LLM-enhanced Cascaded Multi-level Learning on Temporal Heterogeneous Graphs

Fengyi Wang
State Key Laboratory for Novel Software Technology
Nanjing University
Nanjing, China
fengyiwang@smail.nju.edu.cn

Guanghui Zhu*
State Key Laboratory for Novel Software Technology
Nanjing University
Nanjing, China
zgh@nju.edu.cn

Chunfeng Yuan
State Key Laboratory for Novel Software Technology
Nanjing University
Nanjing, China
cfyuan@nju.edu.cn

Yihua Huang
State Key Laboratory for Novel Software Technology
Nanjing University
Nanjing, China
yhuang@nju.edu.cn

ABSTRACT

Learning on temporal heterogeneous graphs (THGs) has attracted substantial attention in applications of information retrieval. Such graphs are ubiquitous in real-world domains like recommender systems and social networks. However, the spatial heterogeneity, rich semantic information, and intricate evolution patterns of THGs make it still difficult to generate high-quality embeddings for graph nodes. In this paper, we focus on two valuable and understudied issues related to THG learning: (a) How to capture the specific evolutionary characteristics of diverse temporal heterogeneous graphs? (b) Due to the heterogeneous nature of the graph, how to capture the unique temporal patterns of different node types? We explore these questions and present our solution by proposing a new method named CasMLN (**Cas**caded **M**ulti-level **L**earning **N**etwork) for THG learning. Through the multi-level learning structure and aggregation methods specifically designed for different levels, we obtain information of multiple levels and fuse them to improve embedding generation. Additionally, we pioneer the use of large language models (LLMs) in the THG field. By leveraging the universality and powerful capabilities of LLMs, our method introduces LLM-based external knowledge to effectively capture the implicit nature of graphs and node types, which helps to enhance type- and graph-level representations. We evaluate our method on several real-world THG datasets for different downstream tasks. Extensive experimental results show that CasMLN outperforms the state-of-the-art baselines in both accuracy and efficiency.

CCS CONCEPTS

• **Computing methodologies → Machine learning**.

*Corresponding author.

KEYWORDS

Temporal Heterogeneous Graphs, Graph Neural Network, Large Language Models, Representation Learning

ACM Reference Format:
Fengyi Wang, Guanghui Zhu, Chunfeng Yuan, and Yihua Huang. 2024. LLM-enhanced Cascaded Multi-level Learning on Temporal Heterogeneous Graphs. In *Proceedings of the 47th International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR '24), July 14–18, 2024, Washington, DC, USA.* ACM, New York, NY, USA, 10 pages. https://doi.org/10.1145/3626772.3657731

1 INTRODUCTION

Temporal heterogeneous graphs (THGs) provide a fundamental and accurate abstraction of real-world networks [31, 39, 40] as they have rich semantic, topological, and temporal information. Such graphs have different types of nodes and edges, and their topologies evolve over time [1]. They are ubiquitous in numerous scenarios, such as recommender systems [12, 29, 44], retrieval models [21], and relational databases [10, 14]. Representation learning on THGs [14, 30, 32, 34] aims to preserve the complex high-order graph information involving node features and graph structures in a low-dimensional embedding space, which requires effective knowledge extraction and dimension reduction. As powerful models for learning from THGs, temporal heterogeneous graph neural networks (THGNNs [2, 32]) have aroused a lot of attention in both academia and industry recently. Existing works mainly focus on aggregating different types of neighborhood information for nodes and dynamically updating node features. However, they ignore the two underlying natures of THGs:

- Graphs across different domains generally have their own unique characteristics and evolution patterns.
- The heterogeneity of THGs implies that different types of nodes have varying properties and change over time following independent processes.

Therefore, learning only at the node level makes the learned node representations drop much latent information and thus results in unsatisfactory performance in many downstream tasks.

Figure 1 illustrates two examples of temporal heterogeneous graphs in different domains. We investigate the frequencies of interactions occurring between different types of nodes. Diverse THGs

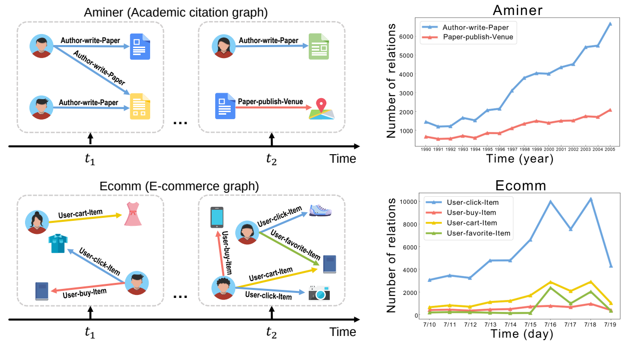


Figure 1: Frequencies of different types of interactions occurring between heterogeneous nodes in THG datasets.

typically have unique patterns of change. In the Aminer dataset, the numbers of various relations grow monotonically. While the number of interactions is relatively smooth or periodic in the e-commerce graph named Ecomm. This is consistent with the characteristics of both datasets. In the real world, the number of papers published per year is constantly increasing, while interactions in e-commerce are always temporally regular. Thus, it is necessary to learn the overall nature of THGs at the graph level. Moreover, as shown in Figure 1, interactions occurring across different node types in the same dataset typically exhibit specific evolution patterns. Only constructing graph-level features leads to the neglect of the heterogeneity of graphs. As a result, representation learning for distinct node types is also indispensable.

In this work, we aim to address the aforementioned problems by proposing a Cascaded Multi-level Learning Network (CasMLN [1]) for THGs. To effectively obtain multi-level information, we have designed specific aggregation strategies. First, we use graph convolution to initially generate node-level representations that aggregate information about different relations. Second, we devise a degree centrality-based strategy to aggregate node features into type-level representations, based on the insight that high-degree nodes tend to have more abundant and important topological information. Third, to generate graph-level representations, we employ an aggregation strategy based on feature importance, which adaptively weighs the impact of different type-level representations. In addition, by harnessing the superior power of large language models (LLMs [26, 33, 42]) and LLM-based embedding models, our method effectively enhances the multi-level representations. By refining the diverse properties implicit in summaries generated by LLMs, CasMLN integrates reliable LLM-based external knowledge. The LLM-based enhancement brings the following benefits:

- Since LLMs possess substantial real-world knowledge, their strong generalization ability helps the model to understand different THG datasets adaptively.
- Heterogeneous graphs generally suffer from skewed data distribution, resulting in the bias of model predictions [35]. LLM-based attribute augmentation helps to compensate for the lack of raw information [25].
- LLMs intuitively provide auxiliary information from an alternative view through mining semantic dependencies in real-world natural language.

Finally, we fuse representations aggregated at multiple levels and use temporal attention for time-domain perception to effectively summarize past chronological information.

In summary, our major contributions are as follows:

- **Novel cascaded multi-level learning approach on THGs.** We are the first to present the cascaded multi-level representation learning framework on THGs, to the best of our knowledge. Our innovation lies in effectively extracting multi-level evolutionary information to improve temporal node embeddings. Specific aggregation strategies based on degree centrality and feature importance are devised for learning at different levels.
- **Exploration of combining THG learning with LLMs.** We pioneer the use of powerful LLMs to capture the underlying properties of temporal heterogeneous graphs and node types. By innovatively introducing the external knowledge extracted by LLMs, our method can provide auxiliary information to enhance type-level and graph-level learning.
- **SOTA Performance.** We evaluate the proposed CasMLN on real-world THG datasets for various downstream tasks. Our model outperforms relevant baselines in cross-domain and cross-task scenarios in terms of both performance and efficiency, demonstrating the advantages of our insights and proposed method.

