World Wide Web (2024) 27:38
https://doi.org/10.1007/s11280-024-01280-5

Multi-temporal heterogeneous graph learning with pattern-aware attention for industrial chain risk detection

Ziheng Li[1] · Yongjiao Sun[1] · Xin Bi[1] · Ruijin Wang[2] · Shi Ying[3] · Hangxu Ji[1]

Received: 17 April 2024 / Revised: 18 May 2024 / Accepted: 29 May 2024 /
Published online: 15 June 2024

Abstract
Analyzing multi-channel data related to the industrial chain through graph representation learning is of significant value for industrial chain risk detection. Multi-channel data related to the industrial chain exhibits strong correlations, which enables such data to be modeled using graph structures. Furthermore, due to multi-channel data being multi-source, heterogeneous, and containing temporal information, industrial chain risk detection can be effectively conducted from the perspective of temporal heterogeneous graph representation learning. However, existing methods primarily obtain node features by sequentially processing the temporal heterogeneous information within a single graph, and they use attention based on discrete tokens to learn long-term information, which overlooks the multi-temporal information interactions between nodes and fails to effectively capture the temporal patterns in long-term information. To address these problems, we propose a multi-temporal heterogeneous graph learning with pattern-aware attention (MTHG-PA) for industrial chain risk detection. Firstly, we propose a method for interactive aggregation of multi-temporal heterogeneous information to learn node features, thereby avoiding the problem of insufficient information interaction between nodes across time. Secondly, we design a temporal pattern-aware attention module to learn the temporal patterns in long-term information, addressing the problem that traditional attention mechanisms can only handle discrete tokens. We conduct extensive experiments on three real-world datasets. The results show that our proposed method achieves superior performance in temporal heterogeneous graph learning compared to state-of-the-art methods.

Keywords Temporal heterogeneous graph · Graph representation learning · Graph neural network · Industrial chain risk detection

✉ Yongjiao Sun
sunyongjiao@mail.neu.edu.cn

[1] College of Computer Science and Engineering, Northeastern University, Shenyang 110819, China

[2] School of Information and Software Engineering, University of Electronic Science and Technology, Chengdu 610054, China

[3] School of Computing, Wuhan University, Wuhan 430072, China

1 Introduction

In the continuously evolving commercial environment, many countries are committed to enhancing the security and stability of the industrial chain. Industrial chain risk management [1–3] is a critical factor in ensuring the safe and robust operation of the industrial chain. However, enterprises cannot obtain comprehensive information on the industrial chain, limiting their ability to analyze and manage industrial chain risks. Nevertheless, industrial chain risks can be mapped and detected through easily accessible multi-channel data. Multi-channel data generated during enterprise operations encompasses various aspects, including transactions, transportation, warehousing, customs clearance, and insurance. We can better understand the operation of the industrial chain by analyzing multi-channel data. Therefore, multi-channel data is of significant application value for industrial chain risk management.

Multi-channel data [4, 5] possesses the characteristics of being multi-sourced, heterogeneous, and complexly interrelated, making it challenging for traditional risk detection methods based on statistical models or rule engines [6] to efficiently analyze the operation of the industrial chain. Therefore, we consider adopting graph representation learning [7] learning for correlation analysis to avoid information bias. Graph representation learning can construct graph models that reflect the complex relationships among various segments of the industrial chain, automatically learning the patterns and connections hidden in the data. This allows the risk status of the industrial chain to be analyzed.

In the electronics and electrical manufacturing industry, the operation of the industrial chain generates a variety of multi-channel data that can be utilized for risk detection. We construct these multi-channel data into a temporal heterogeneous graph and employ graph neural networks to identify abnormal patterns within the industrial chain, thereby detecting potential risks. These abnormal patterns are key factors that influence industrial chain risks. For instance, goods backlog caused by a decrease in inventory turnover rate, reduced customs efficiency, and import-export restrictions, or cost increases triggered by changes in tariff policies and rising transportation costs. Such scenarios could lead to reductions in production capacity and blockages at both the supply and demand ends, posing risks. Through the method proposed in this paper, we can help enterprises promptly identify and take preventive measures against potential industrial chain risk events, thereby avoiding or mitigating potential losses to businesses. This approach not only enhances the efficiency of risk management but also strengthens the ability of enterprises to cope with uncertainties such as supply chain disruptions. Figure 1 provides a detailed description of the entire process.

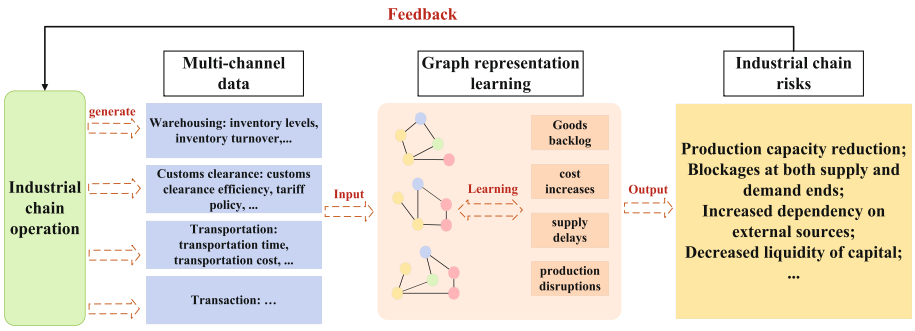


Figure 1 Industrial chain risk detection process

In recent years, with the widespread application of graph representation learning [8–10], numerous studies have employed graph neural networks in the field of risk management and control [11]. For instance, Trirat et al. [12] constructed static and temporal graph models in heterogeneous environmental data, utilizing a multi-view graph neural network with multiple attention modules to effectively learn multiple temporal graphs for correlated accident risk detection. Bi et al. [13] utilized financial news to establish relationships among shareholders of listed companies, constructing a tribe-style graph. They adopted tribal structure encoder and global graph representation learning methods to detect financial risks. Wang et al. [14] used static, dynamic, and long-term temporal modeling to represent users' multifaceted information. Although these methods achieve ideal results in specific types of graph structures, they have certain limitations when dealing with complex and correlated multi-channel data with strong temporal heterogeneous dependencies. Specifically, (1) Existing models typically adopt a sequential approach to process temporal heterogeneous information, often only aggregating the heterogeneous structural information of neighbors within a single timestamp. This approach overlooks the interaction of across-time information between nodes and inadequately explores the dynamic correlations in temporal heterogeneous graphs. Consequently, the generated node embedding does not adequately capture the multi-temporal heterogeneous information present in the graph. (2) Facing long-term time dependence, existing models employ attention mechanisms based on discrete tokens. Although these methods can handle long-term information, they often lack sensitivity to temporal patterns. This limits the model's ability to understand and capture complex temporal patterns in long-term continuous data. Here, temporal patterns refer to various time-related regularities and periodic features manifested in the data, such as seasonal variations, trend changes, and the influence of sudden events.

To address the above challenges, we propose a multi-temporal heterogeneous graph learning with pattern-aware attention (MTHG-PA) for industrial chain risk detection. Specifically, for the first challenge, we propose a method that can interactively aggregate heterogeneous information from different timestamps to capture multi-temporal heterogeneous information in strongly temporal-heterogeneous dependent data. For the second challenge, we design an attention learning module with temporal pattern-aware, which addresses the limitation of traditional attention mechanisms that can only handle long-term information using discrete token-based attention mechanisms.

The main contributions of this paper can be outlined as follows:

(1) We propose a method that interactively aggregates multi-temporal heterogeneous information to learn node features and obtain more effective node embedding representations of heterogeneous information across time.
(2) We design a temporal pattern-aware attention module that employs a temporal pattern attention mechanism to handle long-term information. This module is capable of capturing complex temporal patterns in long-term continuous data.
(3) We conduct experiments on three real-world datasets, including an authentic multi-channel dataset named CHANNELS, which contains information on industrial chain risks. Subsequently, we evaluate the performance of our graph mining task against state-of-the-art methods. The results demonstrate that our proposed MTHG-PA outperforms state-of-the-art baselines.

The remaining sections of this paper are organized as follows: Section 2 introduces the relevant work on risk management and associated technologies. Section 3 defines the problem addressed in this paper and describes the temporal heterogeneous graph structure. Section 4 outlines the overall framework and presents detailed designs. Section 5 presents and discusses

the experimental results. Section 6 offers conclusive remarks regarding the content discussed in this paper.

