Multi-Relation Enhanced Dynamic Hypergraph for Session-based Recommendation

HAOYAN FU, ZHIDA QIN, WENHAO XUE, QIXIAN WANG, and XUFENG LIANG, School of Computer Science and Technology, Beijing Institute of Technology, Beijing, China
TIANYU HUANG, School of Computer Science and Technology, Beijing Institute of Technology, Beijing, China
JOHN C. S. LUI, Computer Science & Engineering Department, The Chinese University of Hong Kong, Hong Kong, China

Session-based recommendation (SBR) systems have increasingly focused on hypergraph-based approaches due to their potent capability in capturing high-order item relationships. Typically, existing approaches rely on sequential item relations to manually construct fixed hypergraphs. However, this methodology neglects the multiple relations inherent in the original sequences, thereby impeding the hypergraph's precision in discerning user preferences. Furthermore, the rigidity of fixed hypergraph structures tends to emphasize explicit relationships, ignoring the latent implicit patterns. In light of this, we present a novel Multi-relation enhanced Dynamic HyperGraph (MDHG) learning framework for session-based recommendation, to model intricate and variable item relations. Initially, we establish three distinct relation graphs which capture separate user behavior patterns to extract personalized interest preferences under differentiated intentions. Subsequently, we propose an enhanced dynamic hypergraph paradigm that adaptively generates hypergraph structures based on prior relation graph, thereby reinforcing and unveiling implicit connectivity relations in a layer-aware manner. Finally, to mitigate the noise among diverse relations, we introduce the maximum mutual information auxiliary task and employ the attention mechanism as a cross-relation aggregator. Extensive experiments on various real-world datasets verify the superiority of our MDHG model. Our code is publicly available at https://github.com/Qin-lab-code/MDHG.

CCS Concepts: • **Information systems** → **Recommender systems**;

Additional Key Words and Phrases: Session-based Recommendation, Hypergraph, Graph Neural Network

This work is supported by the National Nature Science Foundation of China under Grant Number U2441242.
Authors' Contact Information: Haoyan Fu, School of Computer Science and Technology, Beijing Institute of Technology, Beijing, China; e-mail: haoyan-fu@bit.edu.cn; Zhida Qin (corresponding author), School of Computer Science and Technology, Beijing Institute of Technology, Beijing, China; e-mail: zanderqin@bit.edu.cn; Wenhao Xue, School of Computer Science and Technology, Beijing Institute of Technology, Beijing, China; e-mail: aqxuewenhao@bit.edu.cn; Qixian Wang, School of Computer Science and Technology, Beijing Institute of Technology, Beijing, China; e-mail: qxianw@bit.edu.cn; Xufeng Liang, School of Computer Science and Technology, Beijing Institute of Technology, Beijing, China; e-mail: xfliang@bit.edu.cn; Tianyu Huang, School of Computer Science and Technology, Beijing Institute of Technology, Beijing, China; e-mail: huangtianyu@bit.edu.cn; John C. S. Lui, Computer Science & Engineering Department, The Chinese University of Hong Kong, Hong Kong, China; e-mail: cslui@cse.cuhk.edu.hk.

ACM 1558-2868/2026/1-ART48
https://doi.org/10.1145/3779422

ACM Reference format:
Haoyan Fu, Zhida Qin, Wenhao Xue, Qixian Wang, Xufeng Liang, Tianyu Huang, and John C. S. Lui. 2026. Multi-Relation Enhanced Dynamic Hypergraph for Session-based Recommendation. *ACM Trans. Inf. Syst.* 44, 2, Article 48 (January 2026), 28 pages.
https://doi.org/10.1145/3779422

1 Introduction

In e-commerce scenarios [8, 35], recommender systems play a critical role in providing personalized product recommendations to individuals. Typically, these systems rely on the user's historical behavior records and user profiles as fundamental components for generating recommendations. However, in certain popular application scenarios, only the user's current activity behavior is available, making the historical behavior data unavailable [12, 38, 43]. This limitation has led to a growing interest in **session-based recommendation (SBR)**, which focuses on predicting the next item to be recommended based on the user's anonymous session information.

Among the various strategies used in SBR, hypergraph has received considerable attention. Existing session-based hypergraph methods [20, 42] fall into two main categories: one class of methods directly models a sequence of sessions as a hypergraph structure. For instance, DHCN [42] establishes hyperedges by linking item nodes within the same session, utilizing hypergraph convolution for learning. However, this approach ignores the multiple explicit relationships inherent in sequence data and is unable to extract direct patterns of user behaviors. Another class of methods artificially constructs different types of hyperedges. For example, HIDE [20] considers hyperedge structures representing transitions, contexts, and intentions, merging them into a unified hypergraph for comprehensive preference learning.

Despite the effectiveness demonstrated by hypergraph approaches above in the domain of SBR, existing schemes exhibit two notable limitations. First, multiple explicit relationships in a session are easily omitted. Consider a scenario in the Tmall dataset with two users that purchase game consoles, as shown in Figure 1. Obviously, the first user decisively buys the game console and its accessories in sequence, and this sequentiality is the most common user behavior pattern. In contrast, the second user exhibits a more deliberative approach, clicking on multiple game consoles before making a final purchase. This introduces a temporal delay in the data that is not adequately reflected in the sequential user pattern. In addition, coincidental user behavior (e.g., the first user purchasing a dress) further corrupts the purely sequential nature of user behavior. Therefore, focusing on sequential user behavior is currently insufficient and a broader range of behavioral patterns need to be considered.

Second, current hypergraph methods construct static and fixed structures using manual rules which are trained as independent views. Nevertheless, the static construction of hypergraphs remains rigid, lacking the ability to dynamically adapt its structure to align with the recommended goal during the training process. This limitation becomes particularly consequential in sparse SBR scenarios, where a significant amount of noise is present. In this context, relying on a static structure is more likely to produce misleading results. Moreover, the independent hypergraph view utilizes implicit higher-order information, leading to a disconnection with explicit multi-relation learning objectives. As a result, the simple combination of hypergraph structures with multiple item relations can interfere with each other and produce sub-optimal recommendation results.

To tackle the aforementioned issues, this article presents a novel model, namely **multi-relation enhanced dynamic hypergraph (MDHG)**. *To tackle the first issue*, three distinct relations—transition, reversed transition, and co-occurrence—are considered to model the sequential, delayed, and coincidental aspects of user behaviors, respectively. In SBR, since information in a single

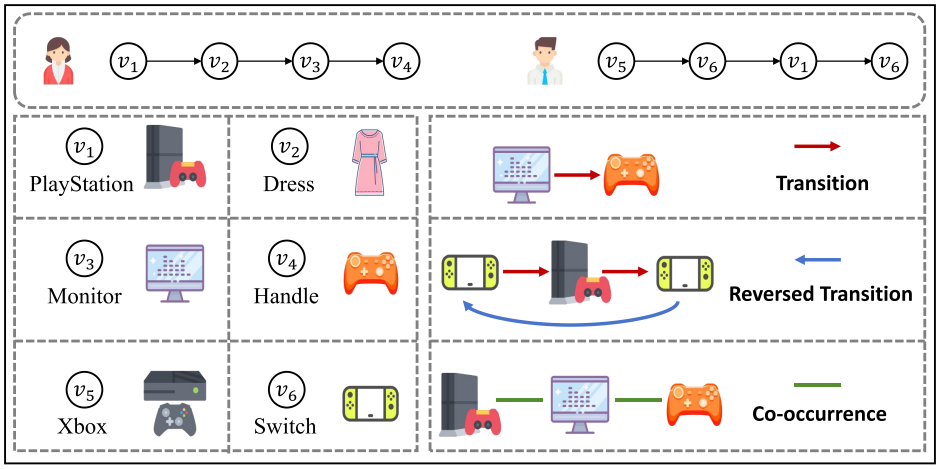


Fig. 1. Three relation examples from the Tmall dataset demonstrate two types of user intent sequences. $v_1 \rightarrow v_2 \rightarrow v_3 \rightarrow v_4$ represents a sequential purchase sequence. $v_5 \rightarrow v_6 \rightarrow v_1 \rightarrow v_6$ reflects a mixed-behavior sequence, where $v_5 \rightarrow v_6 \rightarrow v_1$ represents the browsing behavior, and $v_1 \rightarrow v_6$ represents the transition to the purchase (v_6).

session is constrained, we construct three separate relation graphs to model each of the three relationships to extract user preferences according to specific behavior patterns. Moreover, to reduce the inter-relation noise, we design a self-supervised auxiliary task from the perspective of mutual information. Specifically, we maximize the mutual information between these relations to efficiently exploit user behavior patterns. Then, an attention mechanism is introduced to selectively fuse the representations of different relations to enable accurate recommendations during the recommendation process. *To address the second issue*, an innovative dynamic hypergraph structure is proposed to refine the item relation representations across different graph network layers. Specifically, we dynamically generate the corresponding hypergraph structure based on the item relation representation of each layer and guide the current hypergraph learning direction based on the degree of the relation graph. This approach allows the dynamic hypergraph to capture implicit higher-order relationships within each relation, thereby adaptively improving the corresponding representations. By incorporating this dynamic hypergraph learning process, the model acquires the ability to effectively refine and enhance the item relation representations.

The main contributions of this article are summarized as follows:

—We propose a novel unified learning framework, MDHG, for SBR. Our model integrates both the item relation graphs and dynamic hypergraphs to explore explicit and implicit higher-order relationships holistically.
—We construct three relation graphs to capture distinct explicit relations. To efficiently fuse the item representations learned from the three relations, we design a mutual information maximization scheme to reduce inter-relation noise and achieve the comprehensive representations.
—We design a layer-wise dynamic hypergraph learning mechanism for each relation to deepen the representations. Our mechanism could adaptively refine the representations of the corresponding relation graphs and uncover the implicit relationships.
—Extensive experiments on three real-world datasets show that our model outperforms the state-of-the-art baselines. The ablation experiments and case study results further validate the effectiveness of our model.

2 Related Work

2.1 SBR

SBR systems aim to leverage anonymous session data to enhance the effectiveness of recommendations. Broadly speaking, SBR approaches can be categorized into three main types: traditional, sequential, and graph-based. Traditionally, popular approaches such as **k-nearest neighbors (KNN)**, **Markov chain (MC)**, and collaborative filtering have been utilized, each employing distinct recommendation strategies. Specifically, KNN methods [31] focus on identifying similar sessions to the target session as the foundation for recommendations. MC methods [29, 31] capture the item transition patterns within a session. Collaborative filtering methods [3, 16, 30] model item relationships based on their co-occurrence within a session. While these traditional methods have demonstrated their effectiveness, they often lack the ability to deeply model session data and capture more nuanced representations.

In contrast to traditional methods, sequential approaches [23, 56] in SBR delve deeper into exploring item information within a session, often employing **recurrent neural network (RNN)** models. These models aim to uncover sequential transformation patterns in the data. One notable example is GRU4REC [11], which combines RNN networks with SBR and utilizes **gated recurrent units (GRUs)** to regulate information propagation. Subsequent studies, such as the work by Tan et al. [33], have enhanced model performance by incorporating optimization schemes like data augmentation and modifying data distribution based on RNN. However, RNN-based models assume strict sequential relationships between adjacent items, which can introduce noise and hinder their performance. To address this limitation, attention mechanisms [49] have been conveniently applied in SBR to comprehensively measure the importance of different items within a session. NARM [19] was the first model to integrate attention mechanisms into SBR, leveraging them to compute the user's current interest. Building upon this, STAMP [22] introduced a short-term memory network that utilizes attention to capture the intricate interests within the current session. Nevertheless, the aforementioned sequential approaches predominantly focus on the continuous evolution of items within a session, disregarding higher-order connectivity relationships.

In recent years, graph-based methods [15, 32, 51] have gained popularity in SBR. These methods transform sequential data into graph structures and leverage the topological information encoded in the graph. Graph-based methods have been widely developed in areas such as social recommendation [50] or general recommendation [10]. A noteworthy example of SBR employing graph-based methods is SRGNN [40], which aggregates neighboring nodes and learns sequential relationships on a session graph, thereby establishing itself as a landmark model. Building upon this, the **graph attention model (GAT)**, represented by FGNN [28], was developed. It allows a sensitive computation of the importance of different elements by incorporating attention mechanisms. While previous approaches mainly focused on intra-session relationships, a recent advancement called GCE-GNN [38] exploits global cross-session relationships and integrates global and local representations to enhance recommendations. In more recent developments, some models have shifted their focus toward improving recommendations through auxiliary tasks, such as self-supervised learning [2, 25, 34, 37]. Self-supervised methods maximize the mutual information between positive and negative samples to extract the most important features. Despite the significant results achieved by graph-based approaches in SBR, the use of static graph structures imposes limitations on the exploration space of embedding representations and overlooks many potential higher-order connectivity relationships.

