graph attention [31, 43], MLP-mixer [3] and temporal point processes [16, 36, 39]. Recently, researchers have proposed a simple and effective architecture called SimpleDyG [40], which reformulates dynamic graph modeling as a sequence modeling task. Specifically, it maps the dynamic graph into a series of node sequences and feeds them into a generative sequence model. Subsequently, predicting future events can be framed as a sequence generation problem. However, while these methods provide valuable insights, they are often limited in their ability to adapt to new and evolving patterns in dynamic graphs.

Our work distinguishes itself from prior dynamic graph learning methods through two key innovations. First, while existing approaches predominantly focus on localized temporal contexts or the historical interactions of target nodes, our proposed RAG4DyG employs retrieval-augmented generation mechanisms to retrieve and integrate broader contextual signals from the entire dynamic graph. This approach facilitates a more comprehensive understanding of dynamic interactions, uncovering complex patterns beyond the immediate historical scope of individual nodes. Second, RAG4DyG incorporates a time- and context-aware contrastive learning module for retrieving similar demonstrations, along with a graph fusion strategy to integrate them with the query sequence, enhancing adaptability to new patterns and evolving graph structures.

Retrieval Augmented Generation. Recently, the RAG paradigm has attracted increasing attention [10]. Specifically, RAG first leverages the retriever to search and extract relevant documents from some databases, which then serve as additional context to enhance the generation process. Related studies have demonstrated the great potential of RAG in various tasks such as language processing [18, 19], recommendation systems [4, 44], and computer vision [20, 25]. In the graph modeling field, existing RAG efforts have primarily focused on static [27] and text-attributed graphs to enhance the generation capabilities of Large Language Models (LLMs), supporting graph-related tasks such as code summarization [26] and

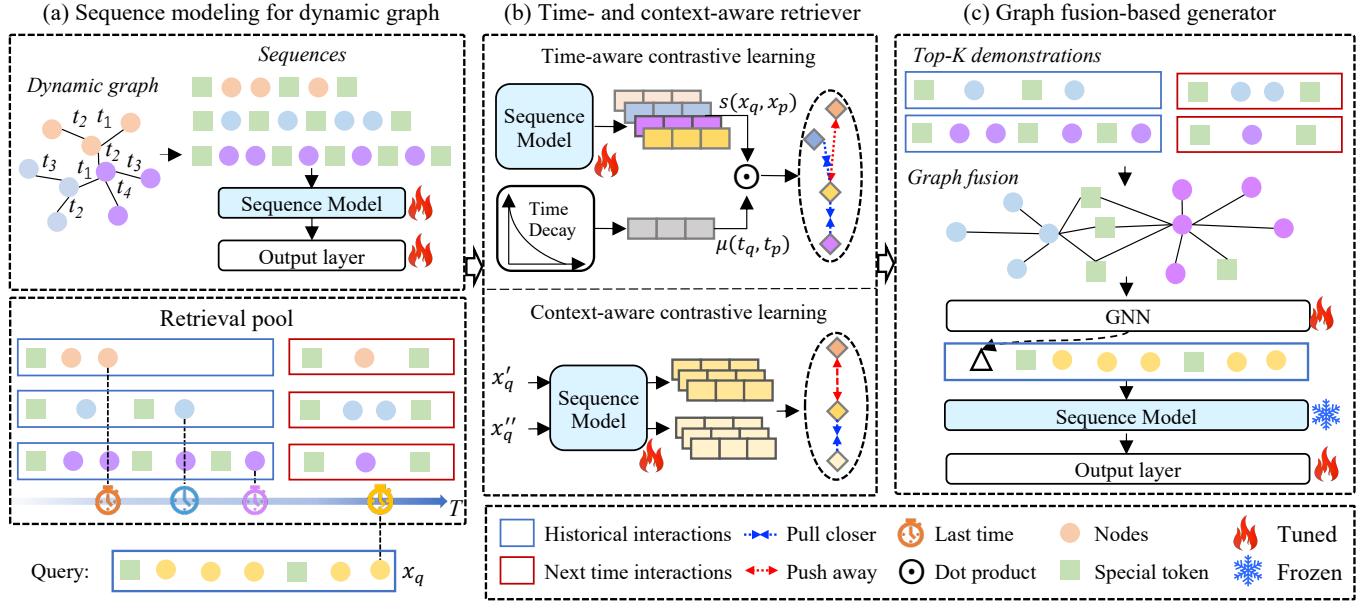


Figure 2: Overall framework of RAG4DyG. (a) Sequence modeling for dynamic graphs. (b) The retriever finds top- K temporally and contextually relevant demonstrations. (c) Graph fusion integrates the retrieved demonstrations for the subsequent generation.

textual graph question answering [12, 13, 17]. However, exploiting RAG techniques for dynamic graphs and graphs without textual information remains largely unexplored.

3 Preliminaries

In this section, we introduce the sequence modeling of dynamic graphs and the problem formulation.

Sequence Mapping of Dynamic Graphs. We denote a dynamic graph as $G = (V, E, F, \mathcal{T})$ comprising a set of nodes V , edges E , a node feature matrix F if available, and a time domain \mathcal{T} . To map a dynamic graph into sequences, we follow SimpleDyG [40]. Specifically, let $D = \{(x_i, y_i)\}_{i=1}^M$ denote the set of training samples, where each sample is a pair (x_i, y_i) , representing the input and output sequences for a target node $v_i \in V$. The input x_i is a chronologically ordered sequence of nodes that have historically interacted with v_i , while the output y_i is the ground truth interactions that occur following the sequence x_i . In notations, we have

$$x_i = [\text{hist}], v_i, [\text{time_1}], v_i^{1,1}, v_i^{1,2}, \dots, [\text{time_t}], v_i^{t,1}, \dots, [\text{time_T}], v_i^{T,1}, \dots, [\text{eohist}], \quad (1)$$

$$y_i = [\text{pred}], [\text{time_T+1}], v_i^{T+1,1}, \dots, [\text{eopred}], \quad (2)$$

where $[\text{hist}]$, $[\text{eohist}]$, $[\text{pred}]$, $[\text{eopred}]$ are special tokens denoting the input and output sequence, and $[\text{time_1}], \dots, [\text{time_T+1}]$ are special time tokens representing different time steps.

Problem Formulation. Dynamic graph modeling aims to learn a model that can predict the future interactions of a target node v_i , given its historical interactions. That is, given x_i in Eq. (1), the task is to predict y_i in Eq. (2). In our RAG framework, we regard the training samples D as a retrieval pool. Given a target node $v_q \in V$, its input sequence x_q is referred to as the *query sequence*.

We first retrieve K demonstrations $R_q = \{(x_k, y_k)\}_{k=1}^K$ for each query sequence x_q based on their contextual and temporal relevance. Next, the retrieved demonstrations R_q are used to enrich the input sequence x_q , which encompasses the historical interactions of the target node v_q . The augmented input $\{R_q, x_q\}$ is designed to enhance the predictions of future events in y_q .