2 Related work

In this section, we first introduce the latest advancements in heterogeneous graph learning, particularly focusing on how graph neural networks handle heterogeneous information and complex structures. Next, we explore recent advancements in the field of temporal graph learning, focusing on both temporal homogeneous and heterogeneous graph neural networks. Finally, we review traditional methods for industrial chain risk detection. We also discuss approaches based on deep learning, which leverage advanced neural network models to enhance prediction accuracy and adaptability.

2.1 Heterogeneous graph learning

In recent years, the rapid development of Graph Neural Networks (GNNs) has facilitated significant progress in methods based on GNNs for heterogeneous graph learning [15–18]. Among them, Heterogeneous Graph Attention Network (HAN) [15] introduces node-level attention and semantic-level attention to evaluate the importance of different heterogeneous neighbor nodes and various meta-paths, thereby effectively learning the complex structure of heterogeneous graphs. HetGNN [16] utilizes random walk techniques to sample fixed-size, specific-type neighbor sets, effectively addressing the integration problem of heterogeneous information through advanced feature transformation modules and sequential aggregation strategies. HGT [17] employs the attention mechanism based on the transformer to dynamically adjust the weights of different meta-relationships, enhancing the model's ability to understand and express heterogeneous graph features.

These methods exhibit outstanding performance in capturing the multifaceted relationships and node attributes in static heterogeneous graphs. However, they demonstrate limitations when dealing with dynamic features associated with temporal heterogeneous graphs. It is worth noting that, the HGT model introduces positional encoding to solve this problem, but it treats the embedding of each node as static and unchanging, which is inconsistent with the dynamic characteristics of real-world scenarios. Therefore, existing models still face the challenge of effectively capturing the dynamics of temporal heterogeneous graphs in real-world applications.

2.2 Temporal graph learning

Temporal homogeneous GNNs [19–21] and spatial-temporal GNNs [22, 23] have received widespread attention and become a research hotspot in the field of graph learning. AdaDyGNN [20] introduces a novel knowledge adaptation mechanism, effectively solving the challenges of knowledge adaptation and integration in dynamic graph neural networks. OTGNet [21] employs a new message-passing mechanism that only transmits class-agnostic knowledge and selects key and diverse triadic subgraph structures, effectively addressing the challenges of information transfer between new and old class nodes in graphs and the issue of class knowledge forgetting. On the basis of these studies, academic circles are increasingly turning their attention to the development of temporal heterogeneous GNNs [24–27]. To be specific, THGCN [24] proposes an innovative heterogeneous GCN component to capture the

feature representation of heterogeneous graphs at each timestamp. By introducing a residual compression aggregation component, THGCN effectively learns temporal feature representations from continuous heterogeneous graphs. DyHATR [25] introduces a new method for temporal heterogeneous network embedding. It combines recurrent neural networks with a temporal attention mechanism to handle temporal information and utilizes a hierarchical attention mechanism to process heterogeneous information. This allows for better capture of the heterogeneity in static snapshots and the evolutionary patterns between consecutive snapshots. HPGE [26] addresses the challenge of effectively embedding dynamic heterogeneous graphs through utilizing a heterogeneous Hawkes process, allowing it to capture the complex structures and temporal dynamics within the graph.

Existing methods process the temporality and heterogeneity of data sequentially, achieving considerable results in specific graph structures. However, this sequential processing approach weakens the connection between the temporal and spatial domains. In addition, when processing long-term temporal dependencies, they capture temporal information discretely, making it difficult to capture temporal patterns in continuous data.

2.3 Industrial chain risk detection

Traditional methods for detecting industrial chain risk primarily rely on manually designed rules [28–30]. For instance, Khalilabadi et al. [29] propose a multi-stage stochastic integer programming model, which manages industrial chain risks through a customized progressive hedging algorithm. Sharma et al. [30] address unprecedented risks in the agricultural supply chain (ASCs) induced by the COVID-19 pandemic by adopting a strategy called Fuzzy Linguistic Quantifier Weighted Aggregation Operation (FLQ-QWAO). However, these manually defined rules are not only susceptible to attack but also struggle to adapt to the evolving operational patterns of the industrial chain. Therefore, a series of deep learning methods [31–33] have been proposed for industrial chain risk detection. Xu et al. [31] construct a deep learning model based on Long Short-Term Memory Networks (LSTM-NN), specifically designed to forecast the usage and popularity of shared bicycles at different time intervals. This enables more effective management of vehicle distribution and user demand, thereby optimizing operational efficiency. Vo et al. [32] combine multivariate Bidirectional Long Short-Term Memory (BiLSTM) networks with reinforcement learning to construct a Deeply Responsible Investment Portfolio (DRIP) model. This model addresses the challenge of optimizing investment decisions and portfolio configuration and achieves the dual objectives of social responsibility and economic benefits. Nikolopoulos et al. [33] introduce a deep learning model utilizing the LSTM architecture for forecasting the growth rates of the industrial chain amidst the COVID-19 pandemic.

These methods utilize historical and real-time data to predict potential risks and accordingly optimize operational strategies. Although they have achieved some success in industrial chain risk detection, they still exhibit shortcomings in adequately modeling the structural information relevant to real-world industrial chain scenarios.

3 Problem definition

In this section, we first construct the multi-channel data into a temporal heterogeneous graph. Then, we introduce the concept of temporal neighbors, which differ from the neighbors within the same graph. Temporal neighbors are defined as the same nodes at different timestamps,

laying the foundation for subsequent multi-temporal information aggregation. Finally, we define industrial chain risk detection as a multi-classification task and provide its representation within the graph.

Definition 1 (Temporal Heterogeneous Graph) *A temporal heterogeneous graph is denoted as* $G = (V, E, A, R, T, E^{'})$*, where* $T = \{t_1, t_2, \ldots t_T\}$ *is a set of timestamps.* $V = \{v_1^{t_1}, v_2^{t_1}, \ldots v_m^{t_T}\}$ *is a set of nodes,* $v_m^{t_T}$ *represent the m-th node at timestamp* t_T*,* $E = \{e_1^{t_1}, e_2^{t_1}, ..., e_q^{t_T}\}$ *is the set of edges,* $e_q^{t_T}$ *represent the q-th edge at timestamp* t_T*. A and R represent the sets of node types and edge relation types respectively, and there exist mapping function* $\varphi_1{:}V \rightarrow A$ *and* $\varphi_2{:}E \rightarrow R$*,* $\mid A \mid + \mid R \mid \geq 3$*.* $E^{'}$ *describes the temporal relationship between t and* $t + 1$*.*

This definition illustrates the structure of the temporal heterogeneous graph and describes its components and their interrelationships. Among the above definitions, the temporal neighbor related to the temporal relationship is defined as below.

Definition 2 (Temporal Neighbors) *At timestamp t, the neighborhood relationship of the n-th node v is defined as* $r \in R$*, the neighbor node is* $N_r^t(v_n) = \{u \mid (u, v_n) \in E^t, \varphi_2(u, v_n) = r\}$*. The temporal neighbor of the n-th node v is denoted as* $N_k^t(v_n) = \{v_n^t \mid t - K \leq t \leq t + K\}$*. These K temporal neighbors represent the same node v from* $t - k$ *to* $t + k$*.*

Utilizing the framework of temporal heterogeneous graphs, we can effectively represent and analyze the complex relationships and temporal dynamics of risk factors within multi-channel data. Given a temporal heterogeneous graph, denoted as $G = (V, E, A, R, T, E')$. Where, each node $v \in V$ represents a specific risk factor associated with the industrial chain. Each edge $e \in E$ represents the relationship between two risk factors. T and E' collectively describe the changes of each risk factor over consecutive timestamps. As time progresses, the multi-channel information contained within the graph continuously changes, which can be viewed as a process of state transitions within the industrial chain. Given this context, we provide the definition of industrial chain risk detection.

Definition 3 (Industrial Chain Risk Detection) *Industrial chain risk detection is regarded as identifying potential risks during the process of state transitions, specifically classifying predefined risk categories of industrial chain statuses over multiple timestamps.By learning a mapping function* $f{:}G \rightarrow Y$*, we frame the task as a graph multi-classification task.*

4 Proposed method

In this section, we introduce how to aggregate heterogeneous graph structural information to obtain heterogeneous information features for each graph. Subsequently, we propose an interactive method for processing multi-temporal heterogeneous graph information to enrich across-time information, enabling the model to discover patterns in temporal data. Next, we propose a temporal pattern-aware multi-head attention mechanism to capture temporal patterns within temporal data, providing more effective feature representations for downstream tasks.