2.2 Hypergraph Learning

Hypergraphs are complex graph structures that can learn higher-order connectivity relationships [1, 21]. While traditional pairwise node relations are limited, the hyperedge of a hypergraph

Table 1. Classification of Hypergraph-based Methods in SBR

Method	Graph Construction	Graph Scope	Representation Learning	Hypergraph Type
DHCN (2021)	Item-Session	Global + Local	HGNN	Static
SHT (2022)	Item-Item	Global + Local	GNN + HGNN	Static
HIDE (2022)	Multi-Relation	Local	HGNN	Static
H3GNN (2023)	Item-Item	Global	GNN + HGNN	Static
BiPNet (2023)	Multi-Relation	Global	HGNN	Static
MHCL (2024)	Item-Item	Global + Local	HGNN	Static
ISHGL (2024)	Item-Item	Global	HGNN	Static
MDHG (ours)	Multi-Relation	Global	GNN + HGNN	Dynamic

can contain several nodes, and the nodes break through the previous binary relations to reach multiplicity. HGNN [6] is the first application of graph convolutional networks on hypergraphs, which laid the foundation for the subsequent development of hypergraphs. In addition, HyperGAT [4] focused on graph attention and applied to hypergraph networks. With the development of hypergraph, it started to be applied in recommendation systems [44, 48, 53]. DHCN [42] ported the classical hypergraph convolution paradigm and used contrast learning to enhance the hypergraph. SHT [41] combined hypergraph with Transformer and proposed a new hypergraph learning method. Different from the above schemes, HIDE [20] constructs different hypergraph structures for different intents and builds a decoupler to learn user interests accurately. Recently, H3GNN [47] applies hypergraph and directed graph aggregators to capture hierarchical information in SBRs. Afterward, BiPNet [52] considers the relations of price, type, and brand of items and applies the hypergraph method. Furthermore, MHCL [9] and ISHGL [5] consider the introduction of self-supervised learning for enhancing item representation and mitigating data sparsity. However, these hypergraph-based SBR methods ignore the complex relational patterns in the session and struggle to adaptively learn implicit relationships. We compare these methods with our model in terms of the dimensions of graph construction, representation learning, and hypergraph type, summarized in Table 1. After comparison, our model considers multiple relations to construct the graph, combining GNN and dynamic HGNN on a global graph range, which is more advanced compared to other methods.

Based on hypergraph learning, dynamic hypergraphs [7, 46] have been widely developed in order to obtain more flexible hypergraph structures. DHGNN [14] first considers a dynamic hypergraph structure, where the hypergraph matrix is automatically updated at each layer of the hypergraph convolution. After that, dynamic hypergraphs gradually became popular in the graph field. HGC-RNN [45] dynamically extends the set of hyperedges by clustering methods such as KNN to adjust the feature representation and hypergraph structure during the embedding process. DyHSL [54] applies dynamic hypergraphs to extract complex spatio-temporal features on the traffic flow forecasting task. Recently, TDHNN [55] further improves the dynamic hypergraphs. It obtains dynamic hyperedge features from the learned distribution and adjusts the number of hyperedges. In the area of recommender systems, DHLCF [39] applies dynamic hypergraphs to collaborative filtering, adaptively modifying the topology of hypergraphs to potentially improve recommendation performance. However, there are multiple complex relationships in SBR, and it is difficult for general dynamic hypergraphs to accurately capture implicit patterns.

3 Preliminaries

The definitions of the key symbols used in this article are shown in Table 2. Formally, let $\mathcal{I} = \{v_1, v_2, \ldots, v_n\}$ denote the set of all items, where $|\mathcal{I}| = n$. We define $s \in S$ to be a single session sequence. For a session sequence, $s = \{v_{s,1}, \ldots, v_{s,t}, \ldots, v_{s,m}\}$, where m is the session length and

Table 2. Notations

Notation	Description
$\mathcal{I} = \{v_1, v_2, \ldots, v_n\}$	The set of items
n	The number of items
S	The set of all sessions
$s = \{v_{s,1}, \ldots, v_{s,t}, \ldots, v_{s,m}\}$	The single session sequence
$v_{s,t}$	The item that users have clicked or interacted from session s at position t
$y_{sv}, \hat{y}_{sv}$	The ground truth and the probability of the next interactive item v for the current session s
$\mathcal{G}_I^{t_1}, \mathcal{V}_I^{t_1}, \mathcal{E}_I^{t_1}$	The transition relation graph and the corresponding set of points and edges
$\mathcal{G}_I^{t_2}, \mathcal{V}_I^{t_2}, \mathcal{E}_I^{t_2}$	The reversed transition relation graph and the corresponding set of points and edges
$\mathcal{G}_I^{c}, \mathcal{V}_I^{c}, \mathcal{E}_I^{c}$	The co-occurrence relation graph and the corresponding set of points and edges
$* \in \{t_1, t_2, c\}$	The symbol of the type of relation graphs
$\mathbf{A}_*, \mathbf{D}_* \in \mathbb{R}^{n \times n}$	The adjacency matrix and degree matrix for the relation graph $*$
$\mathbf{X} \in \mathbb{R}^{n \times d}$	The initial embedding matrix of items
$\mathbf{E}, \hat{\mathbf{E}} \in \mathbb{R}^{n \times d}$	The item embedding before and after relation graphs aggregation
$\hat{\mathbf{H}} \in \mathbb{R}^{K \times K}$	The learned dynamic hypergraph incidence matrix
K	The number of hyperedges
$\mathbf{X}^h \in \mathbb{R}^{n \times d}$	The learned hypergraph embedding of items
$\mathbf{Z}$	The overall session embedding
$\mathbf{W}, \mathbf{a}$	The learnable matrix or vector parameters
α, β	The attention weights
$\mathcal{L}_{main}$	The recommendation task loss
$\mathcal{L}_{auxiliary}$	The auxiliary task loss of maximize mutual information

each $v_{s,t}$ represents the tth item that a user interacted with during the session in chronological order. The task of SBR is to predict the next item $v_{s,m+1}$ that a user is likely to interact with based on the current session s.

In SBR, the probability of the next interactive item v is denoted by $\hat{y}_{sv}$, and the ground truth is expressed as y_{sv} for the current session s. To generate recommendations, candidate items are sorted according to $\hat{y}_{sv}$, and the top-K items are selected as the final recommendation results.

4 Proposed Method

This section presents a framework for capturing different user preferences and learning corresponding embeddings using graph structures. The proposed MDHG is illustrated in Figure 2. We next give a conceptual overview of our approach, discussing in detail the core problem to be solved and the improvements of MDHG over existing methods. After that, we elaborate on the various modules of MDHG. First, we describe the construction of relation graphs, and we demonstrate the graph encoder on these graphs. Second, we propose a method to dynamically generate hypergraphs based on the construction graphs to enhance the representation of propagations. Next, we use a cross-relation aggregation layer to combine representations of different relations. Finally, an auxiliary task is introduced to maximize mutual information between different graphs, which helps to refine their respective representations.

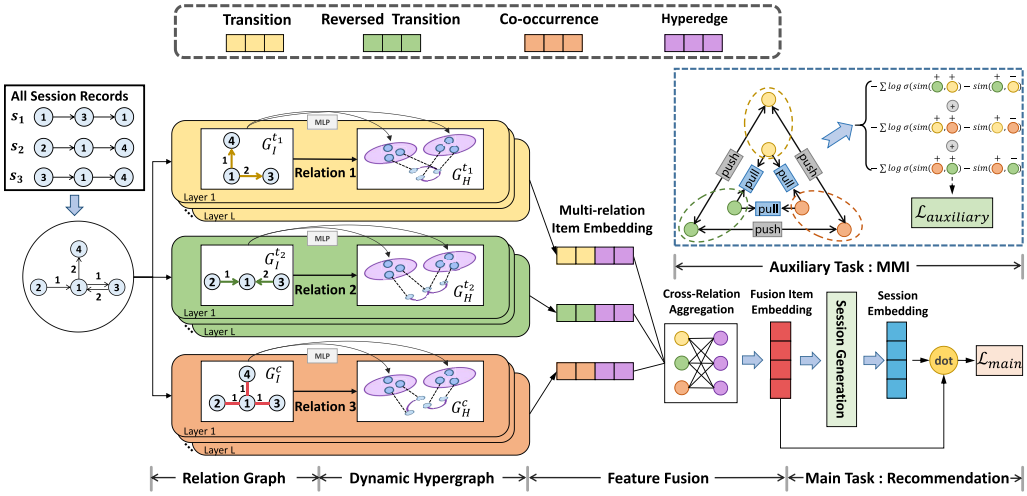


Fig. 2. Overview of the proposed MDHG framework.

4.1 Conceptual Overview

The core goal of this article is to improve the performance of next-item recommendation in SBR. To this end, we propose a key design principle: the "relation graph-dynamic hypergraph" paradigm. After that, we summarize the differences between our model and existing methods.

Existing methods often ignore multiple explicit relations in sessions and rely on static hypergraphs that can't adapt to complex patterns. We propose MDHG, which introduces the multi-relation graphs to capture richer behavioral signals beyond local item-item interactions. Additionally, we develop a layer-wise dynamic hypergraph that first generates hypergraph structures through relation graphs and refines the corresponding item representations at each layer. This design enables more adaptive and comprehensive modeling of user preferences.

Multi-Relation Graph Modeling of User Preferences. In this section, we describe how to build and propagate our model on the item relation graph.

4.1.1 Transition and Reversed Transition Graph. The item transition graph captures the sequential patterns of user preferences across sessions by modeling the transitions between items. This graph can intuitively reflect a user's next purchase intention. In contrast, the reversed transition graph models delay user behavior patterns. Existing works [24] mainly utilize the intra-session sequential transition patterns to capture items' relationships and make predictions. However, due to the limited items and potential noises in a single session, such a method fails to fully consider all item relations and even may produce wrong predictions caused by the noises. To address these limitations, we use all the sessions to construct our graphs and characterize the sequential and delayed patterns among items in a comprehensive view.

Specifically, we first consider two kinds of graphs, the transition graph $\mathcal{G}_I^{t_1}$ and the reversed transition graph $\mathcal{G}_I^{t_2}$. In our constructed graphs, all unique items are considered as nodes, and an edge e_{ij} exists if there is a sequential or reversed order from node i to node j in all sessions. The weight of the edge e_{ij} is determined by the number of times that the relations occur in all sessions, reflecting the importance of more frequent item pairs. Correspondingly, the transition graph only has incoming edges and the reversed transition graph only contains outgoing edges. The incoming edge $e_{ij}^{t_1}$ represents a transition from item v_j to item v_i, while the outgoing edge represents the opposite direction of transition if it exists.

4.1.2 Co-Occurrence Graph. Different with two kinds of transition graph, the co-occurrence graph captures the co-occurrence relation between items in the same session. We use the notion $\mathcal{G}_I^c = (\mathcal{V}_I^c, \mathcal{E}_I^c)$ to denote it. The vertex set $\mathcal{V}_I^c$ is the same as that of the item transition graph, where each unique item is a node. The edge set $\mathcal{E}_I^c$ contains edges that represent the co-occurrence relationship between items in the same session. Specifically, an edge e_{ij} exists in $\mathcal{E}_I^c$ if v_i and v_j co-occur in at least one session. The weight of the edge is determined by the number of sessions where v_i and v_j co-occur. The co-occurrence graph provides a complementary view to consider the semantic relevance of items within the same session in order to understand the contingent behavior of users.