2 RELATED WORKS

2.1 Temporal Heterogeneous Graph Learning

Most representative graph neural networks can be used to model THGs, such as GCN [9], GAT [20], HAN [23], and HGT [6], but they ignore the important heterogeneity or dynamics of THGs. Recent years have witnessed remarkable progress on the development of THG learning methods. The prevalent way is to aggregate neighborhood information of different node types for each node and jointly consider temporal patterns. Different neighbor nodes typically contribute differently in information aggregation, thus the attention mechanism is widely applied to reflect the importance of neighbors. For example, HTGNN [2] proposes intra-relation aggregation to gather neighbor information within the same type of relation, and the inter-relation aggregation to summarize information of spatial neighbors over all relation types, both through attention coefficients. DyHATR [32] introduces hierarchical attention model to capture heterogeneity information of graph nodes and proposes a temporal attentive RNN model to capture deeper evolutionary patterns. DHGAS [41] explores to apply attention-based neural architecture search for dynamic heterogeneous graphs. To summarize, existing methods have focused on generating node-level representations, ignoring the regularities of macroscopic evolution.

2.2 THGNNs Applications in IR

Since many complex networks in the real world can be modeled as temporal heterogeneous graphs, THGNNs have been extensively studied and applied in the field of information retrieval (IR).

In many scenarios of recommender systems [27], the power of THG learning can be utilized to analyze user intents and predict interactions [28, 36, 43, 44]. PDGCN [8] converts interaction sequences into discrete dynamic heterogeneous graphs and learn the structural and temporal information with the path conditioned

[1]The code of CasMLN is available at https://github.com/PasaLab/CasMLN.

graph convolutional network. DHINs [29] introduces dynamic heterogeneous information networks to predict users' preferences. DHGCN [12] uses dynamic heterogeneous graph convolutional network for content-aware recommendation. ISRec [11] proposes modeling evolutionary heterogeneous graphs to perceive the structured intent transition of users for sequential recommendation.

THG learning can also be applied in the realm of search system and ranking algorithms. HEXA [21] is a heterogeneous graph-based context-aware ranking framework. It exploits heterogeneous graphs to organize the contextual information for obtaining ranking results. DHGAT [15] is a dual heterogeneous graph attention network integrated with the two-tower architecture, using the user interaction data from both shop search and product search. GraphTR [13] designs a graph neural network for tag-based ranking on huge heterogeneous networks. SSLF-GR [24] optimizes the ranking scores for webpage ranking by compute the transition and reset probability of the markov process in heterogeneous graphs.

In addition, THG learning also thrives in fields like query and Point-of-Interest (POI) matching [37], question-answering systems [4, 5], and query-focused summarization [16].

3 PRELIMINARIES

3.1 Heterogeneous Graph

A heterogeneous graph [22, 38] has various types of nodes and edges and is defined as: $\mathcal{G} = \{\mathcal{V}, \mathcal{E}\}$, where $\mathcal{V}$ and $\mathcal{E}$ denote the set of nodes and the set of edges, respectively. The graph is associated with a node type mapping function $\phi_n : \mathcal{V} \rightarrow C_n$ and an edge type mapping function $\phi_e : \mathcal{E} \rightarrow C_e$, with the constraint of $|C_n| + |C_e| \geq 2$. Different relations occur between specific types of nodes.

3.2 Temporal Heterogeneous Graph

A temporal heterogeneous graph can be viewed as a series of evolving heterogeneous graph snapshots, formally defined as $\mathcal{G} = \left(\left\{\mathcal{G}^t\right\}_{t=1}^{T}, \phi_n, \phi_e\right)$. $\mathcal{G}^t = (\mathcal{V}^t, \mathcal{E}^t)$ denotes the heterogeneous graph snapshot at t. The total number of timestamps is denoted as T. Also, $\phi_n : \mathcal{V} \rightarrow C_n$ is the node type mapping function and $\phi_e : \mathcal{E} \rightarrow C_e$ is the edge type mapping function, where $\mathcal{V} = \bigcup_{t=1}^{T} \mathcal{V}^t$, $\mathcal{E} = \bigcup_{t=1}^{T} \mathcal{E}^t$, and $|C_n| + |C_e| \geq 2$.

3.3 Temporal Heterogeneous Graph Representation Learning

The goal of temporal heterogeneous graph representation learning is to learn a model that generates a d-dimensional node representation $\mathbf{z}_v^T \in \mathbb{R}^d$ for each node $v \in \mathcal{V}$ based on the given historical snapshots $\mathcal{G} = \left(\left\{\mathcal{G}^t\right\}_{t=1}^{T}, \phi_n, \phi_e\right)$. The learned representations will be used for prediction in various downstream tasks at the next timestamp $T + 1$. High-quality node representations can simultaneously aggregate temporal patterns, heterogeneity, and latent spatial information from graph snapshots.

4 PROPOSED METHOD

4.1 Cascaded Multi-level Learning Framework

The overall framework of the proposed CasMLN is shown in Figure 2, which contains three levels, i.e., node-level, type-level, and graph-level learning. In the cascaded framework, the representations at the next level are aggregated from the ones at the previous level. First of all, the temporal graph at each timestamp undergoes a heterogeneous graph convolution operation in multiple layers, and the output node-level representations serve as the input for type-level learning. Next, we aggregate node representations based on degree centrality to obtain type-level representations, and further aggregate type-level representations based on feature importance to obtain graph-level representations. At both levels, we use the powerful LLM and an embedding model to generate embeddings that integrate external knowledge to augment the aggregated representations. Finally, we fuse the multi-level representations and use temporal attention to optimize node embeddings for downstream tasks. Next, we introduce each component of CasMLN in detail.

4.2 LLM-based External Knowledge Learning

Prompt-based learning provides textual instructions to large language models, which allows LLMs to adapt to distinct tasks and various data domains [3, 19]. We also utilize the power of prompt LLMs to learn the relevant external knowledge for THGs and integrate it into type-level and graph-level representations. We first construct a brief text-based description of the graph and an instruction that requires output in a fixed format, treating them as the prompt for the whole graph: $P(\mathcal{G})$ = {Introduction to the graph; Instruction}. Prompts of node types are devised as: $P(c)$ = {Introduction to type c; Instruction}, $\forall c \in C_n$. Examples of prompt construction can be found in Section 5.4. The LLM (GPT-3.5) is then utilized to generate summaries based on these descriptions, which helps discern implicit information from sequential texts. We use an embedding model (text-embedding-ada-002) to obtain semantic representations containing external knowledge of domain comprehension by encoding the summaries generated by the LLM. OpenAI's text-embedding-ada-002 model is an LLM-based model specifically adapted for embedding generation. Then the embeddings are fed into different MLPs for the purpose of dimensionality reduction:

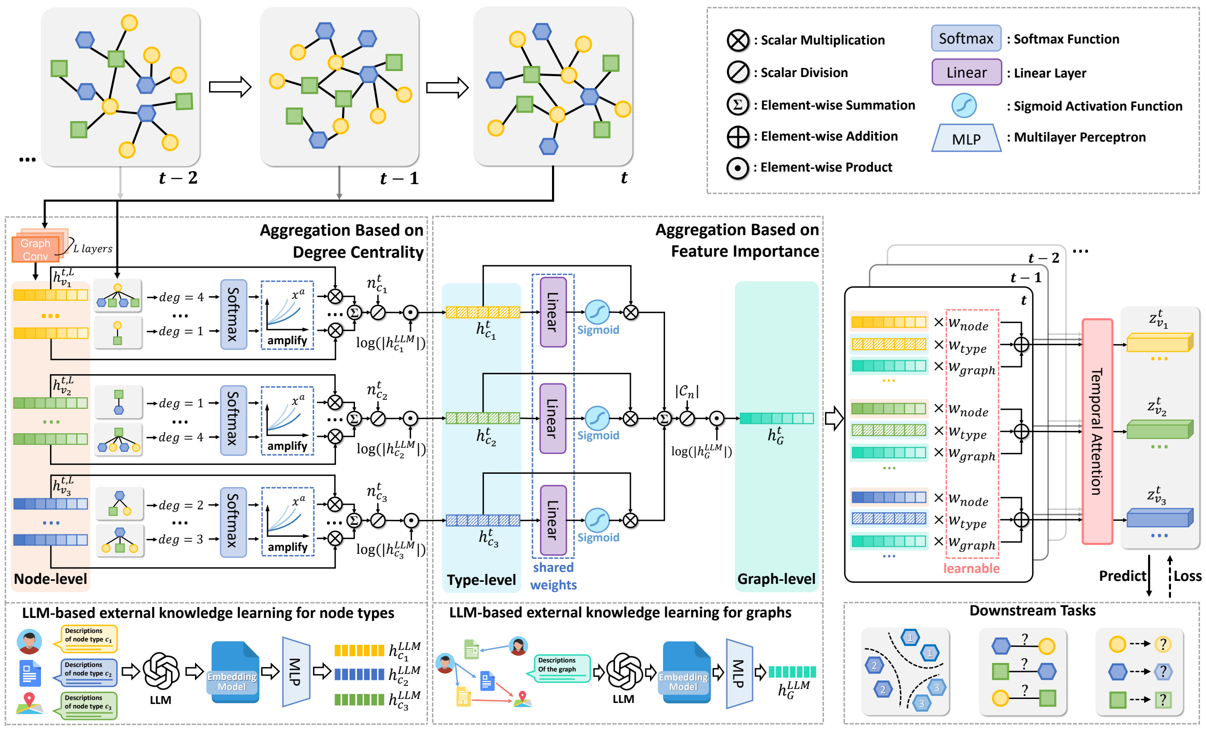
$$\begin{aligned} \mathbf{h}_G^{LLM} &= MLP_G(emb(LLM(P(\mathcal{G})))), \\ \mathbf{h}_c^{LLM} &= MLP_C(emb(LLM(P(c)))),\ \forall c \in C_n. \end{aligned} \tag{1}$$