4.1 Framework overview

The overall framework of the MTHG-PA is shown in Figure 2. The multi-temporal heterogeneous aggregation layer includes the same relationship aggregation, aggregation between

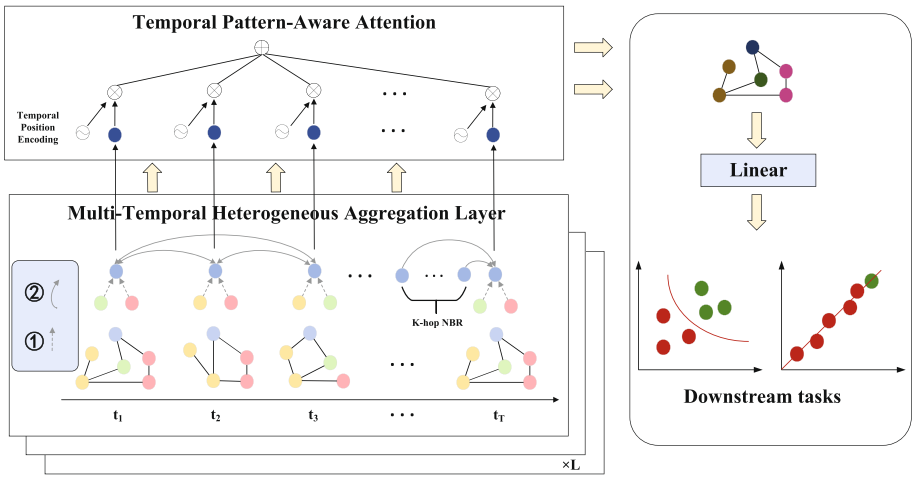


Figure 2 The overall framework of MTHG-PA

different relationships, and multi-temporal information aggregation. The same relationship aggregation and aggregation between different relationship modules are executed at each timestamp, aiming to describe spatial heterogeneity. On the other hand, the multi-temporal information aggregation module operates across multiple timestamps and focuses on aggregating information from a set of K-hop neighbors to capture temporal dependencies.

Within the same aggregation layer, all nodes continuously receive neighbor information from both the same relationship aggregation and aggregation between different relationships, followed by cross-temporal neighbor information aggregation. Throughout this process, the aggregated node embeddings at each layer are transmitted to the next layer. As the model depth increases, information continuously propagates iteratively across both temporal and spatial domains. Finally, the temporal pattern-aware attention module further extracts complex pattern information in the temporal heterogeneous graph along the temporal dimension, enhancing the model's ability to understand and express time series data.

4.2 Aggregation module for same relationship

Due to the inconsistency of node feature vector embeddings within a heterogeneous graph, it is necessary to align them before feeding these features into the aggregation model. The specific method involves applying different mappings to each node, projecting features of various dimensions onto the same feature space. The alignment process is denoted as:

$$H_{v^t}^{(0)} = W_r \cdot X_{N_r(v^t)} \tag{1}$$

where $X_{N_r(v^t)} \in \mathbb{R}^{d_{ini}}$ is the d_{ini}-dimensional initial feature vector, $W_r \in \mathbb{R}^{d \times d_{ini}}$ is a trainable specific mapping transformation matrix, $H_{N_r(v^t)}^{(0)} \in \mathbb{R}^d$ is the embedded representation of node v after mapping.

First, we aggregate nodes of the same relationship type from each heterogeneous graph at the previous layer $l-1$ to generate multiple relationship embeddings at timestamp t. For

each relationship type, the aggregation is as follows:

$$H_{N_r^t(v)}^{(l)} = \Re_{same}\left\{H_u^{(l-1)} \mid u \in N_r^t(v)\right\} = \sigma\left(\sum_{u \in N_r^t(v)} \alpha(u,v)_r^{(l),t} \cdot W_r^{(l),t} \cdot H_u^{(l-1)}\right) \tag{2}$$

where $H_u^{(l-1)}$ is the feature representation of neighbor nodes of the target node v at timestamp t in layer $l-1$ under relation r, and $H_{N_r^t(v)}^{(l)}$ represents the aggregated embedding of the target node v for a specific same type of relationship with nodes. when $l = 1$, the aligned node feature representations are directly used as input. $\sigma(\cdot)$ is LeakyReLU, and $W_r^{(l),t} \in \mathbb{R}^{d \times d}$ is a trainable transformation matrix. $\alpha(u,v)_r^{(l),t}$ represents the attention weights of neighbor nodes of the target node v. We use a self-attention mechanism to learn the weights of different nodes within the same relationship type. For a target node, given the relationship r, the calculation of the attention coefficients of the neighbor node $u = N_r^t(v)$ in layer l at timestamp t is as follows:

$$s(u,v)_r^{(l),t} = \sigma\left(\left[\alpha_r^{(l),t}\right]^T \cdot \left[W_r^{(l),t} \cdot H_v^{(l-1)} \parallel W_r^{(l),t} \cdot H_u^{(l-1)}\right]\right) \tag{3}$$

where $\parallel$ denotes the concatenation of vectors, and $\alpha_r^{(l),t} \in \mathbb{R}^{2d}$ is a trainable attention vector. Then, the computed attention coefficients are normalized using the softmax function to obtain the attention weights $\alpha(u,v)_r^{(l),t}$, calculated as follows:

$$\alpha(u,v)_r^{(l),t} = \frac{\exp\left(s(u,v)_r^{(l),t}\right)}{\sum_{u' \in N_r^t(v)} \exp\left(s(u',v)_r^{(l),t}\right)} \tag{4}$$

Based on the normalized weights of neighboring nodes, the embedding representation $H_{N_r^t(v)}^{(l)}$ of the target node for one relationship is obtained through weighted summation in layer l at timestamp t. The same approach is applied to other relationships as well.

4.3 Aggregation module between different relationships

The embeddings of multiple relation types obtained through $\Re_{same}\{\cdot\}$ aggregation are used as input, and we perform heterogeneous graph attention on these embeddings to obtain the final heterogeneous aggregated embedding representation. The aggregation method is as follows:

$$H_{v,R}^{(l)} = \Re_{diff}\left\{H_{N_r^t(v)}^{(l)} \mid r \in R(v)\right\} = \sum_{r \in R(v)} \beta_{v,r}^{(l),t} \cdot H_{N_r^t(v)}^{(l)} \tag{5}$$

In this equation, $R(v)$ represents the set of various relation features of node v at timestamp t in layer $l-1$, $H_{v,R}^{(l)}$ represents the heterogeneous feature representation of the target node v at timestamp t in layer l, and the attention weights of various relationships for the neighbors of target node v are represented by $\beta_{v,r}^{(l),t}$. We first perform a nonlinear transformation on the embeddings of different relationship types. Subsequently, we average and sum the transformed relationship embeddings to obtain a summation embedding. Attention coefficients are calculated by comparing the similarity between the summation embedding and

the relationship attention vector. For the target node, given a set of relationship types $R(v)$, the calculation of the relationship type attention coefficients of layer l at timestamp t is as follows:

$$s_r^{(l),t} = \frac{1}{|V_r^t|} \sum_{v' \in V_r^t} \left[q^T \tanh\left(W_R^{(l),t} \cdot H_{N_r^t(v)}^{(l)} + b \right) \right] \tag{6}$$

where V_r^t represents the set of nodes connected by relation r at timestamp t, $b \in \mathbb{R}^d$ is a bias vector, $q \in \mathbb{R}^d$ represents the attention vector, and $W_R^{(l),t} \in \mathbb{R}^{d \times d}$ is a trainable transformation matrix. Then, the calculated attention coefficients are normalized to obtain attention weights through the softmax function.The attention weight $\beta_r^{(l),t}$ is calculated as:

$$\beta_r^{(l),t} = \frac{\exp\left(s_r^{(l),t}\right)}{\sum_{r' \in R} \exp\left(s_{r'}^{(l),t}\right)} \tag{7}$$

Based on the normalized weights of different relationship types, the heterogeneous embedding representation $H_{v,R}^{(l)}$ of the target node is obtained through weighted summation in layer l at timestamp t. Figure 3 gives an intuitive explanation of the aggregation module for the same relationship and the aggregation module between different relationships.