4.1.3 Graph Encoder. As the initialization, we map each unique item ID to an initial embedding matrix $\mathbf{X} \in \mathbb{R}^{n \times d}$, where n represents the number of all item IDs, and d denotes the embedding dimension, which is denoted as:

$$\mathbf{X}^0 = [\mathbf{x}_{v_{i_1}}, \mathbf{x}_{v_{i_2}}, \ldots, \mathbf{x}_{v_{i_n}}]^\top, \tag{1}$$

where $\mathbf{X}_{v_{i_n}}$ is the embedding vector corresponding to the nth item. From the above, it is clear that the three levels of graphs focus on different aspects of information. Therefore, we model the item representation by the same initialization embedding $\mathbf{X}^0$ to refine different types of user preferences based on the shared embedding. Then, we uniformly use a graph propagation scheme to learn the different levels of graphs. For convenience, we use the symbol $* \in \{t_1, t_2, c\}$ as a unifying sign to serve as a consistent paradigm for the three types of relation graphs.

Before aggregation, we first map the initial item embedding of each layer to a feature space by a transformation matrix $\mathbf{W}^l \in \mathbb{R}^{n \times d}$. With this feature matrix, the item representation $\mathbf{E}_*^l$ for each layer can be represented as follows:

$$\mathbf{E}_*^l = \mathbf{W}^l \mathbf{X}_*^{l-1}, \tag{2}$$

where $\mathbf{X}_*^{l-1}$ is the final representation of the previous layer, and l represents the current graph convolution layer. By employing a transformation matrix in each layer, the recommendation system can effectively modify and enhance the representation of features at different relations. This allows for more nuanced and fine-grained information processing, enabling the system to capture and leverage relevant patterns and relationships at varying levels of detail.

To be specific, with the adjacency matrix and degree matrix of graph $\mathcal{G}_I^*$, the representations of neighboring nodes are systematically merged into the feature embedding of the present layer. It is crucial to emphasize that, drawing upon the current layer's item embedding $\mathbf{E}_*^l$, we employ the classical message passing mechanism [36] to effectively assimilate neighbor information. Following the insights from [10], we consider the removal of the nonlinear activation function during the process of graph propagation. We formulate this procedure as follows:

$$\hat{\mathbf{E}}_*^l = \mathbf{D}_*^{-1} \mathbf{A}_* \mathbf{E}_*^l. \tag{3}$$

In this neighborhood aggregation process, $\mathbf{A}_* \in \mathbb{R}^{n \times n}$ is the adjacency matrix from which the connectivity of the current ID to any item can be viewed. And $\mathbf{D}_* \in \mathbb{R}^n$ is the degree matrix, which is used to normalize the adjacency matrix.

4.2 Dynamic Hypergraph Learning

4.2.1 Relation Enhanced Hypergraph Construction. Compared to traditional pairwise graphs, hypergraphs could connect more nodes to a hyperedge and are able to capture higher-order connectivity relationships among items, thereby enabling the exploration of more intricate and complex structures. However, existing works [42] always construct a fixed hypergraph before

the training process. As in real world, the distribution of item data within hypergraphs is highly intricate, such fixed construction mechanism often suffers from this limitation.

To effectively learn implicit higher-order relations, we propose a layer-wise dynamic hypergraph construction mechanism to enhance the node representations. Specifically, we generate a hypergraph structure based on the three aforementioned embeddings $\hat{\mathbf{E}}_*^l$ obtained from the relation graph $\mathcal{G}_I^*$ at each layer of the network. Consequently, the hypergraph $\mathcal{G}_H^*$ emerges as an extended representation of the relation graph, facilitating the augmentation of node representations. To streamline the description and maintain clarity, we opt to omit the explicit mention of the graph type $*$ and the graph network layer l and instead focus on detailing the specific characteristics of the hypergraph structure.

The key of constructing of hypergraphs lies in acquiring the hyperedge distribution $\hat{\mathbf{H}} \in \mathbb{R}^{K \times K}$ derived from the present item representation, where $K \in \{K_{t_1}, K_{t_2}, K_c\}$ is a hyperparameter to indicate the number of hyperedges. To achieve this, we employ a **multi-layer perceptron (MLP)** in conjunction with residual connections to enhance the item representation and subsequently transform it into the hyperedge embedding space. Mathematically, this process can be formalized as follows:

$$\mathbf{H} = \sigma_1(\hat{\mathbf{E}}\mathbf{W}_1 + \hat{\mathbf{E}}), \hat{\mathbf{H}} = \sigma_2(\mathbf{H}\mathbf{W}_2), \tag{4}$$

where $\mathbf{W}_1, \mathbf{W}_2$ are the feature weight matrices, and σ_1, σ_2 are the activation function. In our model, σ_1 is *ReLU* and σ_2 is *Softmax*. We choose $\sigma_1 = ReLU$ as the activation function of the initial layer of MLP to mitigate the issue of gradient disappearance. Simultaneously, we select $\sigma_2 = Softmax$ to update the values in the correlation matrix H and transform them into a probability distribution that reflects the connection relationship between nodes and hyperedges. This ensures that the resulting distribution accurately captures the likelihood of associations between nodes and hyperedges.

Our hypergraph construction approach is conducted on both different types of graphs and different layers of GNN during the learning process. In details, the model generates hypergraphs at each specific layer of the GNN, which produces a dynamic constructions process. We want to emphasize that our model enables the learning of diverse types of higher-order information based on the characteristics of the relation graph. Additionally, the construction of hypergraphs varies across layers to effectively capture information at different levels of granularity, thus accommodating the nuanced representation of the underlying data.

4.2.2 Dynamic Hypergraph Convolutional Network. After constructing the adaptive hypergraph, we will now elaborate on the hypergraph convolution process. Consistent with the standard hypergraph propagation methodology, we adopt the "vertex-hyperedge-vertex" paradigm to effectively aggregate implicit higher-order relations. In this approach, the node representation undergoes a convolution process with the hyperedge representation, facilitated by the incidence matrix $\hat{\mathbf{H}}$. Then, the resulting aggregated information is further integrated into the current vertex representation.

In the conventional vertex-to-hyperedge (or hyperedge-to-vertex) process, the degree matrix of the incidence matrix $\hat{\mathbf{H}}$ is often used for normalization, aligning with the established principles of graph convolution. However, for dynamic hypergraphs, the generation of the incidence matrix varies, and the impact of the degree matrix is contingent upon the refinement of the hypergraph construction. To address this challenge, several dynamic hypergraph models resort to unsupervised methods. These methods learn hypergraph construction as an additional task, which may lead to deviation from the recommended task goals.

Diverging from existing approaches, we present a novel hypergraph convolution method tailored to the characteristics of our dynamic hypergraph structure. In this method, instead of using the degree matrix of the hypergraph itself, we use the degree matrix of the relation graph $\mathcal{G}_I^*$. By

incorporating the degree matrix of the relation graph, we aim to facilitate the convergence of the hypergraph structure toward the relation graph structure. Consequently, our dynamic hypergraph can be viewed as a facilitating subgraph that captures essential aspects of the relation graph. The convolution process is formally defined as follows:

$$\begin{aligned} \mathbf{X}^h &= \hat{\mathbf{H}}\mathbf{D}_I^{-1}\hat{\mathbf{H}}^{\mathrm{T}}\hat{\mathbf{E}}, \\ \mathbf{X} &= \mathbf{X}^h + \hat{\mathbf{E}}, \end{aligned} \tag{5}$$

where $\mathbf{X}$ denotes the final node representation of the current layer and $\mathbf{D}_I$ is the corresponding degree matrix of the relation graph. Through dynamic hypergraph convolution, the item features $\hat{\mathbf{E}}$ of the relation graph are transformed into a hypergraph representation $\mathbf{X}^h$. This continuous "relation graph-dynamic hypergraph" convolution process deepens the unique information of a particular relation, but may result in extreme information loss. Thus, in order to preserve both the deepened relation features and the original information, we perform a residual connection operation after the hypergraph convolution and obtain the final node representation $\mathbf{X}$.

The utilization of the degree matrix of the relation graph offers two notable advantages. First, our dynamic hypergraph serves as a refined representation of the relation graph, and the inclusion of $\mathbf{D}_I$ helps align the degree of each row in the dynamic incidence matrix with its corresponding degree in the relation graph. This alignment ensures that the degree of node activity, i.e., the number of node neighbors, remains similar in both graph structures. As a result, nodes within the hypergraph will prioritize the exploration of higher-order relationships that are not directly linked to the relation graph, instead of focusing solely on their immediate neighbors. Second, the "relation graph-hypergraph" approach we adopt is a more general and practical scheme. Unlike alternative approaches that introduce additional fixed hypergraph structures as auxiliary tasks, our scheme is able to unify the training objectives and consistently optimize toward the recommendation task. This approach enables a more efficient and convenient integration of the relation graph and the hypergraph without introducing extra structures or tasks.

As a result of the above process, $\mathbf{X}_*^l$ is the final representation at lth layer of a certain relation $*$. After learning the L layers of relation graph-hypergraph convolution, we obtain the overall item representation $\hat{\mathbf{X}}_*$, which is formally expressed as:

$$\hat{\mathbf{X}}_* = \frac{1}{L+1}\sum_{l=0}^{L}\mathbf{X}_*^l. \tag{6}$$

In contrast to other functions (e.g., max-pooling, concatenation operations), we opt for the average pooling operation as the method to aggregate item representations from different layers, as it has demonstrated superior performance within our model.

4.3 Cross-Relation Aggregation Layer

The previous graph propagation process leads to the generation of multiple relations of item representations, denoted as $\hat{\mathbf{X}}_*$. These relations contain different types of information on the item. To smoothly integrate these embeddings, we employ a cross-relation aggregation approach that combines the acquired item representations in a cohesive manner. This fusion process is achieved through the utilization of an attention layer, which facilitates the learning of specific information pertaining to the item representations across different relations.

However, the representation after dynamic hypergraph learning highlights specific relation patterns and ignores shallow structural information. Thus, the differences between each relation are too large to converge simultaneously to the optimal through the direct use of the attention mechanism. Therefore, we propose a cross-relation aggregation approach, thus integrating the

three relation representations in a well-balanced way. Specifically, we utilize the embedding $\hat{\mathbf{E}}_*$ after learning from relation graphs to compute the attention scores and fuse the three final relation representations accordingly. Formally, the representation of the cross-relation aggregation can be expressed as follows:

$$\begin{aligned} \alpha_* = Atten(\hat{\mathbf{E}}_*) = \frac{\exp(\mathbf{a}^T \cdot \mathbf{W}_c \hat{\mathbf{E}}_*)}{\sum_* \exp(\mathbf{a}^T \cdot \mathbf{W}_c \hat{\mathbf{E}}_*)}, \\ * \in \{t_1, t_2, c\}, \hat{\mathbf{X}} = \alpha_{t_1} \hat{\mathbf{X}}_{t_1} + \alpha_{t_2} \hat{\mathbf{X}}_{t_2} + \alpha_c \hat{\mathbf{X}}_c, \end{aligned} \tag{7}$$

where $\hat{\mathbf{X}}_*$ is the item representation of each relation, $\mathbf{a} \in \mathbb{R}^d, \mathbf{W}_c \in \mathbb{R}^{d \times d}$ are the matrix of learnable parameters, and $\hat{\mathbf{X}}$ is the combined representation of all relations.

4.4 Representation Enhancement with Maximize Mutual Information

Multi-relation graphs facilitate the exploration of diverse user interests and enable a deeper understanding of the information within each relation under the dynamic hypergraph. However, due to the independent nature of the different relations, they often exhibit significant noise and lack coherence. Such noise becomes more pronounced as the dynamic hypergraph increases the depth of graph convolution in each layer. Although we have used the attention mechanism to fuse the information from three relations, this may not be sufficient to mitigate the effects of severe noise.

Inspired by MHCN [48], we exploit the mutual information mechanism to mitigate the variability between different relations. By maximizing the mutual information across relations, we aim to improve the coherence and quality of the learned representations, which ultimately improves the effectiveness of the recommendation system.

We model the representation $\mathbf{X}_k$ from the kth relation as:

$$\mathbf{X}_k = \mathcal{F}_k(P, N_k), \tag{8}$$

where P is true shared preference and N_k is relation-specific noise (N_1, N_2, N_3 mutually independent). To suppress N_k and enhance P, we maximize mutual information between relation representations:

$$\max \sum_{i \neq j} I(\mathbf{X}_i; \mathbf{X}_j). \tag{9}$$

This objective forces representations to agree on shared preference P while discarding independent noise N_k because:

$$I(\mathbf{X}_i; \mathbf{X}_j) = I(\mathcal{F}_i(P, N_i); \mathcal{F}_j(P, N_j)) \leq H(P), \tag{10}$$

equality holds iff $N_i = N_j = 0$. The upper bound $H(P)$ is achieved only when noise vanishes.