4 Proposed Model: RAG4DyG

The RAG4DyG framework enhances dynamic graph modeling by incorporating retrieval-augmented generation techniques to improve predictive accuracy and adaptability. As illustrated in Fig. 2, it first adopts SimpleDyG [40] to model dynamic graphs as sequences of node interactions, leveraging a Transformer-based model to capture temporal dependencies and predict future interactions (Fig. 2(a)). To enrich the modeling process, a time- and context-aware retriever retrieves relevant demonstrations from a retrieval pool D for a given query sequence x_q . This retriever optimizes two contrastive objectives: a time-aware loss, which employs a temporal decay function $\mu(t_q, t_p)$ to prioritize temporally relevant samples, and a context-aware loss, which utilizes sequence augmentation techniques such as masking and cropping to capture structural patterns (Fig. 2(b), Sec. 4.1). Once the top- K demonstrations are retrieved, they are fused into a summary graph G_{fus} , which captures the underlying structural relationships among the retrieved samples. A GNN then processes this graph to generate an enriched representation that is prepended to the query sequence, providing additional context for improved event prediction (Fig. 2(c), Sec. 4.2). By integrating retrieval and graph fusion, RAG4DyG effectively incorporates temporal and contextual relevance, surpassing existing methods in both transductive and inductive dynamic graph scenarios.

4.1 Time- and Context-Aware Retriever

Unlike NLP retrievers, dynamic graph retrieval requires consideration of temporal proximity alongside contextual relevance. To address this, we propose a time- and context-aware retrieval model with two contrastive learning modules. First, we incorporate a time decay mechanism to account for temporal proximity between query and candidate sequences. Second, we use sequence augmentation to capture intrinsic contextual patterns.

Retrieval Annotation. To facilitate contrastive training, we automatically annotate the samples in the retrieval pool D . For each query sequence x_q , we annotate its positive sample x_p^+ from the pool D based on their contextual similarity. We leave the detailed annotation process in Sec. 5.1. Specifically, we adopt the sequence model pre-trained in Fig. 2(a) as the encoder and apply mean pooling to obtain sequence representations. Given a query sequence x_q and a candidate sequence $x_p \in D$, we define their contextual similarity as the dot product of their representations:

$$s(x_q, x_p) = f(x_q)^\top f(x_p), \quad (3)$$

where $f(\cdot)$ denotes our encoder.

Time-aware Contrastive Learning. Temporal information reflects the dynamic changes in historical interactions, which is crucial for dynamic graph modeling. We posit that demonstrations closer in time to the query are more relevant than those further away. Consequently, we utilize a time decay function to account for temporal proximity between the query and candidate sequences, as follows.

$$\mu(t_q, t_p) = \exp(-\lambda|t_q - t_p|), \quad (4)$$

where t_q and t_p represent the last interaction time in the query and candidate sequences¹, respectively. The hyper-parameter λ controls the rate of time decay, determining how quickly the importance of interactions decreases with time. Note that $0 < \mu(\cdot, \cdot) \leq 1$. By using this time decay function, we assign higher importance to the candidates that are temporally closer to the query.

To effectively capture the temporal dynamics of the graph, we incorporate temporal proximity to reweigh the contextual similarity in the contrastive loss:

$$h(x_q, x_p) = s(x_q, x_p)\mu(t_q, t_p). \quad (5)$$

Subsequently, we adopt in-batch negative sampling based on the following training objective:

$$\mathcal{L}_{\text{tcl}} = -\log \frac{\exp(h(x_q, x_p^+))/\tau}{\sum_{j=1}^{2N} \mathbb{1}_{j \neq q} \exp(h(x_q, x_j))/\tau}, \quad (6)$$

where x_p^+ denotes the positive sample of x_q , N is the batch size, and τ is the temperature parameter.

Context-aware Contrastive Learning. To better capture the inherent contextual pattern, we further adopt context-aware contrastive learning with data augmentations. For each sequence, we apply two types of augmentations: masking and cropping, which are widely used for sequence modeling [6, 42]. The masking operator randomly replaces a portion of the tokens in the sequence with a special masking token. The cropping operator randomly deletes

¹In the annotated training data, the query time t_q may precede the candidate time t_p . However, in the validation and test sets, t_p always precedes t_q , preventing leakage from a future time.

a contiguous subsequence from the original sequence, reducing the sequence length while preserving the temporal order of the interactions. These augmentations help the model learn robust representations and capture the inherent structural information of the sequence by focusing on its different parts.

We treat two augmented views of the same sequence as positive pairs, and those of different sequences as negative pairs. Given a sequence x_q and its two distinct augmented views x'_q and x''_q , the contrastive loss is defined as:

$$\mathcal{L}_{\text{ccl}} = -\log \frac{\exp(s(x'_q, x''_q)/\tau)}{\sum_{j=1}^{2N} \mathbb{1}_{j \neq q} \exp(s(x'_q, x'_j)/\tau)}, \quad (7)$$

where τ is the temperature, N is the batch size and $\mathbb{1}$ is an indicator function.

Training and Inference for Retrieval. The training objective of our retrieval model is defined as:

$$\mathcal{L}_{\text{ret}} = \mathcal{L}_{\text{tcl}} + \alpha \mathcal{L}_{\text{ccl}}, \quad (8)$$

where α is a coefficient that balances between the two losses. During testing, we utilize the updated sequence model to extract sequence representations and perform demonstration ranking based on the contextual similarity between the query and candidates.

4.2 Graph Fusion-based Generator

After the retrieval process, we obtain the top- K demonstrations $R_q = \{(x_k, y_k)\}_{k=1}^K$ for the query x_q . A straightforward approach is to directly concatenate them with the query sequence and input to a sequence generation model for prediction. However, this can lead to a lengthy context that limits the model's prediction capabilities. More importantly, it neglects the structural patterns among these demonstrations. Thus, we first fuse the demonstrations into a summary graph, process it using a GNN, and then prepend the graph readout from the GNN to the query for subsequent generation.

Graph Fusion. To effectively fuse the demonstrations in R_q , we construct a summary graph, whose nodes include all tokens in the retrieved demonstrations, and edges represent the interactions between nodes within each sequence. Considering that there are common tokens across the retrieved demonstrations (e.g., recurring nodes in multiple demonstrations and special tokens like *[hist]*, *[time1]*, etc.), we can fuse these demonstrations into a summary graph G_{fus} . We then employ a graph convolutional network (GCN) to capture the structural and contextual information within the fused graph and apply a mean-pooling readout to obtain a representation vector for the graph. The vector is subsequently concatenated with the query sequence representation, as follows.

$$e_{\text{fus}} = \text{MeanPooling}(\text{GCN}(G_{\text{fus}})), \quad (9)$$

$$\tilde{x}_q = [e_{\text{fus}} \parallel x_q], \quad (10)$$

where e_{fus} is the fused graph representation, and \tilde{x}_q is the retrieval-augmented sequence. The augmented sequence is fed into the sequence model, which generates future interactions.

Training and Inference. We adopt the same sequence model with the same training objective [40] as in Fig. 2(a). During training, we freeze the parameters of the sequence model, except for the output layer which is updated along with the GCN parameters used for graph fusion. During testing, we first apply the retriever model

Table 1: Dataset statistics.