emb denotes the text embedding model. $\mathbf{h}_G^{LLM} \in \mathbb{R}^d$ and $\mathbf{h}_c^{LLM} \in \mathbb{R}^d$ denote the latent representation vector of the graph $\mathcal{G}$ and node type c augmented through the external knowledge extracted by the LLM, respectively.

4.3 Node-level Learning Based on Graph Convolution

In each time slice, we first apply relational graph convolution [18] to aggregate information from neighboring nodes of different relation types for each node in the heterogeneous graph snapshot. $h_v^{t,(0)}$ is the input node feature at t. In order to distinguish the data for each relationship type, specialized transformation matrices for different types of relations are assigned at each convolutional layer:

$$\mathbf{h}_v^{t,(l+1)} = relu\left(\sum_{r \in C_e} \sum_{u \in \mathcal{N}_r^t(v)} \frac{1}{c_{v,r}} \left(\mathbf{W}_r^{(l)} \cdot \mathbf{h}_u^{t,(l)}\right) + \mathbf{W}_0^{(l)} \cdot \mathbf{h}_v^{t,(l)}\right), \tag{2}$$



t − 2
t − 1
t
: Scalar Multiplication
: Scalar Division
: Element-wise Summation
: Element-wise Addition
: Element-wise Product
Softmax : Softmax Function
Linear : Linear Layer
: Sigmoid Activation Function
MLP : Multilayer Perceptron
Graph Conv
L layers
Aggregation Based on Degree Centrality
Aggregation Based on Feature Importance
deg = 4
deg = 1
deg = 2
deg = 3
Softmax
amplify
Linear
Sigmoid
Node-level
Type-level
shared weights
Graph-level
Temporal Attention
learnable
Predict
Loss
LLM-based external knowledge learning for node types
Descriptions of node type c1
Descriptions of node type c2
Descriptions of node type c3
LLM
Embedding Model
MLP
LLM-based external knowledge learning for graphs
Descriptions Of the graph
Downstream Tasks

Figure 2: The overall framework of CasMLN .

where $\mathbf{h}_v^{t,(l)}$ is the hidden state of node v in the l-th graph convolutional layer at t. $\mathcal{N}_r^t(v)$ represents the set of relation-r-based neighbors of node v at t. i.e., given an edge type $r \in \mathcal{C}_e$ and a node v at timestamp t, its temporal relation-r-based neighbors is defined as $\mathcal{N}_r^t(v) = \{u \mid (u,v) \in \mathcal{E}^t, \phi_e(u,v) = r\}$. $\mathbf{W}_r^{(l)}$ denotes the transformation matrix corresponding to relation r at the l-th layer. $\mathbf{W}_0^{(l)}$ implies the self-loop. $c_{v,r}$ is the normalization constant. After iterative processing of the L-layer network, the output $\mathbf{h}_v^{t,L}$ is used as the node-level representation of v at timestamp t, which initially summarizes the topological information of the graph.

4.4 Type-level Learning Based on Degree Centrality

In graph theory, degree centrality is a significant criterion to measure the importance of nodes in the whole graph structure. Degree is the number of nodes that are connected to a given node, which helps to discover influential nodes in a network. For a given node type in a heterogeneous graph, nodes with a higher degree tend to have a stronger ability for information transmission and are more critical to the graph topology. Therefore, we devise an aggregation strategy based on degree centrality to generate type-level representations for heterogeneous graph snapshots at each timestamp.

For a given heterogeneous graph snapshot $\mathcal{G}_t$, we first count the degree $deg^t(v)$ of each node v (including both in-degree and out-degree). For each node type $c \in \mathcal{C}_n$, we use the softmax function and node degrees at t to quantify the impact of each node v belonging to type c as follows:

$$\tilde{w}^t(v) = \frac{exp\left(deg^t(v)\right)}{\sum_{u \in \mathcal{V}^t, \phi_n(u)=c} exp\left(deg^t(u)\right)}, \tag{3}$$

and $\phi_n(v) = c$. To further enlarge the differences in the effects of nodes with different degrees on their corresponding type-level information, we amplify the influence of node degrees according to the following equation:

$$w^t(v) = x^{\tilde{w}^t(v)}, \tag{4}$$

where x is the amplification factor, which is a hyper-parameter. Due to the fast magnification nature of the exponential function, the differences brought about by node degrees are further amplified. After calculating the importance of different nodes, we obtain type-specific representations through weighted averaging, which is computed as:

$$\hat{\mathbf{h}}_c^t = \frac{1}{n_c^t}\left(\sum_{v \in \mathcal{V}^t, \phi_n(v)=c} w^t(v) \cdot \mathbf{h}_v^{t,L}\right), \ \forall c \in \mathcal{C}_n, \tag{5}$$

where n_c^t is the number of nodes belonging to type c at timestamp t. Then we integrate the LLM-based representation vectors of node types with each $\hat{\mathbf{h}}_c^t$ to intuitively benefit from relevant external knowledge extracted by LLMs, which can be formally written as:

$$\mathbf{h}_c^t = \hat{\mathbf{h}}_c^t \odot log(|\mathbf{h}_c^{LLM}|), \tag{6}$$

where $\odot$ denotes the element-wise product. The purpose of the logarithmic transform is to control the values and retain the relative proportions of the original data.

4.5 Graph-level Learning Based on Feature Importance

Considering that the importance of different type-level information could vary in different graphs and tasks, we devise an aggregation strategy based on type-level feature importance to adaptively learn graph-level representations. This approach allows for better fusion of type-level representations into high-quality graph-level representations that provide insight into the underlying dependencies of heterogeneous graphs. Once the type-level representations at timestamp t are obtained, a weight that reflects the importance of each type-level representation is learned through a linear layer:

$$w^t(c) = \sigma(\mathbf{W} \cdot \mathbf{h}_c^t + b), \forall c \in C_n, \tag{7}$$

where $\mathbf{W} \in \mathbb{R}^{1\times d}$ and b are learnable parameters, and σ is the sigmoid activation function. Next, we aggregate all the type-level representations through weighted averaging to generate the graph-level representation:

$$\hat{\mathbf{h}}_G^t = \frac{1}{|C_n|}\left(\sum_{c\in C_n} w^t(c) \cdot \mathbf{h}_c^t\right), \tag{8}$$

Similarly, we utilize the LLM-based graph-level representation to enhance $\hat{\mathbf{h}}_G^t$:

$$\mathbf{h}_G^t = \hat{\mathbf{h}}_G^t \odot log(|\mathbf{h}_G^{LLM}|). \tag{9}$$

4.6 Temporal Modeling of Fused Multi-level Representations

To fuse the multiple levels of representations, we use learnable weights to capture the importance of learning at different levels and perform a weighted sum of the representations:

$$\tilde{\mathbf{z}}_v^t = \underbrace{w_{node} \cdot \mathbf{h}_v^{t,L}}_{node-level} + \underbrace{w_{type} \cdot \mathbf{h}_c^t}_{type-level} + \underbrace{w_{graph} \cdot \mathbf{h}_G^t}_{graph-level}, \tag{10}$$

where $\phi_n(v) = c$. w_{node}, w_{type}, and w_{graph} are learnable weights.