Previous work [18] has demonstrated that pairwise ranking loss can be used to estimate lower bounds on mutual information. Thus, the lower bound on the mutual information of any two relations can be estimated as:

$$\mathbb{E}\left[\log \sigma(\underbrace{\text{sim}(\mathbf{X}_i^+, \mathbf{X}_j^+)}_{\text{positive pair}} - \underbrace{\text{sim}(\mathbf{X}_i^+, \mathbf{X}_j^-)}_{\text{negative pair}})\right] \leq I(\mathbf{X}_i; \mathbf{X}_j), \tag{11}$$

where *sim* is the metric function for calculating the consistency of two vectors and σ is the *sigmod* activation function. The above expectation is taken to be negative and denoted as loss L_s. With the above equation, minimizing L_s is equivalent to maximizing $I(\mathbf{X}_i; \mathbf{X}_j)$. At this point, $\mathbf{X}_i$ and $\mathbf{X}_j$ are

forced to encode a shared preference P and the noise N_i and N_j are required to be reduced. Thus, maximizing the mutual information of the two relations mitigates relation-specific noise.

To evaluate the mutual information between relations t_1 and t_2, we employ a specific approach involving positive and negative samples. Within each batch, we select the item embeddings $\hat{\mathbf{X}}_{t_1}^{h,+}$ and $\hat{\mathbf{X}}_{t_2}^{h,+}$ through a lookup table operation. These selected embeddings serve as positive samples for each other and are considered as the ground truth. Subsequently, we generate negative samples $\hat{\mathbf{X}}_{t_2}^{h,-}$ by randomly perturbing the representations of relation t_2 based on the corresponding rows relative to the positive samples. Following the above derivation of the maximization of mutual information, the loss of relations t_1 and t_2 can be expressed as:

$$\mathcal{L}_{s_1} = -\sum_{\mathcal{I}} \log \sigma(sim(\mathbf{X}_{t_1}^{h,+}, \hat{\mathbf{X}}_{t_2}^{h,+}) - sim(\hat{\mathbf{X}}_{t_1}^{h,+}, \hat{\mathbf{X}}_{t_2}^{h,-})), \tag{12}$$

For simplicity, we use the dot product here. Such a loss function optimization reduces the noise between the two relations and achieves better fusion performance.

It is worth noting that we choose the embedding $\mathbf{X}_*^h$ after the dynamic hypergraph rather than the embedding $\hat{\mathbf{X}}_*$ after residual connections. The residual connection operation introduces raw information and weakens the relation-specific patterns. In contrast, the representation $\mathbf{X}_*^h$ is more capable of representing unique relations. In this way, there is greater variation between different relations in the computation of mutual information.

Similarly, relation t_1/t_2 and relation c are computed to obtain the loss functions $\mathcal{L}_{s_2}$, $\mathcal{L}_{s_3}$, respectively. Afterward, we unify the three loss functions as our auxiliary task objectives:

$$\mathcal{L}_{auxiliary} = \mathcal{L}_{s_1} + \mathcal{L}_{s_2} + \mathcal{L}_{s_3}. \tag{13}$$

4.5 Session Generation

After obtaining the final representation of the items, we will compute the overall representation for each session. It is worth noting that the session representation does not only depend on all the internal items but also involves the sequentiality between these items. That is, all the items contribute to the session overall, but the items interacted later in the session are more important to the user's preference. Therefore, we combine overall session information and location information to make better recommendations.

First we combine all items within the session to give overall session information. Given a current session $s = \{v_{s,1}, \ldots, v_{s,t}, \ldots, v_{s,m}\}$, where the corresponding representation of tth position is $\hat{\mathbf{X}}_{v_{s,t}}$. To learn the information of each item inside the session in a balanced way, we apply the average pooling operation to represent the session information:

$$\mathbf{z}^{'} = \frac{1}{m}\sum_{t=1}^{m} \hat{\mathbf{X}}_{v_{s,t}}. \tag{14}$$

Next, inspired by GCE-GNN [38], we learn the location information based on the order of interaction of the session's items. We set the position matrix $\mathbf{p} = [\mathbf{p}_1, \mathbf{p}_2, \ldots, \mathbf{p}_m]$ to learn the corresponding location information inside the session. Similarly, the reversed position embedding $\mathbf{p}_{m-t+1}$ is combined with the vector of items at the corresponding location t:

$$\mathbf{z}_t = tanh([\hat{\mathbf{X}}_{v_{s,t}} \| \mathbf{p}_{m-t+1}]\mathbf{W}_p), \tag{15}$$

where $tanh$ is the activation function, $\mathbf{W}_p \in \mathbb{R}^{2d\times d}$ is the learnable weight matrix, and $\|$ is the concatenation operation.

Then, through a soft attention mechanism, we compute the weights of each item within the session and combine them to obtain the overall session representation:

$$\beta_t = \mathbf{W}_a \sigma(\mathbf{W}_b \mathbf{z}' + \mathbf{W}_c \mathbf{z}_t + \mathbf{b}), \quad \mathbf{Z} = \sum_{t=1}^{m} \beta_t \hat{\mathbf{X}}_{v_{s,t}}, \tag{16}$$

where $\mathbf{W}_a, \mathbf{b} \in \mathbb{R}^d$ and $\mathbf{W}_b, \mathbf{W}_c \in \mathbb{R}^{d \times d}$ are trainable parameters and σ (here we set to *sigmod*) is the activation function.

4.6 Model Optimization

After obtaining the representation of the current session s, we calculate the recommendation score for each candidate item $v_i \in \mathcal{I}$. Concretely, we use the dot product and apply the $softmax$ function:

$$\hat{y}_{si} = Softmax(\mathbf{Z}_s \hat{\mathbf{X}}_{v_i}). \tag{17}$$

To optimize our recommendation main task, we adopt a cross-entropy loss function for each session s, defined formally as follows:

$$\mathcal{L}_{main} = -\sum_{i=1}^{m} y_{si} \log(\hat{y}_{si}) + (1 - y_{si}) \log(1 - \hat{y}_{si}), \tag{18}$$

where y_{si} is the ground truth of the candidate item.

Finally, we uniformly optimize the recommended main task and the self-supervised auxiliary task and compute the total loss function as follows:

$$\mathcal{L}_{total} = \mathcal{L}_{main} + \lambda \mathcal{L}_{auxiliary}, \tag{19}$$

where λ is a hyperparameter for control of the ratio between two tasks. The entire training procedure of our MDHG model is presented in Algorithm 1.

5 Experiments

The experiments aim to address several key **research questions (RQs)** and evaluate the performance of the models in different scenarios:

- —*RQ1*: Does MDHG perform better than benchmark models on real-world datasets?
- —*RQ2*: Are the different components and the different relations valid?
- —*RQ3*: Is the dynamic hypergraph structure really improving the representation learning?
- —*RQ4*: How do different relations actually affect our model?
- —*RQ5*: How do different settings (layer numbers, hyperedge numbers) impact the effectiveness of our models?
- —*RQ6*: How efficient is our model compared to other baselines?
- —*RQ7*: Is our approach well illustrated in real cases?

5.1 Experimental Settings

5.1.1 Datasets. We employ three real-world datasets, namely Tmall,[1] RetailRocket,[2] and Amazon,[3] for our study. The Tmall dataset represents an e-commerce scenario and consists of a 6-month record of purchase activities. On the other hand, the RetailRocket dataset captures browsing history over the same duration. In addition, the Amazon dataset records user reviews of items. The statistics of them are shown in Table 3.

[1] https://tianchi.aliyun.com/dataset/dataDetail?dataId=42.
[2] https://www.kaggle.com/retailrocket/ecommerce-dataset.
[3] http://jmcauley.ucsd.edu/data/amazon/.

Algorithm 1: The Training Procedure of MDHG

Input: The session sequence $\int$, the initial embedding $\mathbf{X}$;
Output: Optimized parameters Θ.

Initialization parameters;
Construct the transition graph $\mathcal{G}_I^{t_1}$, the reversed transition graph $\mathcal{G}_I^{t_2}$ and the co-occurrence graph $\mathcal{G}_I^c$;
for each iteration **do**
  for each batch **do**
    for each Layer **do**
      Learn item representations $\hat{\mathbf{E}}_*^l$ for all relations through Equation (1-3);
      Calculate the dynamic hypergraph incidence matrix $\hat{\mathbf{H}}$ through the item representation $\hat{\mathbf{E}}_*^l$ through Equation (4);
      The item representation $\hat{\mathbf{E}}_*^l$ is deepened by the learned dynamic hypergraph structure $\hat{\mathbf{H}}$ into a representation $\hat{\mathbf{X}}_*^l$ through Equation (5);
    end for
    The final item representation $\hat{\mathbf{X}}_*$ is computed by aggregating multiple layers of item representations with Equation (6);
    Three relation representations are combined by cross-relation aggregation layer with Equation (7);
    The auxiliary loss of the maximize mutual information task is calculated through Equation (12-13);
    Calculate the overall session representation with Equation (14-16);
    Calculate the total loss with Equation (17-19);
  end for
end for
return optimized model parameters Θ.

Table 3. Datasets

Dataset	# Train	# Test	# Items	Average Lengths
Tmall	3,51,268	25,898	40,728	6.69
RetailRocket	4,33,643	15,132	36,968	5.43
Amazon	61,137	41,714	18,888	6.66

Following the SRGNN [40] to data partitioning, we divide the train/test set by timestamps. For RetailRocket and Amazon datasets, sessions before 7 days will be divided into the training set and sessions within 7 days will be divided into the test set. And due to the short time period of Tmall dataset, the timestamp checkpoint for division is set to 100 seconds.

To ensure a fair comparison of different baselines, we follow a standard data processing method similar to the approach outlined in [40]. Specifically, we apply filtering criteria to remove sessions with less than two items and items with less than five occurrences. We also add additional sequence data to segment each session and perform sequence augmentation. Furthermore, in order to reflect the effect of our model in the cold-start scenario, we refer to [15] to make predictions about items with no interactions on the Amazon dataset.

5.1.2 Metrics. The evaluation of SBR models commonly employs several standard metrics, as highlighted in relevant studies such as [13, 53]. Among these metrics, Recall@K, MRR@K,

and NDCG@K are widely recognized as suitable measures for assessing the performance of SBR algorithms.

5.1.3 Baselines. In our evaluation, we compare our algorithm against several state-of-the-art algorithms in the field of SBR. These algorithms include:

— *FPMC*[4] [29] is a sequential recommendation model that combines MCs and matrix factorization techniques to capture sequential patterns in user behavior.
— *GRU4REC*[5] [11] employs an RNN with GRUs to process session data, leveraging the GRU mechanism to model sequential dependencies.
— *NARM*[6] [19] builds upon the RNN structure and incorporates a global representation obtained from the final output of the RNN, as well as a local representation derived from the weights of the overall output. These representations are combined to generate predictions.
— *SASRec*[7] [17] utilizes attention mechanism to model user historical behaviors, inheriting the simplicity of MC-based methods while maintaining the performance of RNN-based methods.
— *CORE*[8] [12] has designed a novel representation consistency encoder to unify the representation space of encoding and decoding processes, making recommendations more accurate.
— *DuoRec*[9] [27] proposes a data augmentation strategy based on dropout, which solves the problem of traditional contrastive learning data augmentation methods being difficult to provide data consistency enhancing samples.
— *MCLRec*[10] [26] utilizes meta-learning methods to update the model augmenters, in order to improve the quality of contrast pairs without increasing the amount of data.
— *SRGNN*[11] [40] transforms the sequential recommendation problem into a graph problem. It utilizes GNNs to learn latent vector representations of items and employs soft attention mechanisms to obtain final user representations.
— *GCE-GNN*[12] [38] considers additional session information and utilizes attention mechanisms to fuse two levels of item representations, enhancing the recommendation performance.
— *RESTC*[13] [34] considers both temporal representations and global collaborative relationships and introduces contrastive learning to match temporal information in the neighborhood of the global graph.
— *DHCN*[14] [42] utilizes hypergraphs to capture higher-order relationships among items within sessions, enabling the model to capture more complex patterns in user behavior.
— *HIDE*[15] [20] constructs hypergraphs by decoupling user intent and employs an auxiliary task of intent prediction to enhance the independence of individual intents, leading to improved recommendation performance.
— *H3GNN*[16] [47] proposes a hierarchical hypergraph structure to capture hierarchical information and incorporates a directed graph aggregator to aggregate sequential information.