4.4 Multi-temporal information aggregation module

In the same aggregation layer l, we interactively process temporal heterogeneous information and perform multi-temporal information aggregation for each node. Here, we use the embeddings aggregated through different relationships of the target node as input. To achieve the aggregation of multi-temporal information, we aggregate $2K$ temporal neighbor nodes of the target node, denoted as $\{v^t \mid t-K \le t \le t+K\}$, where $0 < K < T, t-K = \{t-K, 1\}_{\max}$, and $t+K = \{t+K, T\}_{\min}$. The aggregation method is as follows:

$$\begin{aligned} H_{v^t}^{(l)} &= \Re_{Knn}\left\{ H_{v,R}^{(l)} \mid v \in v^{t \pm K} \right\} \\ &= \sigma\left(\sum_{t \in (t \pm K)} \gamma_v^{(l),t} \cdot W_{TIME} \cdot H_{v^t,R}^{(l)} \right) \end{aligned} \tag{8}$$

where $v^{t \pm K}$ represents the temporal neighbor nodes used for multi-temporal information aggregation at layer l of the target node v, $H_{v^t}^{(l)}$ represents the multi-temporal heterogeneous

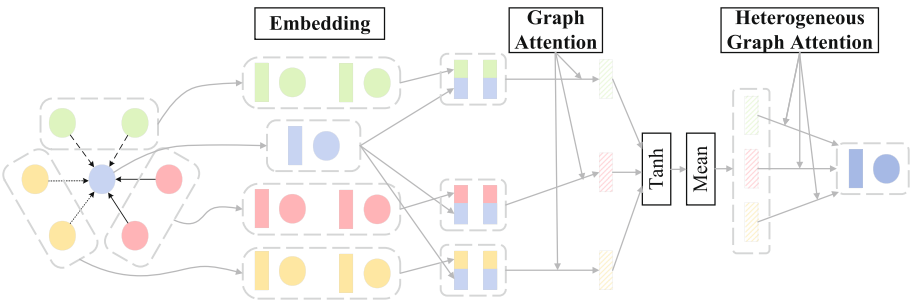


Figure 3 Detailed design of heterogeneous information aggregation

feature representation of the target node v at timestamp t in layer l, and $\gamma_v^{(l),t}$ represents the attention weights of the $2K$ temporal neighbors of the target node v. We utilize a self-attention mechanism to learn the weights of across-time neighboring nodes. When calculating the attention weights of the temporal neighbor nodes of node v, we introduce a temporal decay factor to enrich the representation of multi-temporal features. We use a time difference-based decay factor for nodes at different timestamps, denoted as $\delta_{t,u} = \exp^{-\lambda|t_v - t'_v|}$, where λ is the decay parameter, t_v and t'_v respectively represent the timestamps of node v at t and t'. This approach strengthens the influence of temporal neighbors, so we can better capture across-time features. The attention coefficients of temporal neighbor nodes are calculated as follows:

$$s_v^{(l),t} = \delta_{t,u} \cdot \left(\gamma^T \left[W_{TIME} \cdot H_{v^t,R}^{(l)} \parallel W_{TIME} \cdot H_{v^{t'},R}^{(l)}\right]\right) \tag{9}$$

where $\gamma^T \in \mathbb{R}^d$ represents the attention vector, and $W_{TIME} \in \mathbb{R}^{d \times d}$ denotes the trainable transformation matrix. After normalization using the softmax function, we obtain the attention weights for temporal neighbor nodes, denoted as:

$$\gamma_v^{(l),t} = \frac{\exp\left(s_v^{(l),t}\right)}{\sum_{t' \in (t \pm K)} \left(\exp(s_v^{(l),t'}\right)} \tag{10}$$

Then, we perform a weighted sum of the temporal neighboring node embeddings to calculate the node embedding representation after multi-temporal information aggregation, denoted as $H_{v^t}^{(l)}$. The entire process utilizes the interaction and integration of multi-temporal heterogeneous information to obtain the node embedding representation of across-time heterogeneous information. A graphical explanation of the multi-temporal information aggregation Module is shown in Figure 4.

Algorithm 1 Multi-temporal heterogeneous aggregation algorithm.

Require: temporal heterogeneous Graph G, multi-time K
Ensure: node embedding at all time after multi-temporal heterogeneous aggregation

$H_{v^t}^{(0)} \Leftarrow W_r \cdot X$
for $t = 1$ to T **do**
  for $l = 1$ to L **do**
    $s(u,v)_r^{(l),t} \Leftarrow$ Attention Score $\left(H_v^{(l-1)}, H_u^{(l-1)}\right)$
    $\alpha \Leftarrow softmax\left(s(u,v)_r^{(l),t}\right)$
    $H_{N_r^t(v)}^{(l)} \Leftarrow$ sum of weights $\sum \alpha \cdot H_u^{(l-1)}$
    $s_r^{(l),t} \Leftarrow$ Attention Score $\left(\frac{1}{|V_r^t|} \sum H_{N_r^t(v)}^{(l)}\right)$
    $\beta \Leftarrow softmax\left(s_r^{(l),t}\right)$
    $H_{v,R}^{(l)} \Leftarrow$ sum of weights $\sum \beta \cdot H_{N_r^t(v)}^{(l)}$
    $t' = t \pm K$
    $s_v^{(l),t} \Leftarrow$ Attention Score $\left(H_{v^t,R}^{(l)}, H_{v^{t'},R}^{(l)}\right)$
    $\gamma \Leftarrow softmax\left(s_v^{(l),t}\right)$
    $H_{v^t}^{(l)} \Leftarrow$ sum of weights $\sum \gamma \cdot H_{v^t,R}^{(l)}$
  end for
end for
return $H_{v^1}, H_{v^2}, \ldots, H_{v^T}$

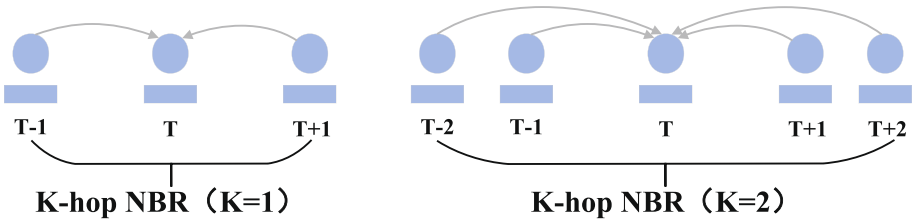


Figure 4 Multi-temporal aggregation manner

The above three modules collectively form the multi-temporal heterogeneous aggregation algorithm, and the pseudocode is listed in Algorithm 1.

4.5 Temporal pattern-aware attention module

The self-attention mechanism is a mechanism used for processing sequential data. It captures dependencies between elements by computing the correlation between each element in a sequence. It maps the input sequence into queries, keys, and values, and then calculates their similarities to generate weights for each position, thus highlighting important information.

The multi-head attention mechanism is an extension of the self-attention mechanism, which utilizes multiple sets of queries, keys, and values to compute different attention representations. By applying a linear transformation to the input and splitting it into multiple heads, each head independently executes self-attention computation. Subsequently, the results from these heads are concatenated to form the final result. This enables the model to capture relationships within the input sequence more comprehensively, enhancing the expressive ability of the model.

The multi-head self-attention mechanism is allowed to capture correlations between elements in a sequence, regardless of their distance. This effectively extends the model's ability to capture long-term information, enabling it to better understand complex associations within the sequence. In the case of complex relationships in the industrial chain, the flexible attention mechanism can help the model make long-term predictions accurately. However, it treats temporal data as discrete points, thereby overlooking various complex relationship patterns that may occur in the short term within the industrial chain. Therefore, it cannot efficiently capture temporal patterns.

To address the aforementioned problem, we design a temporal pattern-aware multi-head attention mechanism that can sense temporal patterns in continuous data from a global perspective. We first add a time factor to the embeddings by using a temporal position encoding function. The calculation is as follows:

$$\tilde{\mathrm{H}}_{v^t} = H_{v^t} + PE(\cdot) \tag{11}$$

where $PE(\cdot)$ is a temporal position encoding function, and $PE(p,2i) = \sin(p/10000^{2i/d_{\mathrm{model}}})$, $PE(p,2i+1) = \cos(p/10000^{2i/d_{\mathrm{model}}})$, where $p \in \{1,2,\ldots T\}$ denotes the temporal position, and i represents the dimension index. By feeding embeddings of different timestamps into this function, we make them discriminable in the temporal dimension. After temporal position encoding, we utilize the proposed temporal pattern-aware multi-head attention to learn the input features of nodes. Subsequently, the output of the proposed method, denoted as H_v^{final}, is used for downstream tasks. Mathematically, the output of our temporal

pattern-aware multi-head attention is denoted as follows:

$$H_v^{\hat{\text{inal}}} = Head_{LTA}(Q, K, V) = Concat(LTAhead_1 ... LTAhead_h)W_o \tag{12}$$

where $W_o \in \mathbb{R}^{h_{LTA} \cdot d_v \times d_{LTA}}$ represents the output projection matrix, h_{LTA} denotes the number of heads in the encoder, d_v represents the output dimension of each head, and d_{LTA} represents the output dimension of the encoder. In addition, $LTAhead_j$ denotes the output heads of the encoder and is calculated as:

$$LTAhead_j = PatternATTN(W_j^Q \cdot Q_{LTA}, W_j^K \cdot K_{LTA}, W_j^V \cdot V) \tag{13}$$

where $W_j^Q, W_j^K, W_j^V \in \mathbb{R}^{d_{hidden} \times d}$ represents the trainable transformation matrices for query (Q), key (K), and value (V), d_{hidden} represents the hidden layer dimension. The calculations for Q_{LTA}, K_{LTA} and $PatternATTN$ are as follows:

$$Q_{LTA} = \Phi_V \left(\widetilde{\mathrm{H}}_{v^t} \right), \quad K_{LTA} = \Phi_V \left(\widetilde{\mathrm{H}}_{v^t} \right) \tag{14}$$

$$PatternATTN(Q, K, V) = \text{softmax} \left(\frac{Q_{LTA} \left(K_{LTA} + P_{attern} PK_{attern} \right)^T}{\sqrt{D}} \right) \cdot \mathrm{V} \tag{15}$$