Dataset	UCI	Hepth	MMConv	Wikipedia	Enron	Reddit
Domain	Social	Citation	Conversation	Behavior	Social	Hyperlink
# Nodes	1,781	4,737	7,415	9,227	42,711	11,901
# Edges	16,743	14,831	91,986	157,474	797,907	62,919

to retrieve top- K demonstrations for each query as introduced in Sec. 4.1. Then we perform graph fusion on these demonstrations and concatenate the fused graph representation with the query sequence as illustrated in Eq. (9) and (10). The concatenated sequence is subsequently fed into the trained model for link prediction.

5 Experiment

In this section, we empirically evaluate the proposed model RAG4DyG compared to state-of-the-art methods and conduct a detailed analysis of the performance.

5.1 Experimental Setup

5.1.1 Datasets. We evaluate the performance of the proposed model on six datasets from different domains, including a communication network UCI [28], a citation network Hepth [23], a multi-turn task-oriented conversation dataset MMConv [24], a behavioral interaction network Wikipedia [14], an email network Enron [45], and a hyperlink network Reddit [21]. We summarize the statistics of these datasets in Table 1. We follow the preprocessing steps of SimpleDyG to map the dynamic graphs into sequences with special tokens [40]. Notably, the Hepth and Reddit datasets exhibit an inductive nature, as they contain previously unseen nodes with no historical interactions. The details of the Wikipedia, Enron and Reddit datasets are provided below, while information on the remaining datasets can be found in SimpleDyG [40].

- **Wikipedia** [14]: This dataset captures the co-editing activity on Wikipedia pages over one month. It is a bipartite interaction network in which editors and wiki pages serve as nodes. Each edge corresponds to an interaction where a user edits a page at a specific timestamp. To facilitate temporal sequence alignment, the dataset is divided into 16 time steps based on the timestamps of the interactions.
- **Enron** [45]: This dataset represents the email communications among employees of Enron Corporation over three years (1999–2002). Nodes represent employees, while edges correspond to emails exchanged between them ordered by the sending timestamps of the emails. For temporal sequence alignment, we split the dataset into 17 time steps based on the timestamps.
- **Reddit** [21]: This dataset represents a subreddit-to-subreddit hyperlink network, derived from timestamped posts containing hyperlinks between subreddits. We focus on hyperlink data within the body of posts, covering the period from 2016 to 2017. The dataset is divided into 12 time steps for temporal sequence alignment based on the post timestamps.

5.1.2 Baselines. We compare our model RAG4DyG with the state-of-the-art dynamic graph models, which include (1) discrete-time approaches: DySAT [32] and EvolveGCN [29]; (2) continuous-time

approaches: DyRep [36], JODIE [22], TGAT [43], TGN [31], TREND [39], GraphMixer [3], IDOL [46] and SimpleDyG [40].

- **DySAT** [32] utilizes self-attention mechanisms to capture both structural and temporal patterns in dynamic graphs through discrete-time snapshots.
- **EvolveGCN** [29] leverages recurrent neural networks to model the evolution of the parameters of a graph convolutional network over discrete time steps.
- **DyRep** [36] models dynamic graphs in continuous time by incorporating both temporal point processes and structural dynamics to capture interactions and node dynamics.
- **JODIE** [22] focuses on user and item embedding trajectories over continuous time, predicting future interactions by modeling user and item embeddings jointly.
- **TGAT** [43] employs temporal graph attention layers and time encoding to capture temporal dependencies and structural information for dynamic graphs.
- **TGN** [31] combines GNNs with memory modules to maintain node states over continuous time, effectively learning from dynamic interactions.
- **TREND** [39] integrates temporal dependencies based on the Hawkes process and GNNs to learn the dynamics of graphs.
- **GraphMixer** [3] introduces a novel architecture that leverages MLP-mixer to learn link-encoder and node encoder for evolving graphs in continuous time.
- **IDOL** [46] is a contrastive learning-based model tailored for dynamic graph representation learning. It utilizes a Personalized PageRank-based algorithm to incrementally update the node embedding and adopt a topology-monitorable sampling method to generate contrastive pairs for efficient training.
- **SimpleDyG** [40] reformulates the dynamic graph modeling as a sequence modeling task and mapped the dynamic interactions of target nodes as sequences with specially designed tokens. It simplifies dynamic graph modeling without complex architectural changes to effectively capture temporal dynamics.

5.1.3 Implementation Details. Following the method outlined in [3, 40], we represent the dynamic graph as an undirected graph. We split all datasets into training, validation, and test sets based on temporal sequence same as SimpleDyG [40]. Given T timesteps in each dataset, the data at the final timestep T is used as the testing set, the data at $T - 1$ is served as the validation set, and the remaining data from earlier timesteps is used for training. All training data including the retrieval pool for the retriever and generator is drawn from this training data split. For retrieval augmented generation model training, we first train SimpleDyG without augmentation using the finetuned parameters. Then we fix the parameters of SimpleDyG except for the last linear layer and fine-tune them with the GCN model. The number of GCN layers in the generator model is 1 for all datasets. We repeat each experiment 10 times and report the average results along with the standard deviation. The number of demonstrations is 7 for all datasets.

To facilitate retrieval model training, we regard the samples in the training dataset as our retrieval pool $D = \{(x_i, y_i)\}_{i=1}^M$ where each pair (x_i, y_i) represents the historical sequence and its corresponding target sequence. Specifically, x_i is the input sequence before the last time step and y_i is the output sequence at the last

Table 2: Performance comparison for dynamic link prediction with mean and standard deviation across 10 runs. Best results are bolded; runners-up are underlined. * indicates that our model significantly outperforms the best baseline based on the two-tail t -test ($p < 0.05$).