[4] https://github.com/stathwang/FPMC.
[5] https://github.com/hidasib/GRU4Rec?tab=readme-ov-file.
[6] https://github.com/Wang-Shuo/Neural-Attentive-Session-Based-Recommendation-PyTorch.
[7] https://github.com/kang205/SASRec.
[8] https://github.com/RUCAIBox/CORE.
[9] https://github.com/RuihongQiu/DuoRec.
[10] https://github.com/QinHsiu/MCLRec.
[11] https://github.com/CRIPAC-DIG/SR-GNN.
[12] https://github.com/CCIIPLab/GCE-GNN.
[13] https://github.com/SUSTechBruce/RESTC-Source-code.
[14] https://github.com/xiaxin1998/DHCN.
[15] https://github.com/yf-li15/HIDE.
[16] https://github.com/Williamy946/H3GNN.

— *ISHGL*[17] [5] proposes an intention-enhanced self-supervised hypergraph learning model to learn higher-order information from the global hypergraph. And it designs a contrastive learning method that does not require additional data enhancement to improve the performance.

5.1.4 Parameter Setup. In alignment with prior studies, we establish the hyperparameters for our experiments. Specifically, we employ the Adam optimizer with a maximum of 20 training epochs. During the training phase, we set the learning rate to 0.0002 for Tmall and 0.0005 for RetailRocket and Amazon to ensure effective optimization. For all datasets, we configure the network architecture with two network layers, the embedding size of 100, and a batch size of 100. For the number of hyperedges, we conduct hyperparameter analysis and determine the optimal settings for each dataset: 40 for Tmall, 60 for RetailRocket, and 80 for Amazon. Additionally, we introduce an auxiliary task coefficient, lambda, which we set to 0.001 to strike a balance between the primary and auxiliary tasks. To ensure fair comparisons with the existing baseline models, we adhere to the parameter settings outlined in their respective original papers.

5.2 Overall Comparison (RQ1)

We present the experimental results in Table 4, highlighting the best performance in each row. Our model outperforms all baseline models across three well-established SBR datasets in terms of Recall@K, MRR@K, and NDCG@K under top-10 and top-20 settings, demonstrating the effectiveness of our proposed approach.

Traditional and RNN recommendation methods perform relatively poorly. FPMC fails to adequately consider the importance of temporal relationships in SBR, resulting in sub-optimal outcomes. And with the introduction of temporal information, RNN methods such as NARM and GRU4Rec improve performance.

All sequential approaches on the first two datasets performed much worse than the graph and hypergraph strategies, demonstrating the potential of graph-based approaches for the exploitation of higher-order information in SBR. On the other hand, in the cold-start scenario of the Amazon dataset, there is a lack of higher-order information to provide to the graph model, so the sequential model performs better. Our model, on the other hand, not only considers multiple relations but also learns implicitly through dynamic hypergraphs, which allows it to overcome the limitation of cold start.

The shift to graph-based convolutional methods, starting with SRGNN, reveals their superiority over RNN-based and sequential approaches across all metrics. Graph neural networks demonstrate their potential for achieving superior results in SBR. GCE-GNN proves particularly effective by leveraging both global and local information, showcasing the benefits of utilizing multiple sources of information. RESTC considers the introduction of temporal information on top of global collaborative information to achieve performance improvements.

Hypergraph-based methods exhibit even stronger performance compared to general graph structures. DHCN directly applies hypergraph convolution methods to SBR, while HIDE employs multiple intents to construct hypergraphs and employs decoupled learning. In addition, H3GNN and ISHGL only consider relationships between items, ignoring the complex relation patterns within a session, and thus exhibit sub-optimal performance. These models achieve impressive results across multiple datasets. However, our model surpasses these benchmarks in all metrics. By implementing dynamic hypergraphs, we adaptively capture higher-order information and learn a comprehensive representation. It is worth mentioning that the HIDE model itself exhibits an overly complex structure, leading to out-of-memory issues for RetailRocket dataset on our RTX 3090 GPU.

[17] https://github.com/YanYanYanYuYao/ISHGL.

Table 4. Performance Comparison of All Methods on Three Datasets

Method	Tmall			RetailRocket			Amazon		
	R@10	M@10	N@10	R@10	M@10	N@10	R@10	M@10	N@10
FPMC	12.88	7.06	8.46	24.88	11.28	15.62	0.28	0.13	0.16
GRU4Rec	11.95	6.14	7.50	34.79	18.74	22.46	0.69	0.22	0.33
NARM	11.36	5.79	7.10	34.64	18.57	22.32	0.51	0.18	0.16
SASRec	11.68	5.78	7.18	32.59	17.07	20.72	2.22	0.82	1.50
CORE	16.05	5.95	8.36	32.97	12.45	17.37	1.05	0.46	0.69
DuoRec	21.60	12.59	14.73	43.62	24.32	28.90	2.16	0.83	1.00
MCLRec	21.66	12.63	14.77	43.45	24.32	28.85	2.01	0.83	1.11
SRGNN	23.19	12.97	15.40	44.41	26.66	30.90	0.89	0.31	0.44
GCE-GNN	27.40	14.73	17.77	48.07	28.10	32.90	1.28	0.50	0.68
RESTC	28.31	14.96	18.14	46.42	27.88	32.35	1.34	0.64	0.83
DHCN	28.65	16.26	19.14	48.10	28.23	32.92	1.97	0.69	0.98
HIDE	30.30	16.57	19.96	-	-	-	0.71	0.31	0.40
H3GNN	23.01	13.89	16.05	28.55	15.84	19.06	0.88	0.49	0.59
ISHGL	23.90	13.19	15.72	44.60	26.93	31.10	1.15	0.52	0.69
MDHG	**32.63**	**19.45**	**22.40**	**48.67**	**29.64**	**34.17**	**2.33**	**0.88**	**1.60**
Improve.	7.7%	17.4%	12.2%	1.2%	5.0%	3.8%	5.0%	6.0%	6.7%
p-Val.	$3e^{-3}$	$1.3e^{-2}$	$1e^{-3}$	$1.4e^{-2}$	$2.4e^{-2}$	$2.7e^{-2}$	$1.5e^{-2}$	$1.5e^{-2}$	$2.3e^{-2}$

Method	Tmall			RetailRocket			Amazon		
	R@20	M@20	N@20	R@20	M@20	N@20	R@20	M@20	N@20
FPMC	15.50	7.26	8.78	30.49	12.25	16.74	0.40	0.14	0.19
GRU4Rec	15.23	6.36	8.29	41.70	19.26	24.36	1.29	0.26	0.48
NARM	14.57	6.01	7.91	42.03	19.04	24.06	0.91	0.21	0.19
SASRec	14.70	5.99	7.95	39.27	17.56	22.48	3.19	0.88	1.77
CORE	20.77	6.29	9.55	40.79	12.96	19.25	1.38	0.48	0.79
DuoRec	26.06	12.90	15.85	52.37	24.93	31.11	3.14	0.95	1.43
MCLRec	25.97	12.93	15.86	51.76	24.91	30.91	3.13	0.91	1.39
SRGNN	27.50	13.27	16.49	51.63	27.13	32.76	1.53	0.35	0.60
GCE-GNN	32.43	15.08	19.04	55.73	28.63	34.84	1.99	0.55	0.86
RESTC	33.73	15.34	19.52	53.53	28.37	34.15	3.04	0.71	1.14
DHCN	34.39	16.66	20.62	54.82	28.74	34.79	3.15	0.77	1.28
HIDE	36.28	16.99	20.87	-	-	-	1.08	0.33	0.50
H3GNN	27.57	14.20	17.20	35.27	16.31	20.67	1.06	0.50	0.64
ISHGL	29.11	13.55	17.03	52.22	27.45	32.98	1.83	0.61	0.72
MDHG	**38.79**	**19.64**	**23.61**	**57.05**	**30.17**	**36.21**	**3.87**	**1.10**	**2.15**
Improve.	6.9%	15.6%	13.1%	2.4%	5.0%	4.1%	21.3%	15.8%	21.5%
p-Val.	$3e^{-3}$	$1e^{-3}$	$2e^{-3}$	$4.6e^{-2}$	$1.3e^{-2}$	$1.1e^{-2}$	$1.9e^{-2}$	$5e^{-3}$	$7e^{-3}$

For each dataset and metric, the optimal model is highlighted in bold, and the second-best is underlined. We report statistically significant improvements of our MDHG model over the second-best, determined by a t-test ($p < 0.05$).

5.3 Ablation Study of Different Modules (RQ2)

In this subsection, we compare the effects of different variants on our MDHG model to verify the effectiveness of our core modules. Specifically, we remove the critical components of the model separately to observe the performance of the model.

— *w/o Relation*: Removes the three item relation graphs and keeps only the dynamic hypergraph structure.

Table 5. Ablation Study

Variants	Tmall		RetailRocket		Amazon	
	Recall@20	**MRR@20**	**Recall@20**	**MRR@20**	**Recall@20**	**MRR@20**
w/o Relation	32.59	16.84	56.34	29.75	1.81	0.43
w/o Hyper	36.37	17.47	56.78	29.08	2.37	0.58
w/o MMI	38.28	18.99	56.71	29.61	2.94	0.75
w/o Cross	38.47	19.35	56.89	29.47	2.09	0.72
w/o Rel-Deg	37.74	18.95	55.96	29.73	2.42	0.64
MDHG	**38.79**	**19.64**	**57.05**	**30.17**	**3.87**	**1.10**

The best results are in bold.

— *w/o Hyper*: Removes the entire part of the hypergraph, keeping only the three relation graphs for training.
— *w/o MMI*: Removes the mutual information maximization task.
— *w/o Cross*: Changes the attention fusion mechanism to a simple sum pooling.
— *w/o Rel-Deg*: Replaces our degree matrix of the relation graph with the degree matrix of the hypergraph itself.

The results of the ablation experiments are shown in Table 5. Based on the experimental results, all five variants exhibit varying degrees of performance degradation. Notably, *w/o Relation*, *w/o Hyper*, and *w/o Rel-Deg* are relatively more pronounced, which highlights the key role played by the multi-relation graph and dynamic hypergraph components. Furthermore, when comparing the RetailRocket dataset to the Tmall and Amazon datasets, the performance degradation of the *w/o Relation* and *w/o Hyper* is less significant. And from Table 5, we can see that the average session lengths of the Tmall and Amazon datasets are 6.69 and 6.66, respectively, while that of the RetailRocket dataset is 5.43. This indicates that the performance of the model with shorter session lengths is more determined by the in-session aggregation results and cannot fully exploit the ability of our multi-relation and dynamic hypergraph modules to capture higher-order information across sessions.

Conversely, both the *w/o MMI* and *w/o Cross* exhibit diminished expressive power across the different relations, resulting in a negative impact on all performance metrics. The *w/o MMI* promotes greater independence between the relations, which, in turn, leads to increased vulnerability to noise. On the other hand, the *w/o Cross* employs a sum pooling operation to directly aggregate the representations of all relations, neglecting the selective representation capabilities of each relation.

These findings highlight the importance of the relation graphs and dynamic hypergraph component and the need for careful consideration when integrating different relation representations. The results emphasize the significance of preserving the interactions between explicit and implicit relations and optimizing the selective representation of diverse relations to achieve improved recommendation performance.

5.4 Ablation Study of Different Relations (RQ2)

One of the main contribution points of our MDHG model is the combination of three relations. Although the previous subsection has demonstrated the validity of the individual modules of our model, there is still a need to explore the impact of the three different relations. Specifically, we build separate models with only a single relation and train them through our "relation graph-dynamic hypergraph" paradigm. The results are shown in Table 6, where the three relations are transition, reversed transition, and co-occurrence.