We explore a concise attention approach to capture deeper temporal patterns among continuous timestamps in the entire historical graph sequence. For all representations of each node v at different timestamps $\{\tilde{\mathbf{z}}_v^1, \tilde{\mathbf{z}}_v^2, \cdots, \tilde{\mathbf{z}}_v^t\}$, the attention scores are learned through a linear layer:

$$a_v^t = \sigma\left(\mathbf{W}_a \cdot \tilde{\mathbf{z}}_v^t + b_a\right), \ t = 1, 2, 3, \cdots, \tag{11}$$

where σ is the sigmoid activation function. $\mathbf{W}_a \in \mathbb{R}^{1\times d}$ and b_a are learnable parameters. The attention scores are then normalized using the softmax function as follows:

$$\tilde{a}_v^t = \frac{exp(a_v^t)}{\sum_{t'\le t} exp(a_v^{t'})}. \tag{12}$$

The normalized scores are then used to fuse all the sequential representations, so as to achieve the final node embedding used for prediction of downstream tasks:

$$\mathbf{z}_v^t = \frac{1}{t}\left(\sum_{t'\le t} \tilde{a}_v^{t'} \cdot \tilde{\mathbf{z}}_v^{t'}\right). \tag{13}$$

5 EXPERIMENTS

5.1 Datasets

We totally used four public real-world THG datasets for evaluation. The statistics of datasets are presented in Table 1, where the time window size controls how much historical data is included for prediction. Descriptions of datasets are as follows:

- Aminer [7] is a dataset about academic citations. The time slices are separated using the publication year (during 1990-2006).
- Ecomm [32] is a real-world heterogeneous bipartite graph of e-commerce, which mainly records shopping behaviors of users within daily snapshots from June 10th, 2019 to June 20th, 2019.
- Yelp [7] is a business review dataset, containing timestamped user reviews and tips on businesses. Three categories of businesses are extracted to construct the temporal graph, including "Fast Food", "Sushi", and "American (New) Food".
- COVID-19 [2] is an epidemic disease dataset, which contains both state and county level daily COVID-19 case reports in the US. It includes graph slices spanning from 05/01/2020 to 02/28/2021.

5.2 Baselines

We compare CasMLN with static/dynamic as well as homogeneous/heterogeneous graph learning methods, which are described below:

- **GCN** [9]: A classical static graph neural network that uses convolutional operations on homogeneous graph structures.
- **GAT** [20]: A classical static graph neural network that works on homogeneous graphs. The neighborhood information is aggregated through an attention mechanism.
- **RGCN** [18]: A static heterogeneous graph neural network that uses specialized transformation matrices for different relations.
- **HAN** [23]: A static heterogeneous graph neural network based on the hierarchical attention, including node-level and semantic-level attentions.
- **HGT** [6]: A static heterogeneous graph neural network that uses node- and edge-type dependent parameters to characterize the heterogeneous attention over each edge.
- **DySAT** [17]: A dynamic graph neural network that employs self-attention to aggregate structural neighborhood and temporal dynamics for node representation learning.
- **DyHATR** [32]: A dynamic heterogeneous graph neural network that uses attention to learn heterogeneous information.
- **HGT+** [6]: A dynamic heterogeneous graph neural network extended from HGT through the relative temporal encoding.
- **HTGNN** [2]: A dynamic heterogeneous graph neural network that utilizes various aggregation mechanisms.

5.3 Task Setup

5.3.1 Link Prediction. We perform link prediction on the Aminer and Ecomm datasets to evaluate different methods. For Aminer, the task is to predict links between author nodes, i.e., whether a pair of authors will coauthor a paper. As for Ecomm, the task is to predict whether a user will interact with an item. The relationships include click, buy, cart, and favorite. We adopt the widely used ROC-AUC and AP (Average Precision) as metrics. Negative edges are sampled randomly from unlinked nodes. For each two nodes, the inner product of the corresponding node embeddings is used for prediction, and the loss function is cross-entropy.

Table 1: Statistics of datasets

Dataset	Domain	# Nodes for each type	# Relations for each type	# Total Time Slices	Time Granularity	#Time Window Size
Aminer	Academic citation	# Paper : 18,464 # Author : 23,035 # Venue : 22	# Paper-publish-Venue : 18,464 # Author-write-Paper : 52,545	16	Year	13
Ecomm	E-commerce	# User : 1,476 # Item : 34,505	# User-click-Item : 57,917 # User-buy-Item : 5,529 # User-cart-Item : 15,066 # User-favorite-Item: 6,701	10	Day	7
Yelp	Business review	# User : 55,702 # Business : 12,524	# User-review-Business : 87,846 # User-tip-Business : 35,508	12	Month	12
COVID-19	Epidemic disease	# State : 54 # County : 3,223	# State-near-State : 269 # State-include-County : 3,141 # County-near-County : 22,176	304	Day	7

5.3.2 Node Classification. We perform node classification on the Yelp dataset to evaluate different methods. The specific task is to predict which of the 3 categories (Fast Food", "Sushi", and "American (New) Food") business nodes belong to. We adopt the widely used Macro-F1 and Recall as metrics. A fully-connected layer with the softmax activation function is applied as the classification layer, and cross-entropy is used as the loss function.

5.3.3 Node Regression. The node regression task conducted on the COVID-19 dataset aims to perform state-level daily new case forecasting. We adopt the widely used MAE (Mean Absolute Errors) and RMSE (Root Mean Squared Errors) as metrics to measure the node regression performance. A fully connected layer with the ReLU activation function is used as the predictor, and MAE Loss is used as the loss function.

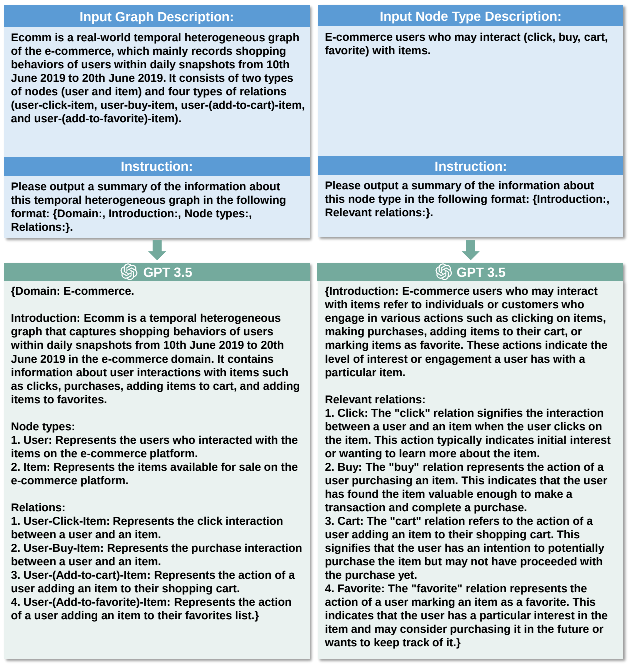
5.4 Implementation Details

For static graph models without considering temporal dependencies, we transform temporal graphs into static ones by merging all graph snapshots. For homogeneous graph models, we transform heterogeneous graphs into homogeneous ones by ignoring the node and edge types. The output embedding dimension is 8 for all models. We employ the Adam optimizer with a learning rate set to 1e-2 and weight decay set to 5e-4. We train each model for 500 epochs with an early stopping strategy with a patience of 50 and select the one with the best result on the validation set for testing. Our method is implemented with Python 3.9.0 and PyTorch 1.12.1. All experiments are conducted on a Tesla V100 GPU with 32GB memory.

To show how we constructed the prompts for our model, a specific example of summary generation for the e-commerce graph is present in Figure 3.