Datasets	Models	DySAT	EvolveGCN	DyRep	JODIE	TGAT	TGN	TREND	GraphMixer	IDOL	SimpleDyG	RAG4DyG
UCI	Recall@5	0.009±0.003	0.072±0.046	0.009±0.008	0.018±0.019	0.022±0.004	0.014±0.010	0.083±0.015	0.097±0.019	0.093±0.029	<u>0.109±0.014</u>	0.111±0.013
	NDCG@5	0.010±0.003	0.064±0.045	0.011±0.018	0.022±0.023	0.061±0.007	0.041±0.017	0.067±0.010	<u>0.104±0.013</u>	0.075±0.022	<u>0.104±0.010</u>	0.122*±0.014
	Jaccard	0.010±0.001	0.032±0.026	0.010±0.005	0.012±0.009	0.020±0.002	0.011±0.003	0.039±0.020	0.042±0.005	0.014±0.002	<u>0.092±0.014</u>	0.097±0.010
Hepth	Recall@5	0.008±0.004	0.008±0.002	0.009±0.006	0.010±0.008	0.011±0.007	0.011±0.006	0.010±0.008	0.009±0.002	0.007±0.002	<u>0.013±0.006</u>	0.019*±0.002
	NDCG@5	0.007±0.002	0.009±0.004	0.031±0.024	0.031±0.021	0.034±0.023	0.030±0.012	0.031±0.003	0.011±0.008	0.011±0.003	<u>0.035±0.014</u>	0.045*±0.003
	Jaccard	0.005±0.001	0.007±0.002	0.010±0.006	0.011±0.008	0.011±0.006	0.008±0.001	0.010±0.002	0.010±0.003	0.006±0.001	<u>0.013±0.006</u>	0.019*±0.002
MMConv	Recall@5	0.108±0.089	0.050±0.015	0.156±0.054	0.052±0.039	0.118±0.004	0.085±0.050	0.134±0.030	0.206±0.001	0.169±0.006	0.170±0.010	<u>0.194±0.005</u>
	NDCG@5	0.102±0.085	0.051±0.021	0.140±0.057	0.041±0.016	0.089±0.033	0.096±0.068	0.116±0.020	0.172±0.029	0.115±0.039	<u>0.184±0.012</u>	0.208*±0.005
	Jaccard	0.095±0.080	0.032±0.017	0.067±0.025	0.032±0.022	0.058±0.021	0.066±0.038	0.060±0.018	0.085±0.016	0.015±0.002	<u>0.169±0.010</u>	0.194*±0.005
Wikipedia	Recall@5	0.003±0.005	0.012±0.01	0.003±0.002	0.017±0.005	0.006±0.004	0.016±0.018	0.022±0.012	0.010±0.007	0.022±0.008	<u>0.356±0.016</u>	0.369*±0.006
	NDCG@5	0.002±0.003	0.008±0.007	0.002±0.002	0.015±0.003	0.005±0.005	0.015±0.022	0.016±0.018	0.007±0.006	0.015±0.005	0.398±0.03	<u>0.389±0.008</u>
	Jaccard	0.001±0.001	0.004±0.004	0.001±0.001	0.007±0.002	0.002±0.002	0.007±0.009	0.007±0.021	0.004±0.002	0.004±0.001	<u>0.320±0.027</u>	0.328±0.007
Enron	Recall@5	0.002±0.004	0.004±0.011	0.021±0.001	0.005±0.005	0.020±0.002	0.001±0.001	0.023±0.003	0.021±0.002	0.024±0.014	<u>0.094±0.005</u>	0.100*±0.003
	NDCG@5	0.001±0.002	0.007±0.020	0.036±0.002	0.061±0.039	0.036±0.001	0.003±0.001	0.027±0.001	0.037±0.001	0.025±0.011	<u>0.114±0.005</u>	0.119*±0.004
	Jaccard	0.001±0.001	0.003±0.009	0.019±0.001	0.011±0.007	0.020±0.001	0.001±0.001	0.012±0.001	0.020±0.002	0.008±0.003	<u>0.068±0.003</u>	0.071*±0.002
Reddit	Recall@5	0.001±0.002	0.006±0.002	0.019±0.004	0.013±0.003	0.001±0.001	0.001±0.001	0.002±0.003	0.001±0.001	0.003±0.002	<u>0.101±0.019</u>	0.119*±0.006
	NDCG@5	0.001±0.002	0.012±0.003	0.020±0.004	0.015±0.002	0.001±0.001	0.002±0.001	0.003±0.002	0.003±0.001	0.005±0.003	<u>0.134±0.012</u>	0.143±0.005
	Jaccard	0.001±0.001	0.003±0.001	0.013±0.004	0.007±0.002	0.001±0.001	0.001±0.002	0.001±0.001	0.002±0.001	0.002±0.001	<u>0.088±0.012</u>	0.096±0.003

time step. We annotate demonstrations based on the Jaccard similarity between the output sequences among all the pairs in D . The Jaccard similarity of two output sequences, y_i and y_j , is given by

$$r(y_i, y_j) = \frac{|y_i \cap y_j|}{|y_i \cup y_j|}. \quad (11)$$

To control the quality of annotated data, we set a threshold of 0.8 to select highly similar demonstrations for each sample. These filtered annotations are then used to train the retriever model. The number of training samples for the UCI, Hepth, MMConv, Wikipedia, Enron and Reddit datasets are 9 578, 8 250, 10 762, 162 408, 2 510 666 and 185 764, respectively.

We ran all the experiments on a Nvidia L40 GPU and tuned the hyper-parameters for all the methods based on the validation set. For all the baselines, we tuned the models based on the hyper-parameters reported in their papers. For our RAG4DyG method, we set the number of layers, heads and dimensions of hidden states for the backbone SimpleDyG to (6, 8, 768), (12, 2, 256), (2, 2, 256), (2, 6, 768), (2, 6, 768), and (2, 8, 512) across the six datasets. The time decay rate λ for the retrieval model in Eq. (4) was tuned according to the time granularity of different datasets, with days for the UCI dataset, months for the Hepth dataset, turns for the MMConv dataset, and hours for other datasets. We explored a range of values $\lambda = \{10^{-4}, 10^{-3}, 10^{-2}, 10^{-1}, 1, 10, 100\}$, ultimately selecting $\lambda = 10^{-4}$ for UCI, $\lambda = 0.1$ for Hepth, $\lambda = 10$ for MMConv and Enron, and $\lambda = 1$ for Wikipedia and Reddit datasets. The coefficient α in the loss function in Eq. (8) was tuned across $\{0.2, 0.4, 0.6, 0.8, 1\}$, resulting in final values of $\alpha = 1$ for UCI and MMConv, $\alpha = 0.4$ for Hepth, and $\alpha = 0.2$ for Wikipedia, Enron and Reddit. Additional

parameter settings for the three datasets are as follows: the temperature τ in the two contrastive learning tasks for all datasets was set to $\tau = 0.1$, the batch size of the retriever model for all datasets was set to $N = 128$, and the masking and cropping portions in context-aware contrastive learning were set to 0.8, 0.8, 0.8, 0.6, 0.6, 0.2 and 0.4, 0.6, 0.6, 0.8, 0.8, 0.8 across the six datasets, respectively.

5.1.4 Evaluation Metrics. Inspired by SimpleDyG [40], we assess the performance of our approach and baselines using three key metrics: Recall@5, NDCG@5, and Jaccard [40]. Recall@5 and NDCG@5 are commonly employed in ranking tasks to evaluate the quality of top-ranked predictions [37]. Specifically, Recall@5 measures the proportion of relevant nodes that appear among the top five predictions, while NDCG@5 considers the ranking positions of the relevant nodes to provide a more nuanced assessment of ranking quality. Additionally, the Jaccard index [15] quantifies the similarity between the predicted and ground truth sequences by calculating the ratio of their intersection to their union.

5.2 Performance Comparison

We assess the performance of RAG4DyG on the dynamic link prediction task, with the results benchmarked against state-of-the-art baselines, as shown in Table 2. Our analysis reveals the following key observations.