Table 6. Ablation Study of Different Relations

Variants	Tmall		RetailRocket		Amazon	
	MRR@20	NDCG@20	MRR@20	NDCG@20	MRR@20	NDCG@20
Multi-relation	**19.64**	**23.61**	**30.17**	36.21	**1.10**	**2.15**
Transition	18.16 (↓8.1%)	22.34 (↓5.7%)	29.66 (↓1.7%)	35.90 (↓0.8%)	1.04 (↓5.8%)	1.98 (↓8.6%)
Reversed transition	18.23 (↓7.7%)	22.41 (↓5.4%)	29.75 (↓1.4%)	35.96 (↓0.7%)	0.97 (↓13.4%)	1.91 (↓12.6%)
Co-occurrence	18.51 (↓6.1%)	22.75 (↓3.8%)	29.92 (↓0.8%)	**36.33**	1.02 (↓7.8%)	1.91 (↓12.6%)

The best results are in bold.

The results of the relation ablation experiments show that the fused multi-relation model outperforms the single-relation model on all datasets. Among them, the performance degradation of the single-relation model is most obvious for the Tmall and Amazon datasets. As shown in Table 3, the RetailRocket dataset has the shortest average length, which may introduce more severe noise and affect the fusion effect of the multi-relation model. Overall, multi-relation fusion can effectively combine beneficial information from different relationships and achieve better performance.

5.5 The Effect of Dynamic Hypergraph for Three Relations (RQ3)

In this subsection, in order to investigate the effectiveness of our "relation graph-dynamic hypergraph" paradigm, we investigate the impact of different graph encoder variants on model performance. Specifically, we construct six variants for three relations: Transition Graph Conv, Reversed Transition Graph Conv, Co-occurrence Graph Conv, Transition Graph Hyper Conv, Reversed Transition Graph Hyper Conv, and Co-occurrence Graph Hyper Conv. The first three variants use only relation graph convolution without dynamic hypergraph learning for the three relations, while the last three variants replace dynamic hypergraphs with static hypergraphs for the three relations. Then we train these variants uniformly on all three datasets and the results are shown in Figure 3.

We can observe that: (1) Removing the dynamic hypergraph structure makes the model performance drop significantly. The original relation graph is only able to perceive neighbors within K hops, which leads to limited relation patterns. (2) Replacing the dynamic hypergraph with a static hypergraph leads to a performance degradation. The improved performance of the static hypergraph compared to the relation graph suggests that hypergraphs are effective in deepening the relation graph. However, the static hypergraph fixes the nodes connected by hyperedges, which limits the adaptation of the original relation graph to the hypergraph. (3) The performance of the same type of variants of the three relations is very close. It suggests that all three relations are equally important and all can capture unique relation patterns under our dynamic hypergraph paradigm. The above observations greatly demonstrate the effectiveness of our dynamic hypergraph structure.

5.6 The Effect of Different Relations in Practical Recommendations (RQ4)

To further validate the effect of different relation patterns on our model, we save the trained model in Section 5.4 for the Tmall dataset and analyze the actual recommendation list.

For a selected session (the item IDs are {18333, 36251, 36251, 32381, 36251, 36292, 36292, 36251, 18333, 26901, 18333}), we give the specific list of recommendations of four models as shown in Figure 4. Where lighter colored items are recommended with higher rankings and scores. The red boxes represent target recommended items (item ID 26900). In the three single-relationship models, both the transition relation and the reversed transition relation are difficult to capture this target item, while it can be accurately recommended by the co-occurrence relation. After the fusion of the three relations, the multi-relation can effectively combine the information of the three relations and

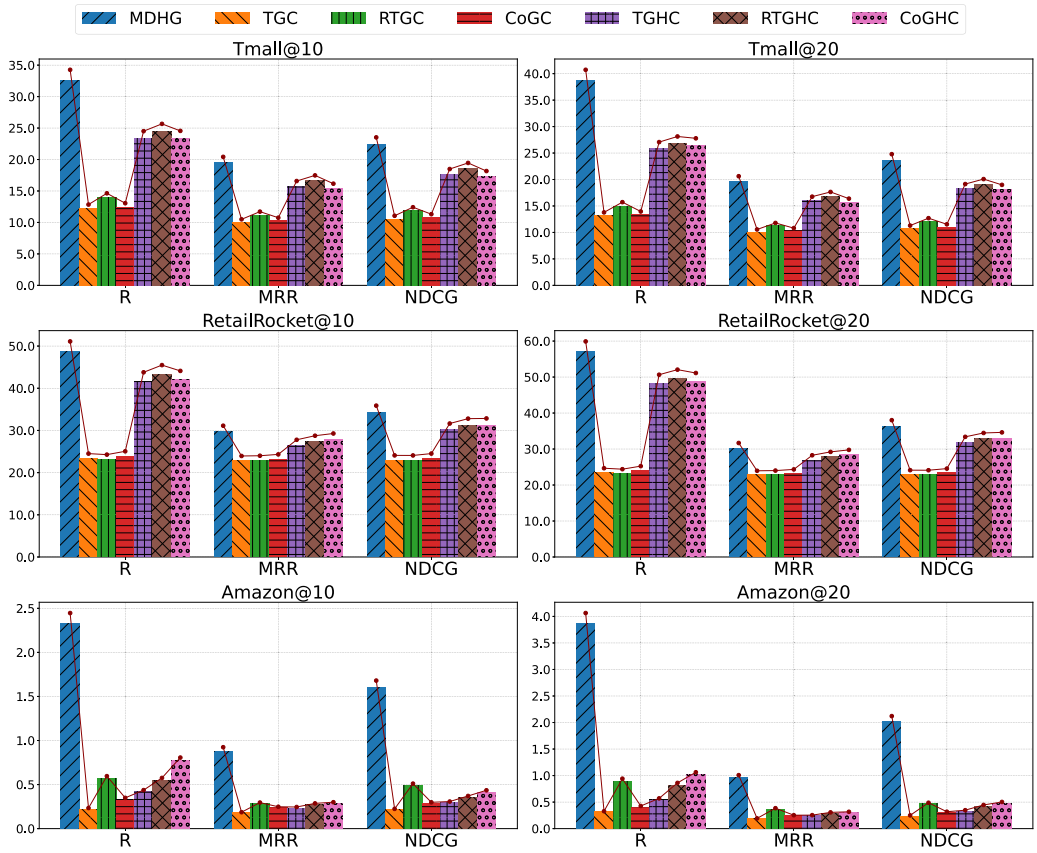


Fig. 3. Performances of different graph and hypergraph encoder.

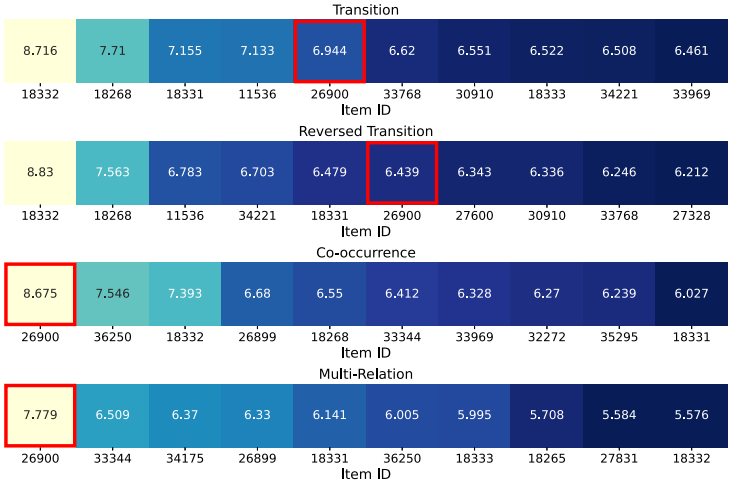


Fig. 4. The example from Tmall for relation case study.

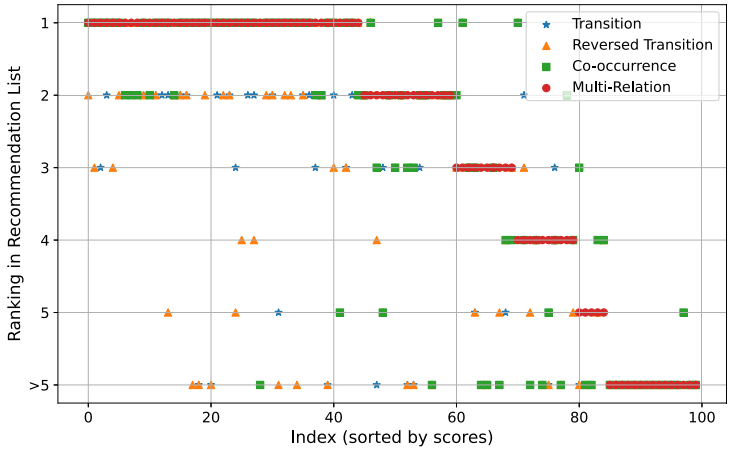


Fig. 5. Score rankings of 100 random sessions for three relation models and our MDHG model.

predict the target item using the co-occurrence relation. The results show that the multi-relation model fuses the three relations so that accurate choices can be made.

In addition, in order to fully demonstrate the help of the multi-relation structure for accurate recommendation, we randomly selected 100 recommendation results under the guarantee of the first 20 hits for different variants. The results are shown in Figure 5. Where the horizontal axis is the sample index and the vertical axis is the ranking of the target item in the recommendation list. For a clear presentation, we sort the results according to the recommendation scores of the multi-relation model from largest to smallest and ensure that the results for the four variants under a single index are corresponding. From the results, it is clear that different relations are sensitive to various degrees of target items. In the vast majority of cases, the multi-relation model can effectively combine the results of the three relations and make the optimal choice.

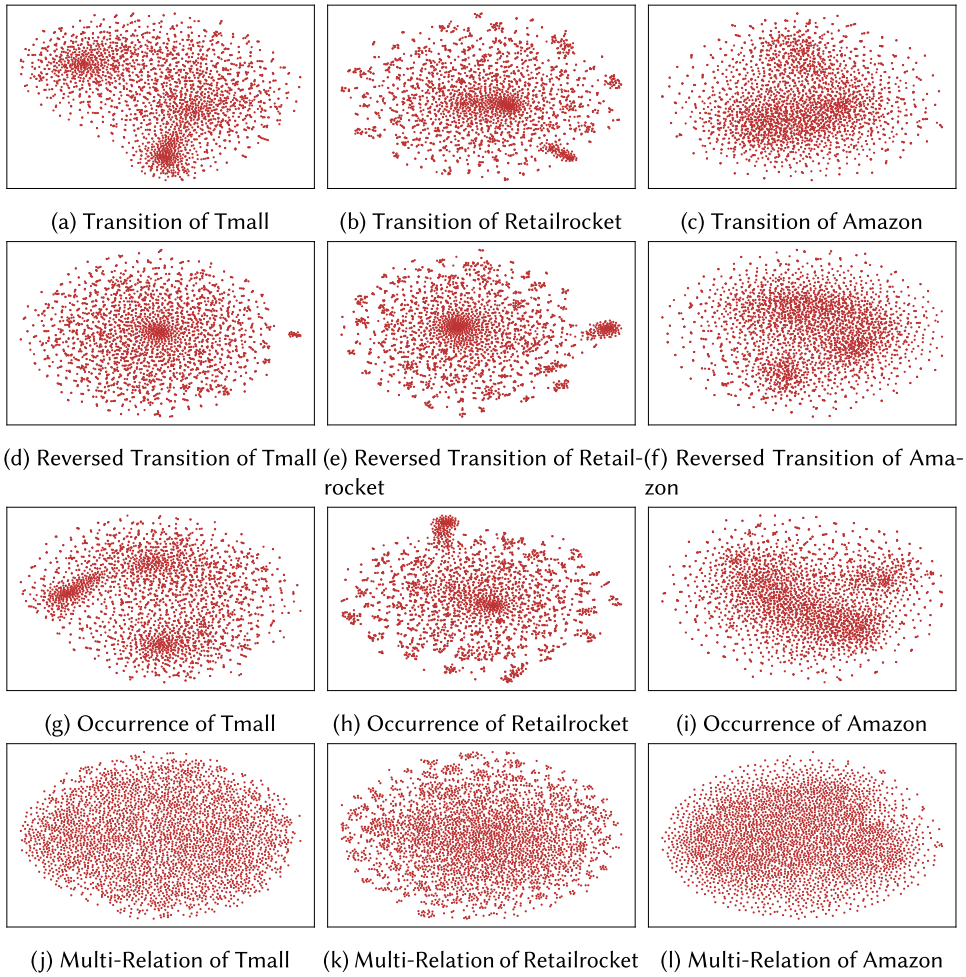
5.7 Visualization Analysis (RQ4)

In this subsection, we use T-SNE to visualize the learned item embeddings to explore the state of different relation patterns in the embedding space. Specifically, we construct three single-relation MDHG models (Transition, Reversed Transition and Occurrence) and train with the MDHG model on the three datasets. We then visualize the learned item representations using the T-SNE method. The results are shown in Figure 6, where the red dots indicate items.