5.5 Quantitative Results

We compare CasMLN with various state-of-art baselines to evaluate performance in several downstream tasks, including link prediction, node classification, and node regression. All models are trained five times. The experimental results of test performance with mean and standard deviation are reported in Table 2. We omit the result for HGT+ on the COVID-19 dataset since the time encoding technique cannot be applied. CasMLN can achieve the best performance on all datasets, indicating the effectiveness of the multi-level learning structure and LLM-based enhancement. On the Ecomm and Yelp datasets, static approaches surprisingly yield better results. This may be due to that the temporal variations in these two datasets are less dramatic. In general, better modeling of the heterogeneity and evolution of THGs is required to achieve good performance.


Input Graph Description:

Ecomm is a real-world temporal heterogeneous graph of the e-commerce, which mainly records shopping behaviors of users within daily snapshots from 10th June 2019 to 20th June 2019. It consists of two types of nodes (user and item) and four types of relations (user-click-item, user-buy-item, user-(add-to-cart)-item, and user-(add-to-favorite)-item).

Instruction:

Please output a summary of the information about this temporal heterogeneous graph in the following format: {Domain:, Introduction:, Node types:, Relations:}.

GPT 3.5

{Domain: E-commerce.

Introduction: Ecomm is a temporal heterogeneous graph that captures shopping behaviors of users within daily snapshots from 10th June 2019 to 20th June 2019 in the e-commerce domain. It contains information about user interactions with items such as clicks, purchases, adding items to cart, and adding items to favorites.

Node types:
1. User: Represents the users who interacted with the items on the e-commerce platform.
2. Item: Represents the items available for sale on the e-commerce platform.

Relations:
1. User-Click-Item: Represents the click interaction between a user and an item.
2. User-Buy-Item: Represents the purchase interaction between a user and an item.
3. User-(Add-to-cart)-Item: Represents the action of a user adding an item to their shopping cart.
4. User-(Add-to-favorite)-Item: Represents the action of a user adding an item to their favorites list.}

Input Node Type Description:

E-commerce users who may interact (click, buy, cart, favorite) with items.

Instruction:

Please output a summary of the information about this node type in the following format: {Introduction:, Relevant relations:}.

GPT 3.5

{Introduction: E-commerce users who may interact with items refer to individuals or customers who engage in various actions such as clicking on items, making purchases, adding items to their cart, or marking items as favorite. These actions indicate the level of interest or engagement a user has with a particular item.

Relevant relations:
1. Click: The "click" relation signifies the interaction between a user and an item when the user clicks on the item. This action typically indicates initial interest or wanting to learn more about the item.
2. Buy: The "buy" relation represents the action of a user purchasing an item. This indicates that the user has found the item valuable enough to make a transaction and complete a purchase.
3. Cart: The "cart" relation refers to the action of a user adding an item to their shopping cart. This signifies that the user has an intention to potentially purchase the item but may not have proceeded with the purchase yet.
4. Favorite: The "favorite" relation represents the action of a user marking an item as a favorite. This indicates that the user has a particular interest in the item and may consider purchasing it in the future or wants to keep track of it.}

Figure 3: An example of constructing prompts for a THG (Ecomm) and a node type (User). The LLM outputs formatted summaries that include the LLM's universal understanding.

5.6 Ablation Study

The core contributing components of CasMLN are designed to effectively exploit the structural and temporal information in THGs, as well as to exploit the role of LLM-based knowledge enhancement. Therefore, we design several variants of our model to verify their effects. The details of these variants are as follows:

- **w/o graph convolution**: To study the impact of graph convolution operations, we remove the graph convolutional layers of CasMLN and denote this variant as w/o graph convolution.
- **w/o multi-level**: We only use the LLM-based representations without using the fused multi-level representations.
- **w/o temporal attention**: We don't use the temporal attention module and simply use the node embeddings at the most recent time slice for prediction.
- **w/o LLM**: We don't introduce LLM-based external knowledge to enhance type- and graph-level representations.

The results of the ablation experiments are presented in Table 3. Removing any component of CasMLN results in a performance

Table 2: Performance comparison between CasMLN and baselines in different downstream tasks. The bold and the underline indicate the best and the second-best, respectively.

Task	Link Prediction				Node Classification		Node Regression	
Dataset	Aminer		Ecomm		Yelp		COVID-19	
Metric	(AUC%)↑	(AP%)↑	(AUC%)↑	(AP%)↑	(F1%)↑	(Recall%)↑	(MAE)↓	(RMSE)↓
GCN	74.22±3.79	73.36±3.48	77.38±1.88	78.30±2.96	39.77±0.90	37.84±1.57	827±59	1462±50
GAT	80.30±0.93	78.32±1.28	79.27±1.48	76.51±1.55	38.10±1.97	38.91±0.73	828±120	1607±232
RGCN	83.57±0.39	81.33±0.42	<u>82.52±0.45</u>	<u>82.48±0.58</u>	38.01±1.14	40.31±0.15	838±92	1597±128
HAN	76.06±2.44	72.74±1.68	68.37±5.39	66.98±5.18	36.90±1.18	39.12±1.70	692±24	1434±45
HGT	77.76±1.27	76.72±1.55	80.54±2.04	78.61±1.62	<u>40.39±1.36</u>	40.63±0.49	640±15	1360±27
DySAT	66.94±0.23	69.86±0.75	75.13±1.00	73.42±1.44	39.24±2.28	37.85±4.20	570±13	1216±19
DyHATR	62.33±5.17	64.60±3.80	72.72±5.65	68.43±6.05	36.24±0.57	40.59±1.35	780±24	1509±41
HGT+	<u>85.16±1.06</u>	<u>83.04±1.35</u>	77.14±1.23	76.77±1.10	38.66±0.83	<u>40.81±0.64</u>	-	-
HTGNN	64.35±3.78	66.57±3.68	78.21±2.49	74.25±6.62	37.45±1.58	39.52±1.30	<u>560±21</u>	<u>1141±38</u>
CasMLN	**88.53±0.27**	**87.25±0.63**	**85.16±0.32**	**86.43±0.61**	**42.21±0.51**	**42.57±0.69**	**544±18**	**1119±12**
Improv.	+3.96%	+5.07%	+3.20%	+4.79%	+4.51%	+4.31%	+2.86%	+1.93%

degradation, where the removal of the multi-level learning results in the largest performance degradation. This corroborates the validity of our motivation for designing the multi-level learning structure.

In addition, to validate the necessity of learning at each level, we investigated the impact of corresponding representations, as shown in Table 4. The variants are described as follows:

- **w/o type-level**: To verify the importance of learning at the type level, we don't use the type-level representations for generating node embeddings.
- **w/o graph-level**: To justify the efficacy of learning at the graph level, we don't use the graph-level representations for generating node embeddings.

Ignoring graph-level or type-level information leads to different degrees of performance drop on different datasets, which is attributed to the fact that diverse THG datasets have distinct needs for representational information at different levels.

5.7 Efficiency

To further compare the efficiency and scalability of different methods, we count the average training time and maximum GPU memory consumption during training. In this experiment we set the network depth of different THG methods to be the same. As shown in Table 5 and Table 6, if we ignore the overhead imposed by the LLM, CasMLN excels in both training time and memory overhead, compared with other temporal graph methods. This implies the better efficiency and scalability of our method, which is mainly due to the simplicity of our multi-level aggregation methods and temporal attention.

5.8 Hyper-parameter Investigation

5.8.1 RGCN Layers. As depicted in Figure 4, CasMLN achieves optimal performance with only one layer of graph convolution on the Ecomm dataset. Whereas on the Aminer dataset, six layers of graph convolution are required. This may be because the academic citation network contains more types of nodes with complex dependencies, resulting in the need for higher-order node representations. There are only two node types in the e-commerce network: user and item. Edge types such as click, cart, and favorite tend to suggest similar attitudes of users towards the item. Thus, this graph does not contain particularly complex underlying relationships between nodes, although the number of edges is relatively high. Therefore, the Ecomm dataset does not require a deep graph convolutional network to aggregate the graph topological information.

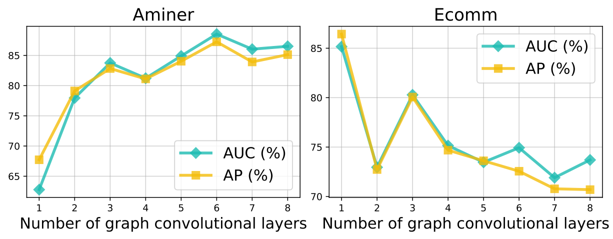


Figure 4: Impact of the number of graph convolutional layers.