First of all, the proposed RAG4DyG generally outperforms all baselines across different datasets under the three metrics. In particular, compared to SimpleDyG, which is also our backbone, RAG4DyG consistently shows superior performance, highlighting the effectiveness of our retrieval-augmented generation framework. Note that GraphMixer performs slightly better in Recall@5 on the MMConv

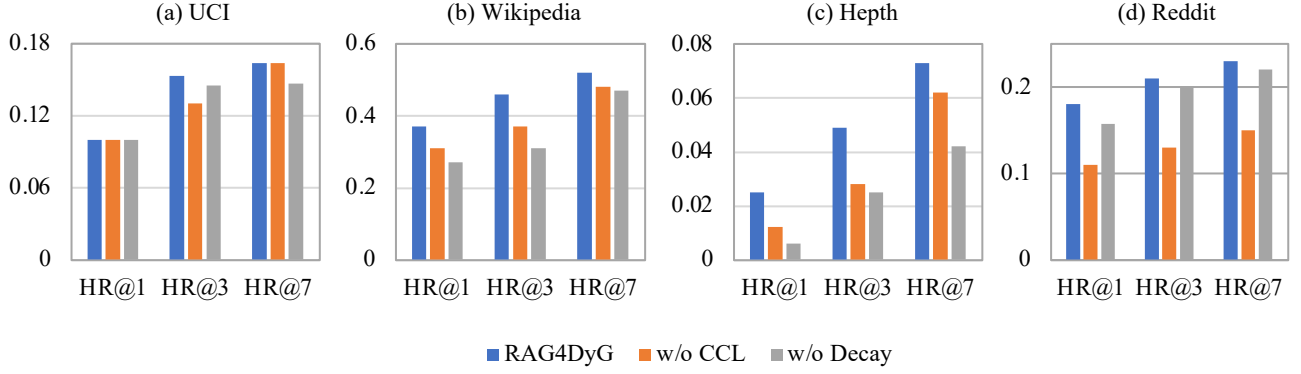


Figure 3: Ablation study for retrieval results.

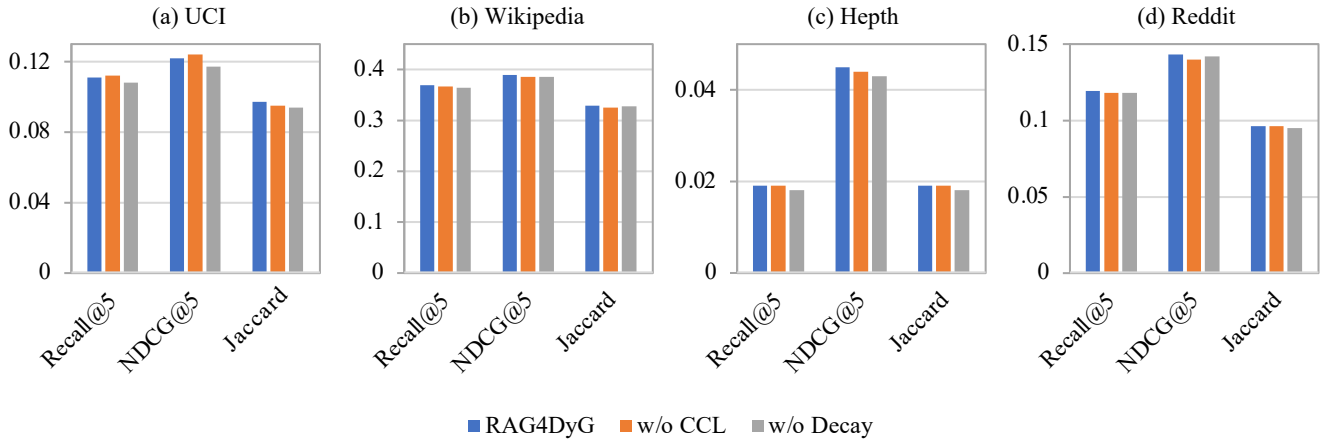


Figure 4: Ablation study for link prediction results.

dataset, but its significantly lower performance in NDCG@5 and Jaccard indicates that its predictions are not ranked optimally or maintaining the overall set integrity compared to RAG4DyG. This indicates that RAG4DyG can better model the temporal and contextual relationships due to the specific design of the retriever and generator. Generally speaking, the performance of SimpleDyG and RAG4DyG which reformulate the dynamic graph link prediction as a sequence generation task show promising performance compared with node pair ranking-based baselines, especially on the Wikipedia dataset which contains a higher frequency of repeated interaction behaviors. This characteristic makes sequence-based models more effective, as they can leverage the temporal consistency and recurrent patterns in the data to better capture the underlying dynamics of the graph.

Second, RAG4DyG exhibits significant advantages in inductive scenarios such as the Hepth and Reddit datasets. This setting is particularly challenging because it involves nodes not seen during training, requiring the model to generalize to entirely new structures and relationships. RAG4DyG’s success is attributed to its retrieval-augmented mechanism, which enhances the model’s ability to generalize by providing rich contextual information relevant to the new, unseen nodes. Unlike models that rely solely on the immediate neighborhood or predefined structures, RAG4DyG

dynamically adapts to the new nodes, ensuring that the predictions are guided by the most relevant and similar historical data.

5.3 Model Analysis

We analyze the behavior of our model RAG4DyG in several aspects, including an ablation study, an investigation of the effectiveness of different retrieval methods, and an analysis of parameter sensitivity.

5.3.1 Ablation Study. To evaluate the effectiveness of different modules in the retrieval model, we compare RAG4DyG with two variants *w/o CCL* and *w/o Decay* which exclude the context-aware contrastive learning and time decay component in the retrieval model. We evaluate the performance for both retrieval and link prediction tasks. We use $HR@k$ (Hit Ratio@ k) metrics for the retrieval model, measuring the proportion of cases where at least one of the top- k retrieved items is relevant. As shown in Fig. 3 and 4, the full model outperforms the two variants, underscoring the benefits of incorporating context-aware contrastive learning and time decay modulation. Notably, the *w/o Decay* variant exhibits the worst performance across both tasks, emphasizing the critical role of time decay in capturing temporal relevance and accurately modeling the evolving dynamics of the graph.

Table 3: Retrieval performance of various retrieval methods.

Method	UCI			Wikipedia			Hepth			Reddit		
	HR@1	HR@3	HR@7	HR@1	HR@3	HR@7	HR@1	HR@3	HR@7	HR@1	HR@3	HR@7
<i>BM25</i>	0.100	0.136	0.200	0.369	0.405	0.488	-	-	-	-	-	-
<i>Jaccard</i>	0.100	0.109	0.146	0.369	0.445	0.430	-	-	-	-	-	-
<i>RAG4DyG</i>	0.100	0.155	0.164	0.369	0.455	0.523	0.025	0.049	0.073	0.180	0.218	0.228

"-" denotes that the method is unable to perform retrieval. The reason is explained in the corresponding description of this table in Sec. 5.3.

Table 4: Generative performance of various retrieval methods.