For each head, before applying self-attention, we introduce a convolution operation to compute Q and K, where $\mathbf{\Phi} : \mathbb{R}^{d_{hidden}} \rightarrow \mathbb{R}^D$ represents the parameters of the convolution kernel, where $D > d_{hidden}$. It maps the original feature space of queries (Q) and keys (K) to a higher-dimensional space to facilitate more complex feature interactions. Specifically, to enable the model to learn and recognize the importance of different temporal patterns during training, we introduce the combination of pattern keys and pattern matrices(PPK), denoted as $P_{attern} PK_{attern}$. Here, P_{attern} and PK_{attern} represent trainable pattern matrix and pattern key, respectively. P_{attern} represents key temporal patterns in temporal heterogeneous graph data, including long-term trend changes and sudden changes, such as periodic changes in the industrial chain due to currency policy and sudden changes caused by natural disasters or political events. Through the interaction of KEY and specific PPK, the model can adjust the weight of specific patterns. With these adjusted attention weights, the model can better understand the complex pattern relationships between temporal graph data, thereby improving its generalization ability and interpretability. The intuitive explanation of the temporal pattern-aware attention module is shown in Figure 5.

4.6 Loss function

By stacking L heterogeneous and multi-temporal aggregation layers, we could generate embedding for each node at each timestamp, denoted as $H_{v^t}^{(l)}$. Then, we take the embeddings aggregated through the temporal pattern-aware attention module as the final embeddings, denoted as H_v^{final}. MTHG-PA can be trained end-to-end on labeled data at timestamp $T+1$, as shown below:

$$\Gamma = \sum_{v \in V^L} \left(G\left(y_v, \hat{y}_v\right) + \lambda \parallel \theta \parallel_2^2 \right), \hat{y}_v = \sigma \left(MLP \left(H_v^{final} \right) \right) \tag{16}$$

where $G(\cdot)$ measures the loss between the true values y_v and the predicted scores $\hat{y}_v$, and $\parallel \theta \parallel_2^2$ is the $L2$ regularization term used to prevent overfitting. Depending on the objectives of different tasks, $G(\cdot)$ can be set as the cross-entropy loss for node classification and link prediction tasks or the mean absolute error for regression tasks.

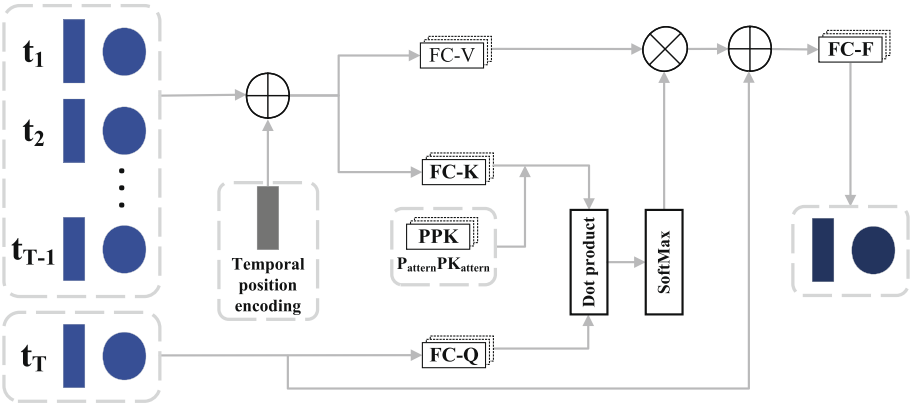


Figure 5 Detailed design of temporal pattern-aware attention module

5 Experiments

In this section, we first introduce the datasets used to evaluate the proposed method and list the baseline models used for comparison. Next, we provide detailed descriptions of the experimental settings to ensure the replicability and transparency of the results. We conduct performance evaluations for both classification and regression tasks to comprehensively demonstrate the efficacy of the proposed method. Furthermore, through ablation experiments, we explore the impact of key modules of the model on performance. Finally, hyperparameter sensitivity is conducted to further investigate the relationship between model performance and hyperparameter settings. The entire experimental section is designed to comprehensively and systematically validate the effectiveness and robustness of our proposed method across different tasks and settings.

5.1 Datasets

We evaluate our model using three real-world datasets. For classification tasks, we utilize the academic network dataset OGBN-MAG and the multi-channel dataset CHANNELS. For node regression tasks, we employ the COVID-19 epidemic network dataset. Each dataset is chosen to reflect distinct aspects of network analysis, ensuring that our model's performance is robust across varying types of network data.

OGBN-MAG [34]: We first employ the OGBN-MAG dataset, extracted from the Microsoft Academic Graph, which comprises 17,764 authors, 282,039 articles, 34,601 topics, and 2,276 institutions and covers a span of 10 years. To ensure its rationality, we specifically select authors who have consistently published at least one article per year throughout this period. Then we construct a temporal heterogeneous graph with four types of relationships for experiments.

COVID-19 [35]: We use a COVID-19 dataset containing 54 states and 3,223 counties, spanning from 05/01/2020 to 02/28/2021. The dataset is sourced from https://coronavirus.1point3acres.com/en, which reports daily confirmed cases, new cases, deaths, and recoveries for both states and counties. We transform this dataset into a temporal heterogeneous graph with three types of relationships for experiments.

CHANNELS: We construct a CHANNELS dataset, which comprises data from 8 different channels and encompasses 73 risk factors, spanning from 01/01/2021 to 01/01/2023. The dataset is provided by enterprises and contains authentic channel data, including risk information within the industrial chain. The graph corresponding to each timestamp represents the status of risk factors in the multi-channel data, reflecting whether there are risks in the industrial chain. We construct this dataset into a temporal heterogeneous graph with three types of relationships for experiments.

We partition each dataset into subsets according to the following ratios: 80% for training, 10% for validation, and 10% for testing. Table 1 shows the details of the three datasets.

5.2 Baselines

Our experiments are compared against four categories of state-of-the-art baseline methods, including neural sequence-based methods, homogeneous graph methods, heterogeneous graph methods, and temporal graph methods.

Neural sequence-based methods: These baselines are capable of capturing temporal dependencies. LSTM [36] is a special type of recurrent neural network (RNN) architecture that handles long-term dependencies through gating mechanisms. Transformer [37] employs the self-attention mechanism to balance the influence of different inputs in processing temporal data.

Homogeneous graph methods: These baselines capture homogeneous structural information. GCN [38] utilizes convolutional operations to the graph structure, effectively capturing local connectivity patterns and feature relationships between vertices. GAT [39] introduces an attention mechanism to dynamically weigh the importance of neighboring nodes during the feature aggregation process.

Heterogeneous graph methods: For static heterogeneous GNNs, we chose RGCN [18] and HGT [17] that do not rely on meta-paths. RGCN distinguishes diverse relationships between nodes by applying different weights to different types of edges. HGT applies a transformer architecture to learn mutual attention among each type of meta-relationship.

Temporal graph methods: temporal graph baselines include a spatiotemporal GNN, a temporal homogeneous GNN, and two temporal heterogeneous GNNs. Among them, CoGNN [22] is a spatiotemporal GNN that processes node features of time series through multilayer perceptrons and aggregates spatial information using GCN with skip connections. DySAT [19] is a temporal homogeneous GNN that captures temporal and structural features of nodes in the graph efficiently using the self-attention mechanism for node representation learning. HPGE [26] and DyHATR [25] are both temporal heterogeneous GNNs. HPGE combines heterogeneous attention and Hawkes processes to model the heterogeneity and temporality of the graph. DyHATR captures heterogeneous information in dynamic environments by adopting a hierarchical attention mechanism, combining RNN and temporal attention to capture time dependencies.