5.8.2 Amplification factor. The performance of CasMLN with different values of the amplification factor using in Eq.4 is shown in Figure 5. The curve for Aminer dataset shows a relatively steady trend, which is because the degree differences among the nodes in Aminer are relatively small and the effect of amplification does not have a significant impact. The curve for the Ecomm dataset exhibits an upward and then downward trend, illustrating the role of amplifying the effect of node degrees. But too large amplification factors may lead to magnified effects of noise and degraded performance.

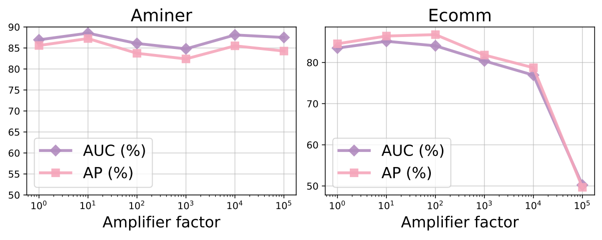


Figure 5: Impact of the amplification factor.

5.8.3 Embedding Dimension. Figure 6 exhibits a rising trend in performance as the embedding dimension increases, followed by a stabilization phase. To strike a balance between accuracy and efficiency, we set the embedding dimension to 8 for all datasets.

Table 3: Ablation study of CasMLN .

Task	Link Prediction				Node Classification		Node Regression	
Dataset	Aminer		Ecomm		Yelp		COVID-19	
Metric	(AUC%)↑	(AP%)↑	(AUC%)↑	(AP%)↑	(F1%)↑	(Recall%)↑	(MAE)↓	(RMSE)↓
w/o graph convolution	83.65±0.68	79.22±0.46	78.15±0.54	78.71±0.77	36.16±1.95	40.23±0.98	575±57	1144±67
w/o multi-level	63.10±0.85	68.17±0.88	73.26±1.29	73.80±1.10	34.73±1.91	39.64±1.72	635±23	1217±61
w/o temporal attention	76.92±2.44	78.52±2.46	81.39±0.73	82.09±1.37	37.02±2.07	40.80±0.56	584±39	1178±73
w/o LLM	86.57±1.40	84.91±1.50	82.66±0.84	83.56±0.72	39.05±3.91	40.73±2.01	576±28	1173±58
CasMLN	**88.53±0.27**	**87.25±0.63**	**85.16±0.32**	**86.43±0.61**	**42.21±0.51**	**42.57±0.69**	**544±18**	**1119±12**

Table 4: Impact of representations at different levels.

Task	Link Prediction				Node Classification		Node Regression	
Dataset	Aminer		Ecomm		Yelp		COVID-19	
Metric	(AUC%)↑	(AP%)↑	(AUC%)↑	(AP%)↑	(F1%)↑	(Recall%)↑	(MAE)↓	(RMSE)↓
w/o type-level	82.57±1.40	82.51±1.09	82.65±0.83	83.23±0.71	37.65±1.07	41.48±0.91	582±33	1147±45
w/o graph-level	81.55±2.62	80.97±1.70	83.15±1.01	83.43±1.25	35.24±3.64	40.51±0.59	571±29	1157±57
CasMLN	**88.53±0.27**	**87.25±0.63**	**85.16±0.32**	**86.43±0.61**	**42.21±0.51**	**42.57±0.69**	**544±18**	**1119±12**

Table 5: The training time (GPU seconds) of different temporal heterogeneous graph methods.

Model	Average training time per epoch			
	Aminer	Ecomm	Yelp	COVID-19
DyHATR	0.92	0.99	1.09	24.87
DySAT	0.37	0.46	0.34	13.23
HTGNN	1.07	1.12	1.23	20.78
CasMLN (w/o LLM)	**0.33**	**0.39**	**0.27**	**11.66**
CasMLN	0.44	0.51	0.47	16.05

Table 6: The GPU memory overhead of training different temporal heterogeneous graph models.

Model	GPU memory overhead (MB)			
	Aminer	Ecomm	Yelp	COVID-19
DyHATR	1155.28	835.3	2521.31	276.70
DySAT	914.66	286.14	1395.95	182.45
HTGNN	678.02	551.01	1657.18	204.27
CasMLN (w/o LLM)	**511.32**	**152.28**	**938.02**	**157.05**
CasMLN	512.42	152.91	938.65	157.67

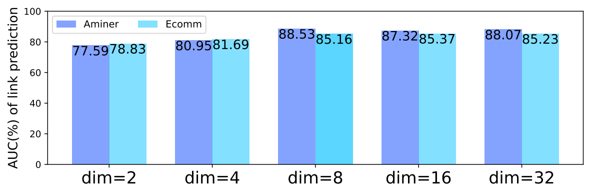


Figure 6: Impact of the embedding dimension.

5.9 Convergence Analysis

In this section, we delve into the convergence speed analysis of CasMLN with other THG methods. As depicted in Figure 7, our proposed CasMLN achieves convergence within 100 epochs. Whereas, HTGNN and DyHATR require more epochs to reach the best performance. Our approach not only improves performance but also notably reduces the training time.

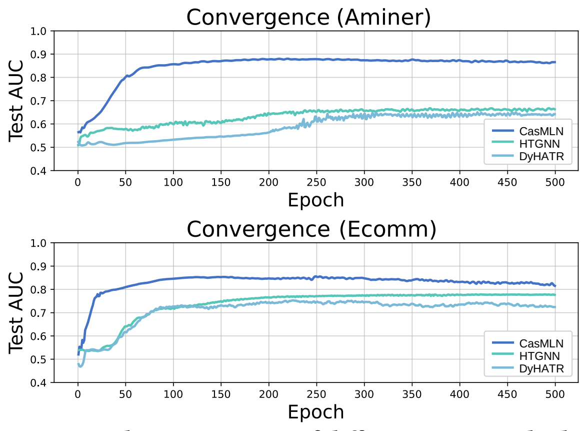


Figure 7: The convergence of different THG methods.

5.10 Visualization

In this section, we further analyze the temporal attention coefficients and evolving embeddings at multiple levels through visualization. The results and findings demonstrate the unique patterns of change across different graph domains and node types. This aligns with our motivation for applying multi-level learning to THGs.

5.10.1 Visualization of temporal attention. We present a visualization of the average temporal attention coefficients learned in Eq.11 of different node types to provide an intuitive understanding of how attention evolves over time in different THGs. For the Aminer dataset, the attention coefficients of venue nodes and the other two types of nodes show different trends of change. This is consistent with the diversity of venues and the regularity of paper publications. In the Ecomm and COVID-19 datasets, the temporal attention coefficients of node types show distinct trends, demonstrating the diversity of evolution patterns in different node types. Whereas, attention coefficients in Yelp are pretty steady, indicating the similarity in the importance of the graph structure at each time slice.

In summary, the temporal coefficients corresponding to different node types show very different changing trends, which validates the necessity of type-level representation learning.

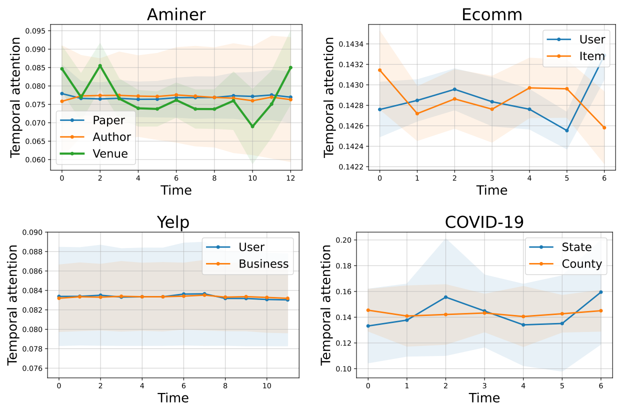


Figure 8: Visualization of temporal attention of different node types in different THG datasets.