Method	UCI			Wikipedia			Hepth			Reddit		
	Recall@5	NDCG@5	Jaccard	Recall@5	NDCG@5	Jaccard	Recall@5	NDCG@5	Jaccard	Recall@5	NDCG@5	Jaccard
<i>BM25</i>	0.111 \pm 0.007	0.121 \pm 0.009	0.093 \pm 0.004	0.368 \pm 0.01	0.389 \pm 0.012	0.325 \pm 0.01	-	-	-	-	-	-
<i>Jaccard</i>	0.104 \pm 0.009	0.113 \pm 0.011	0.088 \pm 0.010	0.368 \pm 0.013	0.388 \pm 0.014	0.321 \pm 0.011	-	-	-	-	-	-
<i>RAG4DyG</i>	0.111 \pm 0.013	0.122 \pm 0.014	0.097 \pm 0.010	0.369 \pm 0.006	0.389 \pm 0.008	0.328 \pm 0.007	0.019 \pm 0.002	0.045 \pm 0.003	0.019 \pm 0.002	0.119 \pm 0.006	0.143 \pm 0.005	0.096 \pm 0.003
<i>GroundTruth</i>	0.121 \pm 0.010	0.129 \pm 0.010	0.107 \pm 0.012	0.390 \pm 0.008	0.400 \pm 0.007	0.340 \pm 0.006	0.028 \pm 0.004	0.062 \pm 0.007	0.028 \pm 0.004	0.121 \pm 0.008	0.145 \pm 0.008	0.099 \pm 0.005

See the note in Table 3 for the explanation of "-".

5.3.2 Effect of Different Retrieval Methods. To further investigate the effectiveness of the retrieval model, we compare our model with two different retrieval methods, namely, BM25 and Jaccard, in Table 3 and 4. BM25 [8] is an extension of the Term Frequency-Inverse Document Frequency (TF-IDF) model, which calculates a relevance score between the query sequence and each candidate sequence in the retrieval pool. The relevance score is derived from the occurrence frequency of the nodes in the query and the retrieval pool. Jaccard [15] measures the similarity between two sets by comparing the size of their intersection to the size of their union. Note that in the citation dataset Hepth and hyperlink dataset Reddit, the queries in the test set contain unseen target nodes that never appear in the retrieval pool and have no historical interactions. As a result, the BM25 and Jaccard scores between the queries and the candidates in the retrieval pool are always zeros. On the other hand, our retrieval model is trained based on the sequence representations. For a query sequence containing only the target node, we can still obtain its representation using the sequence model trained for the retrieval model, and further calculate its contextual similarity with the candidate sequences in the retrieval pool.

In Table 3, we analyze the retrieval performance of different methods. For transductive scenarios, our retrieval model shows comparable performance to other retrieval strategies. Notably, in inductive scenarios like the Hepth and Reddit datasets, BM25 and Jaccard fail to work with new query nodes lacking historical interactions. In contrast, our model can handle them effectively and achieve solid performance.

Table 4 shows the generative performance of different retrieval methods in the dynamic link prediction task. During testing, we apply the retrieval results obtained from different retrieval methods. We also train a model using the ground-truth retrieval results for a more comprehensive comparison. The "GroundTruth" row represents an upper bound on the performance when using ground-truth

retrieval results on the testing data, which, as expected, provides the highest performance metrics. Generally speaking, all retrieval methods show better performance compared to the backbone SimpleDyG without using RAG, demonstrating the effectiveness of the RAG technique for dynamic graph modeling. Our method performs better compared to other retrieval strategies, indicating the effectiveness of contrastive learning in the retrieval model.

5.3.3 Effect of the Number of Demonstrations K . To investigate the influence of the number of demonstrations, we conduct experiments across varying values $K \in \{1, 3, 5, 7, 9\}$. As shown in Fig. 5, a higher number of K yields better prediction performance, that's because more demonstrations provide richer contextual information, especially in the UCI dataset. However, including too many cases may introduce more noise, which can harm the performance. Ultimately, we select $K = 7$ for all datasets.

5.3.4 Effect of Different Fusion Strategies. To further investigate the effectiveness of the fusion strategy for the top- K demonstrations, we conduct experiments with different fusion strategies under $K = 7$. "Concatenation" denotes we directly concatenate the sequences of retrieved demonstrations and prepend them with the query sample sequence and then feed it into the pre-trained SimpleDyG model. "MLP" means we do not consider the graph structure of the demonstrations and replace the graph fusion as an MLP layer (we set the number of the MLP layer as 2). By using the MLP layer, We map the concatenated demonstrations into shorter m -dimensional embeddings (we empirically set m to be 15) and then concatenate it with the query sample. Like graph fusion, we only fine-tune the parameters of the MLP and output layer. The results in Table 5 show that directly concatenating the retrieved demonstrations with the query sample leads to lower performance compared with other strategies. This is because simple concatenation introduces a lengthy context, which can overwhelm the model with irrelevant

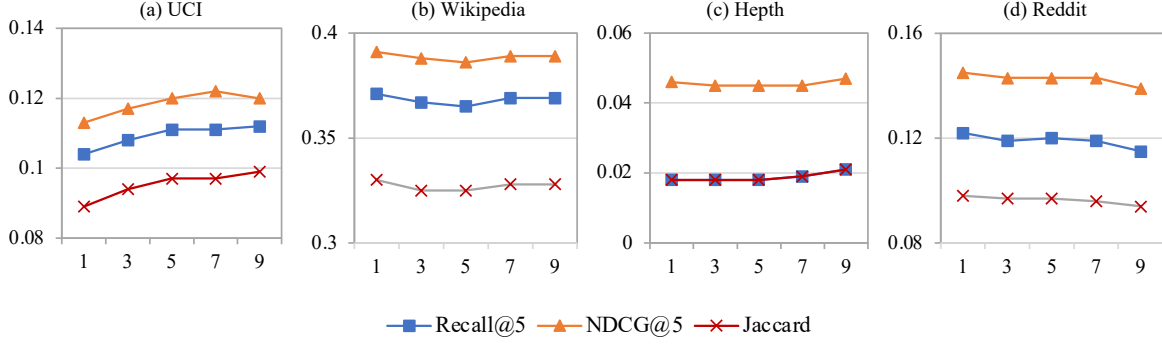
Figure 5: Effect of the number of demonstrations K .

Table 5: Effect of different fusion strategies.

Fusion strategy	UCI			Wikipedia			Hepth			Reddit		
	Recall@5	NDCG@5	Jaccard	Recall@5	NDCG@5	Jaccard	Recall@5	NDCG@5	Jaccard	Recall@5	NDCG@5	Jaccard
<i>Concatenation</i>	0.033±0.019	0.036±0.018	0.029±0.016	0.210±0.019	0.232±0.021	0.206±0.019	0.001±0.002	0.007±0.002	0.002±0.002	0.001±0.001	0.003±0.003	0.001±0.001
<i>MLP</i>	0.102±0.018	0.106±0.017	0.089±0.016	0.356±0.006	0.371±0.009	0.321±0.007	0.006±0.002	0.015±0.002	0.006±0.002	0.108±0.006	0.132±0.005	0.090±0.003
<i>GraphFusion</i>	0.111±0.013	0.122±0.014	0.097±0.010	0.369±0.006	0.389±0.008	0.328±0.007	0.019±0.002	0.045±0.003	0.019±0.002	0.119±0.006	0.143±0.005	0.096±0.003

information, and it neglects the structural relationships inherent in the demonstrations. The “MLP” strategy improves upon this by mapping the concatenated demonstrations into a shorter feature space, effectively reducing noise and emphasizing more relevant features. This approach yields better results than simple concatenation but still falls short compared to the “GraphFusion” strategy. The superior performance of the “GraphFusion” strategy highlights the importance of considering both the content and the structure of the demonstrations in the fusion process.