5.3 Experimental settings

For the homogeneous graph model, we don't consider the heterogeneity of the graph but directly use the structural information of the entire graph as the input of the model. In the experiments with the best results, we select the Adam optimizer due to its efficiency in handling sparse gradients and its adaptability in different applications. The learning rate is set to 4e-3, a value determined through a series of preliminary tests that suggest this rate

Table 1 Statistics of datasets

Dataset	Node	Relation	Time granularity	Time range	Data split
OGBN-MAG	# Author: 17,764	Author-Paper(Owns)	Year	2010 - 2019	Training:80%
	# Paper: 282,039	Paper-Paper(Cites)			Validation:10%
	# Topic: 34,601	Paper-Field(Affiliated)			Testing:10%
	# Institution: 2,276	Author-Institution(Affiliated)			
COVID-19	# State: 54	State-State(Near)	Day	05/01/2020-	Training:80%
	# County: 3,223	State-County(Includes)		02/28/2021	Validation:10%
		County-County(Near)			Testing:10%
CHANNELS	#Channels:8	channel-risk factor(Includes)	Day	01/01/2021-	Training:80%
	#Risk Factor:73	channel-channel(interrelated)		01/01/2023	Validation:10%
		risk factor-risk factor(interrelated)			Testing:10%

optimizes the convergence speed without causing instability in the training process. A weight decay of 5e-4 is used to regularize the model by mitigating the risk of overfitting through the penalization of large weights. A dropout rate of 0.45 is selected after experimentation; it provides the best trade-off between model complexity and performance, effectively reducing overfitting while maintaining sufficient capacity to learn relevant patterns. The empirical evidence suggests that setting the number of aggregation layers at 2 is optimal. Increasing the layers beyond this number tends to yield diminishing performance returns. This decline may be attributed to the over-smoothing effect, where node features become excessively similar. Considering the complexity and size of the datasets, we configure different dimensions for the hidden layers and set distinct patience values for the three datasets to ensure the model captures sufficient feature interactions while preventing overfitting. On the OGBN-MAG dataset, the hidden layer dimension is 32 with a patience of 50. On the COVID-19 dataset, the hidden layer dimension is 4 with a patience of 12. And on the CHANNELS dataset, the hidden layer dimension is 8 with a patience of 15. We used LeakyReLU as the activation function and trained all models for a fixed 500 epochs. Then we reported the test performance. The implementation of all baselines and MTHG-PA models is carried out using Python 3.9.16, PyTorch 2.0.1, and Deep Graph Library (DGL) 0.9.1. The experiments are conducted on a machine equipped with an Intel Core i7 13700K CPU, NVIDIA GeForce GTX 4090 Ti GPU, and 128GB RAM.

5.4 Performance evaluation for classification tasks

We conduct performance evaluations and comparative experiments on the OGBN-MAG dataset from the Microsoft Academic Graph and the CHANNELS dataset provided by companies. We respectively address the problem of predicting new co-author relationships among authors and detecting industrial chain risks. We partition the datasets into training, validation, and test sets in an 8:1:1 ratio. The link prediction task is defined as predicting co-author links that do not exist at timestamp T but may appear at timestamp $T + 1$. The risk detection task is defined as detecting the types of risks that may have occurred in the past T timestamp. We set the time window size of the MAG dataset to $T = 3$ and the time window size of the CHANNELS dataset to $T = 30$. Through the MTHG-PA model, we obtain graph embedding representations and train classification tasks using cross-entropy loss. We evaluate the performance of the model using two common metrics: AUC (Area Under Curve) and AP (Average Precision).

The experimental results are presented in Table 2, with the best performance highlighted in bold. From the results, we can draw the following conclusions:(1) Graph representation learning models capable of handling heterogeneous information outperform models that can only handle homogeneous graphs, highlighting the importance of modeling heterogeneous information. (2) When the model simultaneously considers temporal and heterogeneous information, performance is significantly improved, demonstrating the importance of temporal and spatial dependencies in enhancing prediction accuracy. (3) Our proposed MTHG-PA model outperforms all other baselines by capturing across-time information and temporal patterns, demonstrating its efficiency and effectiveness in handling complex networks.

5.5 Performance evaluation for regression tasks

We conduct performance evaluation and comparison experiments for regression tasks on the COVID-19 datasets. Specifically, we follow the same data partitioning approach as

Table 2 Results of comparison experiments

Model	OGBN-MAG		COVID-19		CHANNELS
	AUC	AP	MAE	RMSE	AUC
LSTM	81.37 ± 0.42	78.56 ± 0.29	720 ± 115	1504 ± 258	69.87 ± 0.63
Transformer	82.89 ± 0.36	79.81 ± 0.25	691 ± 54	1458 ± 182	70.15 ± 0.50
GCN	78.81 ± 1.53	76.46 ± 1.95	846 ± 101	1674 ± 204	67.56 ± 1.49
GAT	80.23 ± 2.07	77.58 ± 1.74	821 ± 91	1612 ± 211	68.11 ± 2.01
RGCN	80.34 ± 2.21	78.11 ± 1.65	833 ± 95	1640 ± 198	68.19 ± 2.13
HGT	85.30 ± 1.20	82.68 ± 1.20	805 ± 88	1598 ± 200	71.49 ± 1.21
CoGNN	85.87 ± 1.52	83.11 ± 0.92	653 ± 45	1298 ± 60	71.81 ± 1.38
DySAT	86.36 ± 0.24	83.83 ± 0.29	672 ± 74	1373 ± 96	72.08 ± 0.52
HPGE	88.88 ± 0.73	86.18 ± 0.82	656 ± 66	1303 ± 81	73.59 ± 0.90
DyHATR	89.49 ± 0.65	86.24 ± 0.91	643 ± 36	1282 ± 59	73.96 ± 0.86
MTHG-PA (ours)	$\mathbf{91.58 \pm 0.75}$	$\mathbf{89.83 \pm 1.05}$	$\mathbf{502 \pm 17}$	$\mathbf{1107 \pm 30}$	$\mathbf{75.83 \pm 0.93}$

OGBN-MAG and CHANNELS, using an 8:1:1 ratio, and set the task as a node value prediction problem. We set the time window size for the COVID-19 dataset to $T = 12$, which means that predictions are based on historical data from the past 12 days to forecast future values. After obtaining node embedding using our model, we train node and edge inputs using the Mean Absolute Error. Subsequently, we evaluate the performance of the model using two widely used metrics: MAE (Mean Absolute Error) and RMSE (Root Mean Square Error).

In Table 2, we present the comparative experimental results of the MTHG-PA model against other baselines in terms of MAE and RMSE metrics, highlighting the optimal performance in bold. The results demonstrate the node prediction performance across 54 states in the COVID-19 dataset. In addition to the conclusions drawn from the classification tasks, we observe that temporal models outperform static graph models in terms of performance. This result emphasizes the importance of temporal models for effectively capturing crucial information, particularly for data where the graph structure remains stable but node features change over time. This result further confirms the necessity of considering time factors in temporal data analysis, especially in scenarios like epidemic spread and industrial chain dynamics, where accounting for time is crucial for improving prediction accuracy.

5.6 Ablation experiments

The MTHG-PA model consists of three key modules: heterogeneous information aggregation, multi-temporal information aggregation, and temporal pattern-aware attention. To validate the effectiveness of each module, we design the following three sets of experiments:

(1) Validation of the effectiveness of the heterogeneous information aggregation module: In this experiment, we modify the heterogeneous information aggregation part of the model to handle homogeneous graphs. Specifically, we remove the information aggregation steps between different relationships described in the model and directly aggregate all neighbor nodes of the target node through mean pooling. This modification aims to evaluate the contribution of heterogeneous relationship aggregation to the model performance. We express it with MTHG-PA_w/o_het.

(2) Validation of the effectiveness of the multi-temporal information aggregation module: To validate the importance of the multi-temporal information aggregation module in highlighting patterns, we conduct experiments by removing this module, specifically setting $K = 0$. This means that we directly use node features aggregated through heterogeneous information aggregation without multi-temporal information aggregation. Through this approach, we can validate the impact of the multi-temporal information aggregation module on the overall model performance. We express it with MTHG.
(3) Validation of the effectiveness of the temporal pattern-aware attention module: In this experiment, we replace the improved multi-head attention mechanism $Head_{LTA}$ with the traditional multi-head attention mechanism. This is done to aggregate the node information after multi-temporal information aggregation in a discrete manner, thereby validating the effectiveness of the temporal pattern-aware attention module in capturing temporal patterns. We express it with HG-PA.