5.10.2 Visualization of temporal embeddings. To intuitively understand how the overall evolutionary patterns of Temporal Heterogeneous Graphs in different domains change dynamically over time, we visualize the graph-level representations across time in Figure 9. In general, different THGs have distinctive graph-level feature distributions with certain temporal regularities. This implies that the graph-level representations learned by CasMLN somewhat capture the overall evolution of the graph.

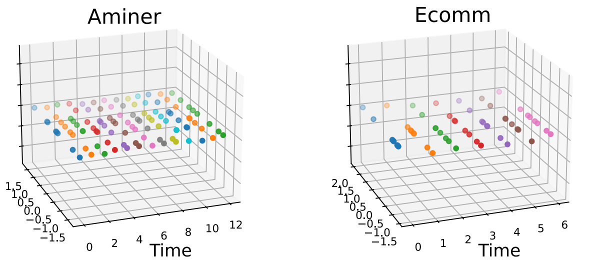


Figure 9: Evolving graph-level embeddings (8-dimensional) of Aminer and Ecomm datasets over time.

For type-level characterizations, we employ a PCA plot to visualize the type-level representations across time on different datasets. In Figure 10, representations of the same node type at each time slice are of the same color. It can be seen that even in the same graph, various type-level representations are disparately distributed in the feature space and show evolutionary trends with certain differences. This observation further reinforces the rationality behind our motivation for multi-level learning.

5.11 Case Study

To gain more intuitive insights into our model, we conduct a case study on the Ecomm dataset, as shown in Figure 11. Overall, there is a significant difference between the link probabilities output by CasMLN for real and fake edges, but this is ambiguous for CasMLN

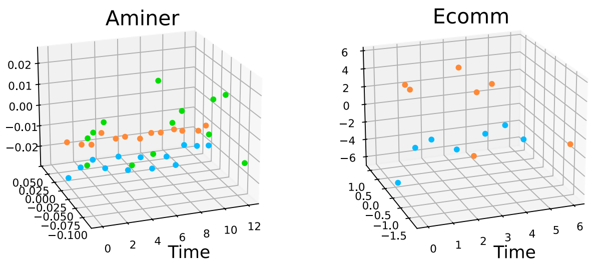


Figure 10: Evolving type-level embeddings over time. The embedding space is reduced to 2 dimensions through PCA.

w/o multi-level. Based on the case study, we can confirm the effectiveness of our method and the benefits of incorporating the cascaded multi-level learning.

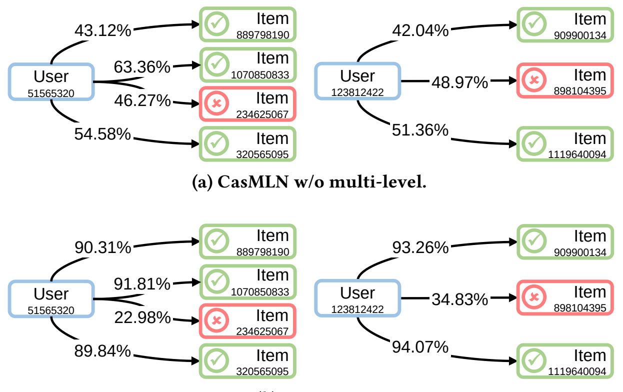


Figure 11: Visualization of examples of link prediction in the Ecomm dataset. The green color indicates a real link, otherwise red. The figure shows the link probabilities for different edges output by models.

6 CONCLUSION

In this paper, we proposed a novel cascaded multi-level representation learning method named CasMLN for THGs. Through degree centrality-based and feature importance-based aggregation, we obtain, respectively, the type- and graph-level representations. Moreover, we utilized the power of LLMs and embedding models to enhance type- and graph-level representations though LLM-based external knowledge extracted from summaries. Multi-level representations are fused and the corresponding evolution patterns are modeled by a concise temporal attention module for time-domain perception. Extensive experiments demonstrate the advantage of our model over state-of-the-art baselines in terms of both performance and efficiency.

ACKNOWLEDGMENT

This work was supported by the National Natural Science Foundation of China (#62102177), the Natural Science Foundation of Jiangsu Province (#BK20210181), and the Collaborative Innovation Center of Novel Software Technology and Industrialization, Jiangsu, China.

REFERENCES

[1] Ranran Bian, Yun Sing Koh, Gillian Dobbie, and Anna Divoli. 2019. Network embedding and change modeling in dynamic heterogeneous networks. In *Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval*. 861–864.

[2] Yujie Fan, Mingxuan Ju, Chuxu Zhang, and Yanfang Ye. 2022. Heterogeneous temporal graph neural network. In *Proceedings of the 2022 SIAM International Conference on Data Mining (SDM)*. SIAM, 657–665.

[3] Taoran Fang, Yunchao Zhang, Yang Yang, Chunping Wang, and Lei Chen. 2022. Universal Prompt Tuning for Graph Neural Networks. *arXiv preprint arXiv:2209.15240* (2022).

[4] Yue Feng, Zhen Han, Mingming Sun, and Ping Li. 2022. Multi-hop open-domain question answering over structured and unstructured knowledge. In *Findings of the Association for Computational Linguistics: NAACL 2022*. 151–156.

[5] Shen Gao, Yuchi Zhang, Yongliang Wang, Yang Dong, Xiuying Chen, Dongyan Zhao, and Rui Yan. 2022. HeteroQA: Learning towards question-and-answering through multiple information sources via heterogeneous graph modeling. In *Proceedings of the Fifteenth ACM International Conference on Web Search and Data Mining*. 307–315.

[6] Ziniu Hu, Yuxiao Dong, Kuansan Wang, and Yizhou Sun. 2020. Heterogeneous graph transformer. In *Proceedings of the web conference 2020*. 2704–2710.

[7] 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. Research Track: European Conference, ECML PKDD 2021, Bilbao, Spain, September 13–17, 2021, Proceedings, Part I 21*. Springer, 388–403.

[8] Mengyuan Jing, Yanmin Zhu, Yanan Xu, Haobing Liu, Tianzi Zang, Chunyang Wang, and Jiadi Yu. 2023. Learning shared representations for recommendation with dynamic heterogeneous graph convolutional networks. *ACM Transactions on Knowledge Discovery from Data* 17, 4 (2023), 1–23.

[9] Thomas N Kipf and Max Welling. 2016. Semi-supervised classification with graph convolutional networks. *arXiv preprint arXiv:1609.02907* (2016).

[10] Srijan Kumar, Xikun Zhang, and Jure Leskovec. 2019. Predicting dynamic embedding trajectory in temporal interaction networks. In *Proceedings of the 25th ACM SIGKDD international conference on knowledge discovery & data mining*. 1269–1278.

[11] Haoyang Li, Xin Wang, Ziwei Zhang, Jianxin Ma, Peng Cui, and Wenwu Zhu. 2021. Intention-aware sequential recommendation with structured intent transition. *IEEE Transactions on Knowledge and Data Engineering* 34, 11 (2021), 5403–5414.

[12] Tingting Liang, Lin Ma, Weizhong Zhang, Haoran Xu, Congying Xia, and Yuyu Yin. 2022. Content-aware recommendation via dynamic heterogeneous graph convolutional network. *Knowledge-Based Systems* 251 (2022), 109185.

[13] Qi Liu, Ruobing Xie, Lei Chen, Shukai Liu, Ke Tu, Peng Cui, Bo Zhang, and Leyu Lin. 2020. Graph neural network for tag ranking in tag-enhanced video recommendation. In *Proceedings of the 29th ACM international conference on information & knowledge management*. 2613–2620.

[14] Wenjuan Luo, Han Zhang, Xiaodi Yang, Lin Bo, Xiaoqing Yang, Zang Li, Xiaohu Qie, and Jieping Ye. 2020. Dynamic heterogeneous graph neural network for real-time event prediction. In *Proceedings of the 26th ACM SIGKDD international conference on knowledge discovery & data mining*. 3213–3223.

[15] Xichuan Niu, Bofang Li, Chenliang Li, Rong Xiao, Haochuan Sun, Hongbo Deng, and Zhenzhong Chen. 2020. A dual heterogeneous graph attention network to improve long-tail performance for shop search in e-commerce. In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. 3405–3415.