5.3.5 Time Complexity Analysis. The time complexity of our model RAG4DyG aligns with that of the vanilla Transformer, scaling as $O(n^2)$, where n denotes the sequence length. We measured the training time per epoch across various approaches using the UCI dataset to assess its efficiency. The results shown in Table 6 indicate that our model achieves faster or comparable training cost compared to the baseline methods. In contrast, approaches that integrate temporal components (e.g., RNNs or self-attention mechanisms) with structural elements (e.g., GNNs or GATs) face significant computational challenges due to the complexity of combining these modules.

6 Conclusion

In this work, we proposed RAG4DyG, a novel retrieval-augmented framework for dynamic graph modeling that overcomes the limitations of existing approaches by integrating broader temporal and contextual information. By leveraging the retrieval-augmented generation paradigm, RAG4DyG retrieves high-quality demonstrations from a retrieval pool and incorporates them effectively into the modeling process. The framework includes a time-aware contrastive learning module to prioritize temporally relevant samples and a graph fusion strategy to seamlessly integrate these retrieved demonstrations with the query sequence, enriching the historical

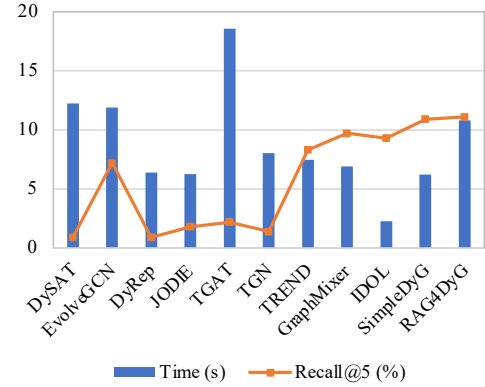


Figure 6: Time efficiency and Recall@5 of different methods.

context with extended temporal insights. Extensive experiments on diverse real-world datasets demonstrate the effectiveness of RAG4DyG in achieving state-of-the-art performance for dynamic graph modeling in both transductive and inductive scenarios.

Acknowledgments

This research / project is supported by the Ministry of Education, Singapore, under its Academic Research Fund Tier 2 (Proposal ID: T2EP20122-0041 and T2EP20123-0052). Any opinions, findings and conclusions or recommendations expressed in this material are those of the author(s) and do not reflect the views of the Ministry of Education, Singapore. The authors would also like to thank HOANG Thi Linh for her assistance with this work.