The results of the ablation experiments are shown in Table 3. In the first set of module effectiveness validation experiments, the experimental group utilizing the heterogeneous relation aggregation module outperform the control group MTHG-PA_w/o_het. This demonstrates that the aggregation of heterogeneous information is crucial for effectively capturing dependencies between nodes and for more deeply learning the structural features of heterogeneous graphs. In the evaluation of the effectiveness of the multi-temporal information aggregation module, the performance of the model is higher than HG-PA but slightly lower than MTHG-PA after removing this module. This result indicates that our designed multi-temporal information aggregation module can significantly enhance the representational capability of nodes and effectively enrich across-time information. In the experiment of the temporal pattern-aware attention module, after replacing this module with the traditional multi-head attention mechanism, the model performance is lower than MTHG-PA, confirming the effectiveness of our proposed temporal pattern-aware attention module in capturing temporal patterns. In summary, these findings underscore the significant contributions of the three key modules included in our proposed MTHG-PA model to improve the final performance.

5.7 Hyperparameter sensitivity

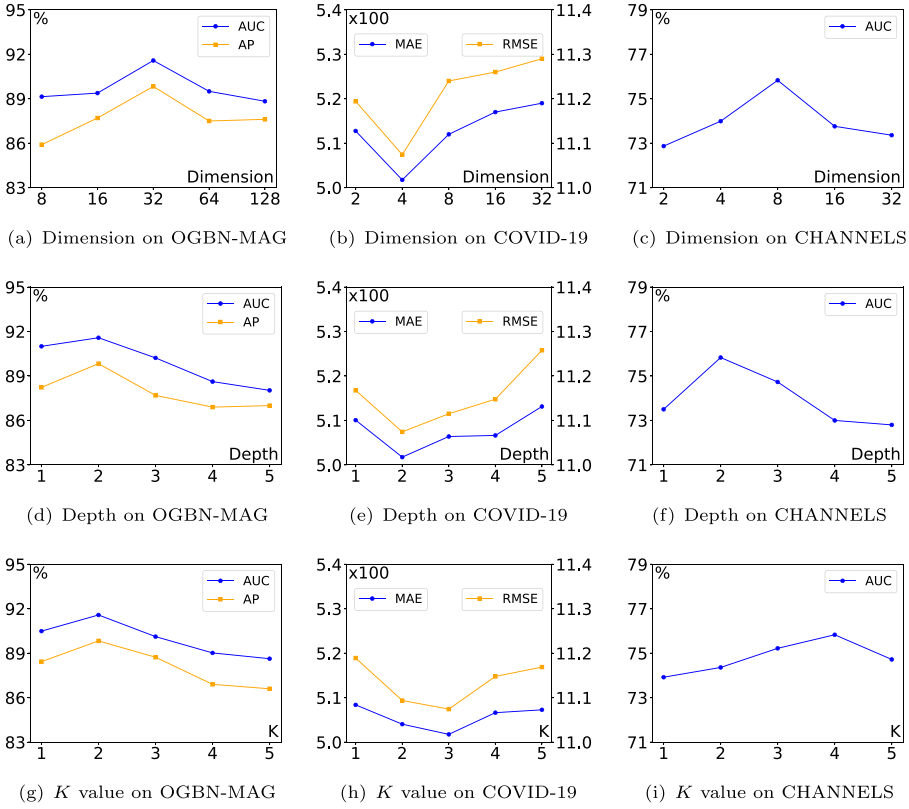
We further analyze the sensitivity of node embedding dimension, network depth (number of aggregation layers), and the value of K on three datasets.

(1) Node embedding dimension: We conduct experiments by increasing the embedding dimensions from 8 to 128 for the OGBN-MAG dataset, from 2 to 32 for the COVID-19 dataset, and from 2 to 32 for the CHANNELs dataset. As shown in the experimental

Table 3 Results of ablation experiments

Experimental group	OGBN-MAG		COVID-19		CHANNELS
	AUC	AP	MAE	RMSE	AUC
MTHG-PA_w/o_het	87.48 ± 0.93	86.32 ± 0.84	599 ± 66	1324 ± 85	73.29 ± 1.21
MTHG	89.57 ± 0.86	87.76 ± 1.35	557 ± 61	1201 ± 81	74.16 ± 1.05
HG-PA	89.01 ± 0.90	86.93 ± 1.23	573 ± 54	1247 ± 76	73.51 ± 1.27
MTHG-PA	$\mathbf{91.58 \pm 0.75}$	$\mathbf{89.83 \pm 1.05}$	$\mathbf{502 \pm 17}$	$\mathbf{1107 \pm 30}$	$\mathbf{75.83 \pm 0.93}$

Boldface represents the results of the best-performing model



(a) Dimension on OGBN-MAG (b) Dimension on COVID-19 (c) Dimension on CHANNELS

(d) Depth on OGBN-MAG (e) Depth on COVID-19 (f) Depth on CHANNELS

(g) K value on OGBN-MAG (h) K value on COVID-19 (i) K value on CHANNELS

Figure 6 Hyperparameter sensitivity results

results in Figure 6a-c, with the increase in node embedding dimensions, the model performance initially improves and then declines. This trend indicates that the increase in dimensions enables the model to capture more nuanced feature information, thereby enhancing its predictive capability. However, excessively large dimensions can lead to overfitting, particularly in the COVID-19 and CHANNEL datasets. This suggests that the optimal choice of embedding dimensions lies in balancing expressive power with generalizability.

(2) Network depth: We conduct experiments by varying the depth of the model from 1 to 5 on three different datasets. The experimental results are shown in Figure 6d-f show that the model performs best when the depth is 2. This is because increasing the depth of a network typically enables more in-depth feature interactions and more complex representations. However, beyond a certain depth, issues such as vanishing or exploding gradients can occur, leading to very slow or unstable parameter updates, which adversely affect model performance. Therefore, in cases where the complexity of the data does not necessitate deeper architectures, it is necessary to adopt a moderate depth to prevent training difficulties and improve convergence.

(3) Value of K: We explore the impact of adjusting the K value from 1 to 5 on the model performance on three datasets. As shown in Figure 6g-i, the results show that the model performance initially increases and then decreases with the increase of K values. From

this phenomenon, it is evident that selecting an appropriate value for K is crucial for the model, as it effectively integrates temporal context information, enriches the embeddings of the nodes, and avoids overwhelming the model with redundant information. This indicates that there exists a critical range for K that can fully leverage the advantages of temporal data without introducing excessive noise or complexity, thereby avoiding overfitting. In applications within dynamic environments, adjusting the K value according to the temporal characteristics of the data can significantly enhance the performance of the model.

6 Conclusion

In this paper, we propose MTHG-PA, a novel method for multi-temporal heterogeneous graph learning with pattern-aware attention, aimed at industrial chain risk detection. MTHG-PA comprises two key methods: the interactive aggregation of multi-temporal heterogeneous information to enrich the embedding representation of node dynamic correlations, and the temporal pattern-aware attention module, which handles long-term information in a non-discrete manner to capture temporal patterns from continuous temporal data. We conduct experiments on three real-world datasets, and the results demonstrate that MTHG-PA outperforms state-of-the-art baselines.

The future work will delve into enhancing the scalability of MTHG-PA for larger and more complex industrial networks, exploring its adaptability to different industries or domains, and investigating its robustness against noisy or incomplete data. Additionally, further research could focus on incorporating additional contextual information, such as external factors that influence the industrial chain, to improve the accuracy and applicability of MTHG-PA in real-world scenarios.

Acknowledgements This work is supported by the National Key R&D Program of China under Grant No. 2022YFB3304300; and the NSFC under Grant No. U23A20297 and 62072087; and the Guangdong Basic and Applied Basic Research Foundation under Grant No. 2023A1515110268; and Postdoctoral Research Foundation of Northeastern University under Grant No. 20230303.

Author Contributions Y. Sun and X. Bi provided the conceptual design of the study, Z. Li wrote the manuscript and completed the experiments, R. Wang, S. Ying, and H. Ji provided supervision for the paper. All the authors reviewed the manuscript.

Funding The authors would like to acknowledge the support provided by the National Key R&D Program of China under Grant No. 2022YFB3304300; and the NSFC under Grant No. U23A20297; and the Guangdong Basic and Applied Basic Research Foundation under Grant No. 2023A1515110268; and Postdoctoral Research Foundation of Northeastern University under Grant No. 20230303.