[16] Xiang Ren, Yujing Wang, Xiao Yu, Jun Yan, Zheng Chen, and Jiawei Han. 2014. Heterogeneous graph-based intent learning with queries, web pages and wikipedia concepts. In *Proceedings of the 7th ACM international conference on Web search and data mining*. 23–32.

[17] 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 *Proceedings of the 13th international conference on web search and data mining*. 519–527.

[18] Michael Schlichtkrull, Thomas N Kipf, Peter Bloem, Rianne Van Den Berg, Ivan Titov, and Max Welling. 2018. Modeling relational data with graph convolutional networks. In *The Semantic Web: 15th International Conference, ESWC 2018, Heraklion, Crete, Greece, June 3–7, 2018, Proceedings 15*. Springer, 593–607.

[19] Xiangguo Sun, Hong Cheng, Jia Li, Bo Liu, and Jihong Guan. 2023. All in One: Multi-Task Prompting for Graph Neural Networks. (2023).

[20] Petar Velickovic, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Lio, and Yoshua Bengio. 2018. Graph attention networks, international conference on learning representations. In *International Conference on Learning Representations*. 1–2.

[21] Shuting Wang, Zhicheng Dou, and Yutao Zhu. 2023. Heterogeneous Graph-based Context-aware Document Ranking. In *Proceedings of the Sixteenth ACM International Conference on Web Search and Data Mining*. 724–732.

[22] Xiao Wang, Deyu Bo, Chuan Shi, Shaohua Fan, Yanfang Ye, and S Yu Philip. 2022. A survey on heterogeneous graph embedding: methods, techniques, applications and sources. *IEEE Transactions on Big Data* 9, 2 (2022), 415–436.

[23] Xiao Wang, Houye Ji, Chuan Shi, Bai Wang, Yanfang Ye, Peng Cui, and Philip S Yu. 2019. Heterogeneous graph attention network. In *The world wide web conference*. 2022–2032.

[24] Wei Wei, Bin Gao, Tie-Yan Liu, Taifeng Wang, Guohui Li, and Hang Li. 2015. A ranking approach on large-scale graph with multidimensional heterogeneous information. *IEEE transactions on cybernetics* 46, 4 (2015), 930–944.

[25] Wei Wei, Xubin Ren, Jiabin Tang, Qinyong Wang, Lixin Su, Suqi Cheng, Junfeng Wang, Dawei Yin, and Chao Huang. 2023. LLMRec: Large Language Models with Graph Augmentation for Recommendation. arXiv:2311.00423 [cs.IR]

[26] Junchao Wu, Shu Yang, Runzhe Zhan, Yulin Yuan, Derek F Wong, and Lidia S Chao. 2023. A survey on llm-gernerated text detection: Necessity, methods, and future directions. *arXiv preprint arXiv:2310.14724* (2023).

[27] Shiwen Wu, Fei Sun, Wentao Zhang, Xu Xie, and Bin Cui. 2022. Graph neural networks in recommender systems: a survey. *Comput. Surveys* 55, 5 (2022), 1–37.

[28] Shu Wu, Yuyuan Tang, Yanqiao Zhu, Liang Wang, Xing Xie, and Tieniu Tan. 2019. Session-based recommendation with graph neural networks. In *Proceedings of the AAAI conference on artificial intelligence*, Vol. 33. 346–353.

[29] Tao Xie, Yangjun Xu, Liang Chen, Yang Liu, and Zibin Zheng. 2021. Sequential recommendation on dynamic heterogeneous information network. In *2021 IEEE 37th International Conference on Data Engineering (ICDE)*. IEEE, 2105–2110.

[30] Yuanzhen Xie, Zijing Ou, Liang Chen, Yang Liu, Kun Xu, Carl Yang, and Zibin Zheng. 2021. Learning and updating node embedding on dynamic heterogeneous information network. In *Proceedings of the 14th ACM International Conference on Web Search and Data Mining*. 184–192.

[31] Yu Xie, Bin Yu, Shengze Lv, Chen Zhang, Guodong Wang, and Maoguo Gong. 2021. A survey on heterogeneous network representation learning. *Pattern recognition* 116 (2021), 107936.

[32] Hansheng Xue, Luwei Yang, Wen Jiang, Yi Wei, Yi Hu, and Yu Lin. 2021. Modeling dynamic heterogeneous network for link prediction using hierarchical attention with temporal rnn. In *Machine Learning and Knowledge Discovery in Databases: European Conference, ECML PKDD 2020, Ghent, Belgium, September 14–18, 2020, Proceedings, Part I*. Springer, 282–298.

[33] Jingfeng Yang, Hongye Jin, Ruixiang Tang, Xiaotian Han, Qizhang Feng, Haoming Jiang, Bing Yin, and Xia Hu. 2023. Harnessing the power of llms in practice: A survey on chatgpt and beyond. *arXiv preprint arXiv:2304.13712* (2023).

[34] Ying Yin, Li-Xin Ji, Jian-Peng Zhang, and Yu-Long Pei. 2019. DHNE: Network representation learning method for dynamic heterogeneous networks. *IEEE Access* 7 (2019), 134782–134792.

[35] Yuxin Ying, Fuzhen Zhuang, Yongchun Zhu, Deqing Wang, and Hongwei Zheng. 2023. CAMUS: Attribute-Aware Counterfactual Augmentation for Minority Users in Recommendation. In *Proceedings of the ACM Web Conference 2023* (Austin, TX, USA) *(WWW '23)*. Association for Computing Machinery, New York, NY, USA, 1396–1404. https://doi.org/10.1145/3543507.3583538

[36] Chunyuan Yuan, Jiacheng Li, Wei Zhou, Yijun Lu, Xiaodan Zhang, and Songlin Hu. 2021. DyHGCN: A dynamic heterogeneous graph convolutional network to learn users' dynamic preferences for information diffusion prediction. In *Machine Learning and Knowledge Discovery in Databases: European Conference, ECML PKDD 2020, Ghent, Belgium, September 14–18, 2020, Proceedings, Part III*. Springer, 347–363.

[37] Zixuan Yuan, Hao Liu, Yanchi Liu, Denghui Zhang, Fei Yi, Nengjun Zhu, and Hui Xiong. 2020. Spatio-temporal dual graph attention network for query-poi matching. In *Proceedings of the 43rd international ACM SIGIR conference on research and development in information retrieval*. 629–638.

[38] Chuxu Zhang, Dongjin Song, Chao Huang, Ananthram Swami, and Nitesh V Chawla. 2019. Heterogeneous graph neural network. In *Proceedings of the 25th ACM SIGKDD international conference on knowledge discovery & data mining*. 793–803.

[39] Daokun Zhang, Jie Yin, Xingquan Zhu, and Chengqi Zhang. 2018. Network representation learning: A survey. *IEEE transactions on Big Data* 6, 1 (2018), 3–28.

[40] Ziwei Zhang, Peng Cui, and Wenwu Zhu. 2020. Deep learning on graphs: A survey. *IEEE Transactions on Knowledge and Data Engineering* 34, 1 (2020), 249–270.

[41] Zeyang Zhang, Ziwei Zhang, Xin Wang, Yijian Qin, Zhou Qin, and Wenwu Zhu. 2023. Dynamic Heterogeneous Graph Attention Neural Architecture Search. (2023).

[42] Wayne Xin Zhao, Kun Zhou, Junyi Li, Tianyi Tang, Xiaolei Wang, Yupeng Hou, Yingqian Min, Beichen Zhang, Junjie Zhang, Zican Dong, et al. 2023. A survey of large language models. *arXiv preprint arXiv:2303.18223* (2023).

[43] Guorui Zhou, Na Mou, Ying Fan, Qi Pi, Weijie Bian, Chang Zhou, Xiaoqiang Zhu, and Kun Gai. 2019. Deep interest evolution network for click-through rate prediction. In *Proceedings of the AAAI conference on artificial intelligence*, Vol. 33. 5941–5948.

[44] Guorui Zhou, Xiaoqiang Zhu, Chenru Song, Ying Fan, Han Zhu, Xiao Ma, Yanghui Yan, Junqi Jin, Han Li, and Kun Gai. 2018. Deep interest network for click-through rate prediction. In *Proceedings of the 24th ACM SIGKDD international conference on knowledge discovery & data mining*. 1059–1068.