The following observations can be made based on Figure 6: (1) Single relation tends to concentrate in some areas. As the dynamic hypergraph deepens a particular pattern of relation, it makes the item nodes tend to cluster together driven by that pattern. (2) Different scales of datasets play an important role in the distribution of nodes. The Amazon dataset is relatively smaller and thus captures sparse specific relation patterns. (3) Our MDHG model has a more even distribution of items over discrete space. We use the maximize mutual information and cross-relation aggregation modules to balance the impact on item representations while preserving each relation-specific information. These observations support the superiority of our multi-relation modeling.

5.8 Hyperparameter Experiments (RQ5)

In this subsection, we discuss the effect of two important parameters in our model, the number of network layers L and the number of hyperedges K, on the effectiveness of our model. We conduct experiments on all three datasets.



(a) Transition of Tmall (b) Transition of Retailrocket (c) Transition of Amazon

(d) Reversed Transition of Tmall (e) Reversed Transition of Retail-rocket (f) Reversed Transition of Amazon

(g) Occurrence of Tmall (h) Occurrence of Retailrocket (i) Occurrence of Amazon

(j) Multi-Relation of Tmall (k) Multi-Relation of Retailrocket (l) Multi-Relation of Amazon

Fig. 6. Visualization of learned item embeddings in different relation patterns on three real-world datasets using T-SNE. Red dots indicate items. (a)–(i) Item representations tend to cluster in discrete space under separate relation modes. (j)–(l) The fused item representations explore most of the space more evenly.

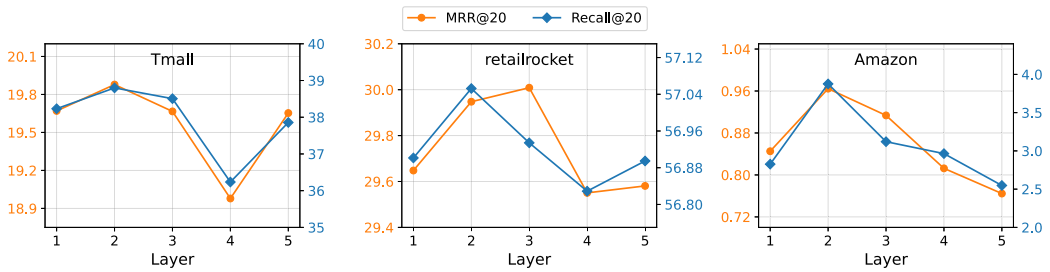


Fig. 7. The impacts of layer number.

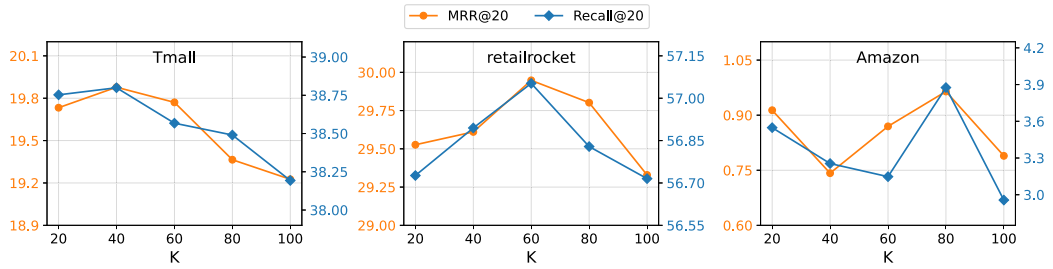


Fig. 8. The impacts of K number.

5.8.1 The Impacts of Layer Numbers. We vary the number of layers from 1 to 5. The experimental results, depicted in Figure 7, reveal that the model achieves optimal performance when $L = 2$. Notably, the model maintains a consistently satisfactory performance across the first three layers, while deeper layers exhibit a significant degradation in performance. We attribute this observation to the dynamic hypergraph structure, which introduces an increased number of parameters within each layer of the network. Excessive augmentation of the network layers can lead to overfitting, whereby the model becomes too specialized to the training data and performs poorly on unseen data. Consequently, a balance must be struck between the number of layers and the model's generalization ability. The results indicate that adding more layers beyond a certain threshold does not contribute to improved performance and instead hampers the model's effectiveness. Therefore, to ensure optimal performance while avoiding overfitting, we recommend setting the number of network layers to 2 in this particular model configuration. This choice strikes a balance between capturing the necessary complexity of the data and preventing the model from becoming overly complex and prone to overfitting.

5.8.2 The Impacts of Hyperedge Numbers. To investigate the impacts of the number of hyperedges on the model's performance, we conduct experiments with K values ranging from 20 to 100. As shown in Figure 8, the best performance is achieved when $K = 40$ for the Tmall dataset, while $K = 60$ yields the optimal results for the RetailRocket dataset and $K = 80$ for the Amazon dataset. Tmall, being a small-scale dataset, contains fewer intricate relationships between items. Consequently, setting a smaller number of hyperedges proves suitable for capturing the underlying patterns in this dataset. In contrast, the RetailRocket dataset, with its larger scale and potentially more complex relationships, benefits from a slightly higher number of hyperedges to effectively represent the nuanced interactions between items. Finally, for the Amazon dataset, it reaches optimality with the maximum number of hyperedges. Since this dataset is too sparse and has many new items, a large number of hyperedges are needed to explore possible structures. These findings emphasize the importance of considering dataset characteristics when determining the number of hyperedges. The fixed number of hyperedges may not yield optimal results, as different datasets exhibit diverse complexities and relation structures.

5.9 The Analysis of Running Time (RQ6)

First, we analyze the time complexity of our MDHG model and compare it with a general hypergraph convolution process. The key operation of the MDHG model is "relation graph-dynamic hypergraph" learning. The relation graph part consists of feature transformation and matrix multiplication with complexity $O(n * d^2 + |E| * d)$. The dynamic hypergraph module is divided into dynamic hyperedge construction and hypergraph convolution, with complexity of $O(n*d^2+n*d*k)$ and $O(n*k^2+n*k*d)$, respectively. Overall, the time complexity of our model is $O(n * d^2 + |E| * d + n * k * d + n * k^2)$.

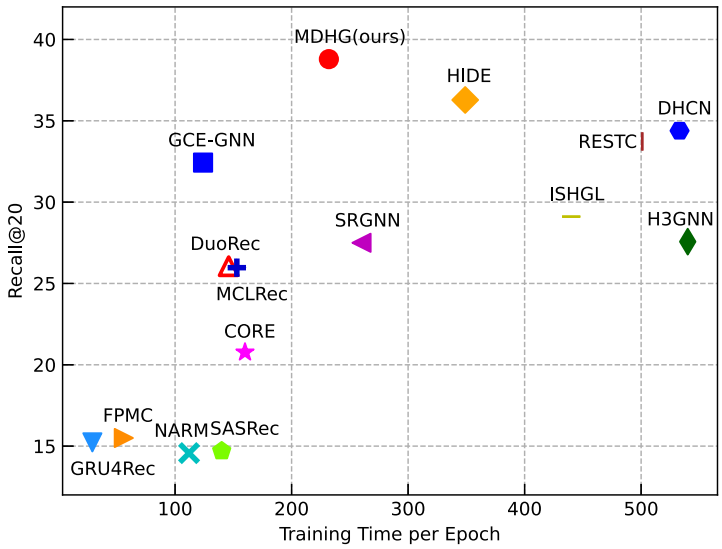


Fig. 9. Running time (seconds) of each epoch.

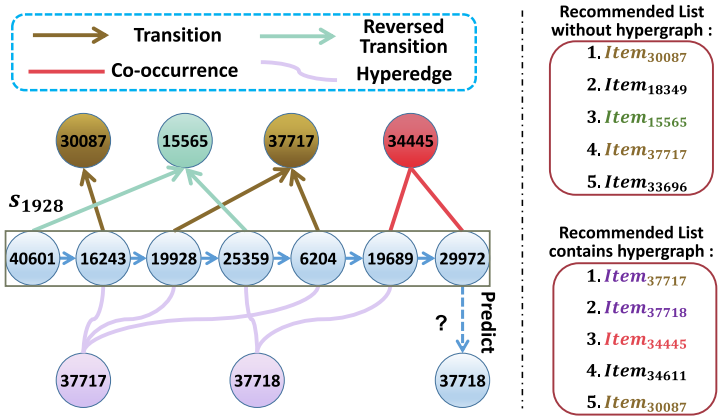


Fig. 10. A case study was conducted to evaluate the effectiveness of the hypergraph module.

In addition, the general hypergraph convolution has a complexity of $O(|E| * d + n * d^2)$. In contrast, our MDHG only adds $O(n * k * d + n * k^2)$, where k is the number of setup hyperedges, where $k < d \ll n$. Therefore, the time complexity of our model is acceptable and exhibits good scalability.

Then we evaluate the running time between our proposed methodology and all baseline recommendation strategies. As part of this study, we process iteratively for 20 epochs, using Tmall dataset for each of strategies mentioned above. Next, we compute the average temporal metrics for these iterations. Figure 9 presents the outcomes that demonstrate the superiority of our approach in terms of run-time efficiency for all instances. On the one hand, our approach is innovative in its concurrent training of hypergraphs alongside relation graphs, significantly improving operational efficiency while also outperforming other hypergraph-based methods such as DHCN, HIDE, H3GNN and ISHGL. On the other hand, compared to traditional baselines, although our approach runs slightly slower than theirs, it has a huge advantage in terms of performance.

5.10 Case Study of Overall Model (RQ7)

In this subsection, we use concrete instances to analyze the practical impact of different relation graphs and dynamic hypergraph on the effect of our model, as shown in Figure 10. For session 1,928, it is first connected to other nodes through explicit relations (e.g., 30,087, 15,565). In the model without the hypergraph structure, it can be seen that the explicit nodes are ranked at the top. And when we additionally apply the hypergraph structure, the current session is able to connect to implicit high-order nodes that could not be captured before through the hyperedges (e.g., 37,717, 37,718). This is reflected in the recommendation list, where the target item 37,718 is successfully predicted. Notably, item 37,717 is captured by both explicit and implicit relationships and is therefore placed first by our model.

6 Conclusion

In this article, we propose a novel MDHG learning framework for SBR systems. The goal is to address the limitations of existing methods, which fail to fully exploit the explicit user behavior patterns and implicit higher-order information. Our MDHG framework introduces three relation graphs to efficiently model high-order item relations and capture a complete representation of user behavior patterns. Additionally, a dynamic layer-wise hypergraph learning mechanism was employed to adaptively generate hypergraphs at each layer during the training process for each relation. A self-supervised learning paradigm is used to reduce inter-relation noise, and an attention mechanism is incorporated to comprehensively integrate different relation representations. Extensive experiments on real-world datasets demonstrate the effectiveness of the MDHG model compared to state-of-the-art baselines.

References

[1] Sameer Agarwal, Kristin Branson, and Serge Belongie. 2006. Higher order learning with graphs. In *Proceedings of the 23rd International Conference on Machine Learning*, 17–24.

[2] Guojia An, Jing Sun, Yuhan Yang, and Fuming Sun. 2024. Enhancing collaborative information with contrastive learning for session-based recommendation. *Information Processing & Management* 61, 4 (2024), 103738.

[3] Wanyu Chen, Fei Cai, Honghui Chen, and Maarten De Rijke. 2019. Joint neural collaborative filtering for recommender systems. *ACM Transactions on Information Systems* 37, 4 (2019), 1–30.

[4] Kaize Ding, Jianling Wang, Jundong Li, Dingcheng Li, Huan Liu, and Yang Liu. 2020. Be more with less: Hypergraph attention networks for inductive text classification. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. Bonnie Webber, Trevor Cohn, Yulan He (Eds.), Association for Computational Linguistics, 4927–4936.

[5] Xiu Susie Fang, Yonggang Wu, Jinhu Lu, Xiaoyu Gu, Guohao Sun, and Yong Zhan. 2024. Intent enhanced self-supervised hypergraph learning for session-based recommendation. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*. Springer, 83–99.