Data Availability No datasets were generated or analysed during the current study.

Declarations

Competing interests The authors declare no competing interests.

Ethics approval and consent to participate This article does not contain any studies involving human participants and/or animals by any of the authors.

Consent for publication Informed consent was obtained from all individual participants.

References

1. Yang, M., Lim, M.K., Qu, Y., Ni, D., Xiao, Z.: Supply chain risk management with machine learning technology: A literature review and future research directions. Comput. Ind. Eng. **175**, 108859 (2023)
2. Schroeder, M., Lodemann, S.: A systematic investigation of the integration of machine learning into supply chain risk management. Logistics **5**(3), 62 (2021)
3. Kosasih, E.E., Margaroli, F., Gelli, S., Aziz, A., Wildgoose, N., Brintrup, A.: Towards knowledge graph reasoning for supply chain risk management using graph neural networks. Int. J. Prod. Res. 1–17 (2022)
4. Bernstein, F., Song, J.-S., Zheng, X.: Free riding in a multi-channel supply chain. Nav. Res. Logist. (NRL) **56**(8), 745–765 (2009)
5. Yan, R.: Managing channel coordination in a multi-channel manufacturer-retailer supply chain. Ind. Mark. Manag. **40**(4), 636–642 (2011)
6. Ho, W., Zheng, T., Yildiz, H., Talluri, S.: Supply chain risk management: a literature review. Int. J. Prod. Res. **53**(16), 5031–5069 (2015)
7. Song, X., Li, J., Cai, T., Yang, S., Yang, T., Liu, C.: A survey on deep learning based knowledge tracing. Knowl.-Based Syst. **258**, 110036 (2022)
8. Liu, J., Chen, Y., Huang, X., Li, J., Min, G.: Gnn-based long and short term preference modeling for next-location prediction. Inf. Sci. **629**, 1–14 (2023)
9. Xu, C., Zhao, W., Zhao, J., Guan, Z., Song, X., Li, J.: Uncertainty-aware multiview deep learning for internet of things applications. IEEE Trans. Ind. Inform. **19**(2), 1456–1466 (2022)
10. Feng, K., Li, C., Yuan, Y., Wang, G.: Freekd: Free-direction knowledge distillation for graph neural networks. In: Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining, pp. 357–366 (2022)
11. Zhang, C., Chen, J., Shu, T., Tan, J.: Enterprise event risk detection based on supply chain contagion. In: 2022 IEEE 9th International Conference on Data Science and Advanced Analytics (DSAA), pp. 1–10 (2022)
12. Trirat, P., Yoon, S., Lee, J.-G.: Mg-tar: multi-view graph convolutional networks for traffic accident risk prediction. IEEE Trans. Intell. Transp. Syst. **24**(4), 3779–3794 (2023)
13. Bi, W., Xu, B., Sun, X., Wang, Z., Shen, H., Cheng, X.: Company-as-tribe: Company financial risk assessment on tribe-style graph with hierarchical graph neural networks. In: Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining, pp. 2712–2720 (2022)
14. Wang, D., Zhang, Z., Zhou, J., Cui, P., Fang, J., Jia, Q., Fang, Y., Qi, Y.: Temporal-aware graph neural network for credit risk prediction. In: Proceedings of the 2021 SIAM International Conference on Data Mining (SDM), pp. 702–710 (2021)
15. Wang, X., Ji, H., Shi, C., Wang, B., Ye, Y., Cui, P., Yu, P.S.: Heterogeneous graph attention network. In: The World Wide Web Conference, pp. 2022–2032 (2019)
16. Zhang, C., Song, D., Huang, C., Swami, A., Chawla, N.V.: Heterogeneous graph neural network. In: Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, pp. 793–803 (2019)
17. Hu, Z., Dong, Y., Wang, K., Sun, Y.: Heterogeneous graph transformer. In: Proceedings of the Web Conference 2020, pp. 2704–2710 (2020)
18. Schlichtkrull, M., Kipf, T.N., Bloem, P., Van Den Berg, R., Titov, I., Welling, M.: Modeling relational data with graph convolutional networks. In: The Semantic Web: 15th International Conference, ESWC 2018, pp. 593–607 (2018)
19. Sankar, A., Wu, Y., Gou, L., Zhang, W., Yang, H.: 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, pp. 519–527 (2020)
20. Li, H., Li, C., Feng, K., Yuan, Y., Wang, G., Zha, H.: Robust knowledge adaptation for dynamic graph neural networks. IEEE Transactions on Knowledge and Data Engineering (2024)
21. Feng, K., Li, C., Zhang, X., Zhou, J.: Towards open temporal graph neural networks. Preprint arXiv:2303.15015 (2023)
22. Kapoor, A., Ben, X., Liu, L., Perozzi, B., Barnes, M., Blais, M., O'Banion, S.: Examining covid-19 forecasting using spatio-temporal graph neural networks. Preprint arXiv:2007.03113 (2020)
23. Deng, S., Wang, S., Rangwala, H., Wang, L., Ning, Y.: Cola-gnn: Cross-location attention based graph neural networks for long-term ili prediction. In: Proceedings of the 29th ACM International Conference on Information & Knowledge Management, pp. 245–254 (2020)
24. Zheng, Y., Zhang, X., Chen, S., Zhang, X., Yang, X., Wang, D.: When convolutional network meets temporal heterogeneous graphs: an effective community detection method. IEEE Trans. Knowl. Data Eng. **35**(2), 2173–2178 (2021)

25. Xue, H., Yang, L., Jiang, W., Wei, Y., Hu, Y., Lin, Y.: 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, pp. 282–298 (2021)
26. Ji, Y., Jia, T., Fang, Y., Shi, C.: Dynamic heterogeneous graph embedding via heterogeneous hawkes process. In: Machine Learning and Knowledge Discovery in Databases. Research Track: European Conference, ECML PKDD 2021, pp. 388–403 (2021)
27. Liu, X., Miao, C., Fiumara, G., De Meo, P.: Information propagation prediction based on spatial–temporal attention and heterogeneous graph convolutional networks. IEEE Transactions on Computational Social Systems (2023)
28. Babazadeh, R., Razmi, J., Pishvaee, M.S., Rabbani, M.: A sustainable second-generation biodiesel supply chain network design problem under risk. Omega **66**, 258–277 (2017)
29. Khalilabadi, S.M.G., Zegordi, S.H., Nikbakhsh, E.: A multi-stage stochastic programming approach for supply chain risk mitigation via product substitution. Comput. Ind. Eng. **149**, 106786 (2020)
30. Sharma, R., Kamble, S.S., Gunasekaran, A., Kumar, V., Kumar, A.: A systematic literature review on machine learning applications for sustainable agriculture supply chain performance. Comput. Oper. Res. **119**, 104926 (2020)
31. Xu, C., Ji, J., Liu, P.: The station-free sharing bike demand forecasting with a deep learning approach and large-scale datasets. Transp. Res. Part C Emerg. Technol. **95**, 47–60 (2018)
32. Vo, N.N., He, X., Liu, S., Xu, G.: Deep learning for decision making and the optimization of socially responsible investments and portfolio. Decis. Support Syst. **124**, 113097 (2019)
33. Nikolopoulos, K., Punia, S., Schäfers, A., Tsinopoulos, C., Vasilakis, C.: Forecasting and planning during a pandemic: Covid-19 growth rates, supply chain disruptions, and governmental decisions. Eur. J. Oper. Res. **290**(1), 99–115 (2021)
34. Hu, W., Fey, M., Zitnik, M., Dong, Y., Ren, H., Liu, B., Catasta, M., Leskovec, J.: Open graph benchmark: Datasets for machine learning on graphs. Adv. Neural Inf. Process. Syst. **33**, 22118–22133 (2020)
35. Coronavirus Statistics. https://coronavirus.1point3acres.com/en. Accessed 4 Apr 2024
36. Hochreiter, S., Schmidhuber, J.: Long short-term memory. Neural Comput. **9**(8), 1735–1780 (1997)
37. Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A.N., Kaiser, Ł., Polosukhin, I.: Attention is all you need. Adv. Neural Inf. Process. Syst. **30** (2017)
38. Kipf, T.N., Welling, M.: Semi-supervised classification with graph convolutional networks. Preprint arXiv:1609.02907 (2016)
39. Veličković, P., Cucurull, G., Casanova, A., Romero, A., Lio, P., Bengio, Y.: Graph attention networks. arXiv preprint arXiv:1710.10903 (2017)