References

- [1] Uri Alon and Eran Yahav. 2020. On the Bottleneck of Graph Neural Networks and its Practical Implications. In *International Conference on Learning Representations*.
- [2] Deli Chen, Yankai Lin, Wei Li, Peng Li, Jie Zhou, and Xu Sun. 2020. Measuring and relieving the over-smoothing problem for graph neural networks from the topological view. In *AAAI Conference on Artificial Intelligence*, Vol. 34. 3438–3445.
- [3] Weilin Cong, Si Zhang, Jian Kang, Baichuan Yuan, Hao Wu, Xin Zhou, Hanghang Tong, and Mehrdad Mahdavi. 2022. Do We Really Need Complicated Model Architectures For Temporal Networks?. In *International Conference on Learning Representations*.
- [4] Huy Dao, Yang Deng, Dung D Le, and Lizi Liao. 2024. Broadening the View: Demonstration-augmented Prompt Learning for Conversational Recommendation. In *International ACM SIGIR Conference on Research and Development in Information Retrieval*. 785–795.
- [5] Songgaojun Deng, Huzefa Rangwala, and Yue Ning. 2019. Learning dynamic context graphs for predicting social events. In *ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. 1007–1016.
- [6] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. In *Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1*. 4171–4186.
- [7] Mingjiang Duan, Tongya Zheng, Yang Gao, Gang Wang, Zunlei Feng, and Xinyu Wang. 2024. DGA-GNN: Dynamic Grouping Aggregation GNN for Fraud Detection. In *AAAI Conference on Artificial Intelligence*, Vol. 38. 11820–11828.
- [8] Hui Fang, Tao Tao, and ChengXiang Zhai. 2004. A formal study of information retrieval heuristics. In *International ACM SIGIR Conference on Research and Development in Information Retrieval*. 49–56.
- [9] ZhengZhao Feng, Rui Wang, TianXing Wang, Mingli Song, Sai Wu, and Shuibing He. 2024. A Comprehensive Survey of Dynamic Graph Neural Networks: Models, Frameworks, Benchmarks, Experiments and Challenges. *arXiv preprint arXiv:2405.00476* (2024).
- [10] Yunfan Gao, Yun Xiong, Xinyu Gao, Kangxiang Jia, Jinliu Pan, Yuxi Bi, Yi Dai, Jiawei Sun, and Haofen Wang. 2023. Retrieval-augmented generation for large language models: A survey. *arXiv preprint arXiv:2312.10997* (2023).
- [11] Liangzhe Han, Bowen Du, Leilei Sun, Yanjie Fu, Yisheng Lv, and Hui Xiong. 2021. Dynamic and multi-faceted spatio-temporal deep learning for traffic speed forecasting. In *ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. 547–555.
- [12] Xiaoxin He, Yijun Tian, Yifei Sun, Nitesh V Chawla, Thomas Laurent, Yann LeCun, Xavier Bresson, and Bryan Hooi. 2024. G-retriever: Retrieval-augmented generation for textual graph understanding and question answering. *arXiv preprint arXiv:2402.07630* (2024).
- [13] Yuntong Hu, Zhihan Lei, Zheng Zhang, Bo Pan, Chen Ling, and Liang Zhao. 2024. GRAG: Graph Retrieval-Augmented Generation. *arXiv preprint arXiv:2405.16506* (2024).
- [14] Shenyang Huang, Farimah Poursafaei, Jacob Danovitch, Matthias Fey, Weihua Hu, Emanuele Rossi, Jure Leskovec, Michael Bronstein, Guillaume Rabusseau, and Reihaneh Rabbany. 2024. Temporal graph benchmark for machine learning on temporal graphs. *Advances in Neural Information Processing Systems* 36 (2024).
- [15] Paul Jaccard. 1901. Étude comparative de la distribution florale dans une portion des Alpes et des Jura. *Bull Soc Vaudoise Sci Nat* 37 (1901), 547–579.
- [16] Yugang Ji, Tianrui Jia, Yuan Fang, and Chuan Shi. 2021. Dynamic heterogeneous graph embedding via heterogeneous hawkes process. In *Machine Learning and Knowledge Discovery in Databases, Part I*. 388–403.
- [17] Xinke Jiang, Rihong Qiu, Yongxin Xu, Yichen Zhu, Ruizhe Zhang, Yuchen Fang, Chu Xu, Junfeng Zhao, and Yasha Wang. 2024. Ragraph: A general retrieval-augmented graph learning framework. *Advances in Neural Information Processing Systems* 37 (2024), 29948–29985.
- [18] Zhengbao Jiang, Frank F Xu, Luyu Gao, Zhiqing Sun, Qian Liu, Jane Dwivedi-Yu, Yiming Yang, Jamie Callan, and Graham Neubig. 2023. Active Retrieval Augmented Generation. In *Conference on Empirical Methods in Natural Language Processing*. 7969–7992.
- [19] Vladimir Karpukhin, Barlas Oguz, Sewon Min, Patrick Lewis, Ledell Wu, Sergey Edunov, Danqi Chen, and Wen-tau Yih. 2020. Dense Passage Retrieval for Open-Domain Question Answering. In *Conference on Empirical Methods in Natural Language Processing*. 6769–6781.
- [20] Jooyeon Kim, Eulrang Cho, Sehyung Kim, and Hyunwoo J Kim. 2024. Retrieval-Augmented Open-Vocabulary Object Detection. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 17427–17436.
- [21] Srikanth Kumar, William L Hamilton, Jure Leskovec, and Dan Jurafsky. 2018. Community interaction and conflict on the web. In *the World Wide Web Conference*. 933–943.
- [22] Srikanth Kumar, Xikun Zhang, and Jure Leskovec. 2019. Predicting dynamic embedding trajectory in temporal interaction networks. In *ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. 1269–1278.
- [23] Jure Leskovec, Jon Kleinberg, and Christos Faloutsos. 2005. Graphs over time: densification laws, shrinking diameters and possible explanations. In *ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. 177–187.
- [24] Lizi Liao, Le Hong Long, Zheng Zhang, Minlie Huang, and Tat-Seng Chua. 2021. MMConv: an environment for multimodal conversational search across multiple domains. In *International ACM SIGIR Conference on Research and Development in Information Retrieval*. 675–684.
- [25] Haotian Liu, Kilho Son, Jianwei Yang, Ce Liu, Jianfeng Gao, Yong Jae Lee, and Chunyuan Li. 2023. Learning customized visual models with retrieval-augmented knowledge. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 15148–15158.
- [26] Shangqing Liu, Yu Chen, Xiaofei Xie, Jing Kai Siow, and Yang Liu. 2021. Retrieval-Augmented Generation for Code Summarization via Hybrid GNN. In *International Conference on Learning Representations*.
- [27] Zhengyang Mao, Wei Ju, Yifang Qin, Xiao Luo, and Ming Zhang. 2023. Rahnet: Retrieval augmented hybrid network for long-tailed graph classification. In *ACM International Conference on Multimedia*. 3817–3826.
- [28] Pietro Panzarasa, Tore Opsahl, and Kathleen M Carley. 2009. Patterns and dynamics of users' behavior and interaction: Network analysis of an online community. *Journal of the American Society for Information Science and Technology* 60, 5 (2009), 911–932.
- [29] Aldo Pareja, Giacomo Domeniconi, Jie Chen, Tengfei Ma, Toyotaro Suzumura, Hiroki Kanezashi, Tim Kaler, Tao Schardl, and Charles Leiserson. 2020. Evolvegcn: Evolving graph convolutional networks for dynamic graphs. In *AAAI Conference on Artificial Intelligence*, Vol. 34. 5363–5370.
- [30] Stephen E Robertson and K Sparck Jones. 1976. Relevance weighting of search terms. *Journal of the American Society for Information science* 27, 3 (1976), 129–146.
- [31] Emanuele Rossi, Ben Chamberlain, Fabrizio Frasca, Davide Eynard, Federico Monti, and Michael Bronstein. 2020. Temporal graph networks for deep learning on dynamic graphs. *arXiv preprint arXiv:2006.10637* (2020).
- [32] Aravind Sankar, Yanhong Wu, Liang Gou, Wei Zhang, and Hao Yang. 2020. Dysat: Deep neural representation learning on dynamic graphs via self-attention networks. In *ACM International Conference on Web Search and Data Mining*. 519–527.
- [33] Weiping Song, Zhiping Xiao, Yifan Wang, Laurent Charlin, Ming Zhang, and Jian Tang. 2019. Session-based social recommendation via dynamic graph attention networks. In *ACM International Conference on Web Search and Data Mining*. 555–563.
- [34] Mengzhu Sun, Xi Zhang, Jiaqi Zheng, and Guixiang Ma. 2022. Ddgcnn: Dual dynamic graph convolutional networks for rumor detection on social media. In *AAAI conference on Artificial Intelligence*, Vol. 36. 4611–4619.
- [35] Haoran Tang, Shiqing Wu, Guandong Xu, and Qing Li. 2023. Dynamic graph evolution learning for recommendation. In *International ACM SIGIR Conference on Research and Development in Information Retrieval*. 1589–1598.
- [36] Rakshit Trivedi, Mehrdad Farajtabar, Prasenjeet Biswal, and Hongyuan Zha. 2019. Dyrep: Learning representations over dynamic graphs. In *International Conference on Learning Representations*.
- [37] Xiang Wang, Xiangnan He, Meng Wang, Fuli Feng, and Tat-Seng Chua. 2019. Neural graph collaborative filtering. In *International ACM SIGIR conference on Research and development in Information Retrieval*. 165–174.
- [38] Yanbang Wang, Yen-Yu Chang, Yunyu Liu, Jure Leskovec, and Pan Li. 2021. Inductive Representation Learning in Temporal Networks via Causal Anonymous Walks. In *International Conference on Learning Representations*.
- [39] Zhihao Wen and Yuan Fang. 2022. Trend: Temporal event and node dynamics for graph representation learning. In *ACM The Web Conference*. 1159–1169.
- [40] Yuxia Wu, Yuan Fang, and Lizi Liao. 2024. On the Feasibility of Simple Transformer for Dynamic Graph Modeling. In *ACM The Web Conference 2024*. 870–880.
- [41] Yuxia Wu, Lizi Liao, Gangyi Zhang, Wenqiang Lei, Guoshuai Zhao, Xueming Qian, and Tat-Seng Chua. 2022. State graph reasoning for multimodal conversational recommendation. *IEEE Transactions on Multimedia* (2022).
- [42] Xu Xie, Fei Sun, ZhaoYang Liu, Shiwen Wu, Jinyang Gao, Jiandong Zhang, Bolin Ding, and Bin Cui. 2022. Contrastive learning for sequential recommendation. In *IEEE International Conference on Data Engineering*. IEEE, 1259–1273.
- [43] Da Xu, Chuanwei Ruan, Evren Korpeoglu, Sushant Kumar, and Kannan Achan. 2020. Inductive representation learning on temporal graphs. (2020).
- [44] Chenchen Ye, Lizi Liao, Suyu Liu, and Tat-Seng Chua. 2022. Reflecting on experiences for response generation. In *ACM International Conference on Multimedia*. 5265–5273.
- [45] Jiasheng Zhang, Jialin Chen, Menglin Yang, Aosong Feng, Shuang Liang, Jie Shao, and Rex Ying. 2024. DTGB: A Comprehensive Benchmark for Dynamic Text-Attributed Graphs. *arXiv preprint arXiv:2406.12072* (2024).
- [46] Zulun Zhu, Kai Wang, Haoyu Liu, Jintang Li, and Siqiang Luo. 2024. Topology-monitorable Contrastive Learning on Dynamic Graphs. In *ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. 4700–4711.