[6] Yifan Feng, Haoxuan You, Zizhao Zhang, Rongrong Ji, and Yue Gao. 2019. Hypergraph neural networks. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 33, 3558–3565.

[7] Yue Gao, Yifan Feng, Shuyi Ji, and Rongrong Ji. 2022. Hgnn+: General hypergraph neural networks. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 45, 3 (2022), 3181–3199.

[8] Yingqiang Ge, Shuya Zhao, Honglu Zhou, Changhua Pei, Fei Sun, Wenwu Ou, and Yongfeng Zhang. 2020. Understanding echo chambers in e-commerce recommender systems. In *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval*, 2261–2270.

[9] Liangmin Guo, Shiming Zhou, Haiyue Tang, Xiaoyao Zheng, and Yonglong Luo. 2025. Multi-behavior hypergraph contrastive learning for session-based recommendation. *IEEE Transactions on Knowledge and Data Engineering* 37, 3 (2025), 1325–1338. DOI: https://doi.org/10.1109/TKDE.2024.3523383

[10] Xiangnan He, Kuan Deng, Xiang Wang, Yan Li, Yongdong Zhang, and Meng Wang. 2020. Lightgcn: Simplifying and powering graph convolution network for recommendation. In *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval*, 639–648.

[11] Balázs Hidasi, Alexandros Karatzoglou, Linas Baltrunas, and Domonkos Tikk. 2016. Session-based recommendations with recurrent neural networks. In *Proceedings of the 4th International Conference on Learning Representations (ICLR '16)*. Yoshua Bengio and Yann LeCun (Eds.). Retrieved from http://arxiv.org/abs/1511.06939

[12] Yupeng Hou, Binbin Hu, Zhiqiang Zhang, and Wayne Xin Zhao. 2022. Core: Simple and effective session-based recommendation within consistent representation space. In *Proceedings of the 45th International ACM SIGIR Conference on Research and Development in Information Retrieval*, 1796–1801.

[13] Jongwon Jeong, Jeong Choi, Hyunsouk Cho, and Sehee Chung. 2022. FPAdaMetric: False-positive-aware adaptive metric learning for session-based recommendation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 36, 4039–4047.

[14] Jianwen Jiang, Yuxuan Wei, Yifan Feng, Jingxuan Cao, and Yue Gao. 2019. Dynamic hypergraph neural networks. In *Proceedings of the 28th International Joint Conference on Artificial Intelligence*, 2635–2641.

[15] Di Jin, Luzhi Wang, Yizhen Zheng, Guojie Song, Fei Jiang, Xiang Li, Wei Lin, and Shirui Pan. 2023. Dual intent enhanced graph neural network for session-based new item recommendation. In *Proceedings of the ACM Web Conference 2023*, 684–693.

[16] Santosh Kabbur, Xia Ning, and George Karypis. 2013. Fism: Factored item similarity models for top-n recommender systems. In *Proceedings of the 19th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 659–667.

[17] Wang-Cheng Kang and Julian McAuley. 2018. Self-attentive sequential recommendation. In *Proceedings of the 2018 IEEE International Conference on Data Mining (ICDM)*. IEEE, 197–206.

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

[19] Jing Li, Pengjie Ren, Zhumin Chen, Zhaochun Ren, Tao Lian, and Jun Ma. 2017. Neural attentive session-based recommendation. In *Proceedings of the 2017 ACM on Conference on Information and Knowledge Management*, 1419–1428.

[20] Yinfeng Li, Chen Gao, Hengliang Luo, Depeng Jin, and Yong Li. 2022. Enhancing hypergraph neural networks with intent disentanglement for session-based recommendation. In *Proceedings of the 45th International ACM SIGIR Conference on Research and Development in Information Retrieval*, 1997–2002.

[21] Yinfeng Li, Chen Gao, Quanming Yao, Tong Li, Depeng Jin, and Yong Li. 2022. DisenHCN: Disentangled hypergraph convolutional networks for spatiotemporal activity prediction. In *Proceedings of the 2022 IEEE 38th International Conference on Data Engineering (ICDE)*. IEEE.

[22] Qiao Liu, Yifu Zeng, Refuoe Mokhosi, and Haibin Zhang. 2018. STAMP: Short-term attention/memory priority model for session-based recommendation. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, 1831–1839.

[23] Tianze Luo, Yong Liu, and SinnoJialin Pan. 2024. Collaborative sequential recommendations via multi-view GNN-transformers. *ACM Transactions on Information Systems* 42, 6 (2024), 1–27. DOI: https://doi.org/10.1145/3649436

[24] Zhiqiang Pan, Fei Cai, Wanyu Chen, Honghui Chen, and Maarten De Rijke. 2020. Star graph neural networks for session-based recommendation. In *Proceedings of the 29th ACM International Conference on Information & Knowledge Management*, 1195–1204.

[25] Andreas Peintner, Amir Reza Mohammadi, and Eva Zangerle. 2025. Efficient session-based recommendation with contrastive graph-based shortest path search. *ACM Transactions on Recommender Systems* 3, 4 (2025), 1–24.

[26] Xiuyuan Qin, Huanhuan Yuan, Pengpeng Zhao, Junhua Fang, Fuzhen Zhuang, Guanfeng Liu, Yanchi Liu, and Victor Sheng. 2023. Meta-optimized contrastive learning for sequential recommendation. In *Proceedings of the 46th International ACM SIGIR Conference on Research and Development in Information Retrieval* (SIGIR '23). ACM, New York, NY, 89–98.

[27] Ruihong Qiu, Zi Huang, Hongzhi Yin, and Zijian Wang. 2022. Contrastive learning for representation degeneration problem in sequential recommendation. In *Proceedings of the 15th ACM International Conference on Web Search and Data Mining (WSDM '22)*. ACM, New York, NY, 813–823.

[28] Ruihong Qiu, Jingjing Li, Zi Huang, and Hongzhi Yin. 2019. Rethinking the item order in session-based recommendation with graph neural networks. In *Proceedings of the 28th ACM International Conference on Information and Knowledge Management*, 579–588.

[29] Steffen Rendle, Christoph Freudenthaler, and Lars Schmidt-Thieme. 2010. Factorizing personalized Markov chains for next-basket recommendation. In *Proceedings of the 19th International Conference on World Wide Web*, 811–820.

[30] J. Ben Schafer, Dan Frankowski, Jon Herlocker, and Shilad Sen.2007. Collaborative filtering recommender systems. In *The Adaptive Web: Methods and Strategies of Web Personalization*, Springer Berlin Heidelberg, Berlin, 291–324.

[31] Guy Shani, David Heckerman, Ronen I. Brafman, and Craig Boutilier. 2005. An MDP-based recommender system. *Journal of Machine Learning Research* 6, 9 (2005), 1265–1295.

[32] Jiajie Su, Chaochao Chen, Weiming Liu, Fei Wu, Xiaolin Zheng, and Haoming Lyu. 2023. Enhancing hierarchy-aware graph networks with deep dual clustering for session-based recommendation. In *Proceedings of the ACM Web Conference 2023*, 165–176.

[33] Yong Kiam Tan, Xinxing Xu, and Yong Liu. 2016. Improved recurrent neural networks for session-based recommendations. In *Proceedings of the 1st Workshop on Deep Learning for Recommender Systems*, 17–22.

[34] Zhongwei Wan, Xin Liu, Benyou Wang, Jiezhong Qiu, Boyu Li, Ting Guo, Guangyong Chen, and Yang Wang. 2023. Spatio-temporal contrastive learning-enhanced GNNs for session-based recommendation. *ACM Transactions on Information Systems* 42, 2 (2023), 1–26.

[35] Wenjie Wang, Fuli Feng, Xiangnan He, Hanwang Zhang, and Tat-Seng Chua. 2021. Clicks can be cheating: Counterfactual recommendation for mitigating clickbait issue. In *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval*, 1288–1297.

[36] Xiang Wang, Xiangnan He, Meng Wang, Fuli Feng, and Tat-Seng Chua. 2019. Neural graph collaborative filtering. In *Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval*, 165–174.

[37] Xiao Wang, Nian Liu, Hui Han, and Chuan Shi. 2021. Self-supervised heterogeneous graph neural network with co-contrastive learning. In *Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining*, 1726–1736.

[38] Ziyang Wang, Wei Wei, Gao Cong, Xiao-Li Li, Xian-Ling Mao, and Minghui Qiu. 2020. Global context enhanced graph neural networks for session-based recommendation. In *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval*, 169–178.

[39] Chunyu Wei, Jian Liang, Bing Bai, and Di Liu. 2022. Dynamic hypergraph learning for collaborative filtering. In *Proceedings of the 31st ACM International Conference on Information & Knowledge Management*, 2108–2117.

[40] 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.

[41] Lianghao Xia, Chao Huang, and Chuxu Zhang. 2022. Self-supervised hypergraph transformer for recommender systems. In *Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, 2100–2109.

[42] Xin Xia, Hongzhi Yin, Junliang Yu, Qinyong Wang, Lizhen Cui, and Xiangliang Zhang. 2021. Self-supervised hypergraph convolutional networks for session-based recommendation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 35, 4503–4511.

[43] Liqi Yang, Linhan Luo, Lifeng Xin, Xiaofeng Zhang, and Xinni Zhang. 2022. DAGNN: Demand-aware graph neural networks for session-based recommendation. In *Proceedings of the 45th International ACM SIGIR Conference on Research and Development in Information Retrieval*.

[44] Yuhao Yang, Chao Huang, Lianghao Xia, Yuxuan Liang, Yanwei Yu, and Chenliang Li. 2022. Multi-behavior hypergraph-enhanced transformer for sequential recommendation. In *Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, 2263–2274.

[45] Jaehyuk Yi and Jinkyoo Park. 2020. Hypergraph convolutional recurrent neural network. In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, 3366–3376.

[46] Nan Yin, Fuli Feng, Zhigang Luo, Xiang Zhang, Wenjie Wang, Xiao Luo, Chong Chen, and Xian-Sheng Hua. 2022. Dynamic hypergraph convolutional network. In *Proceedings of the 2022 IEEE 38th International Conference on Data Engineering (ICDE)*. IEEE, 1621–1634.

[47] Zhizhuo Yin, Kai Han, Pengzi Wang, and Xi Zhu. 2023. H3gnn: Hybrid hierarchical hypergraph neural network for personalized session-based recommendation. *ACM Transactions on Information Systems* 42, 3 (2023), 1–30.

[48] Junliang Yu, Hongzhi Yin, Jundong Li, Qinyong Wang, Nguyen Quoc Viet Hung, and Xiangliang Zhang. 2021. Self-supervised multi-channel hypergraph convolutional network for social recommendation. In *Proceedings of the Web Conference 2021*, 413–424.

[49] Jiahao Yuan, Zihan Song, Mingyou Sun, Xiaoling Wang, and Wayne Xin Zhao. 2021. Dual sparse attention network for session-based recommendation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 35, 4635–4643.

[50] Chengyuan Zhang, Yang Wang, Lei Zhu, Jiayu Song, and Hongzhi Yin. 2021. Multi-graph heterogeneous interaction fusion for social recommendation. *ACM Transactions on Information Systems* 40, 2 (2021), 1–26.

[51] Peiyan Zhang, Jiayan Guo, Chaozhuo Li, Yueqi Xie, Jae Boum Kim, Yan Zhang, Xing Xie, Haohan Wang, and Sunghun Kim. 2023. Efficiently leveraging multi-level user intent for session-based recommendation via atten-mixer network. In *Proceedings of the 16th ACM International Conference on Web Search and Data Mining*, 168–176.

[52] Xiaokun Zhang, Bo Xu, Fenglong Ma, Chenliang Li, Yuan Lin, and Hongfei Lin. 2023. Bi-preference learning heterogeneous hypergraph networks for session-based recommendation. *ACM Transactions on Information Systems* 42, 3 (2023), 1–28.

[53] Xiaokun Zhang, Bo Xu, Liang Yang, Chenliang Li, Fenglong Ma, Haifeng Liu, and Hongfei Lin. 2022. Price does matter! Modeling price and interest preferences in session-based recommendation. In *Proceedings of the 45th International ACM SIGIR Conference on Research and Development in Information Retrieval*, 1684–1693.

[54] Yusheng Zhao, Xiao Luo, Wei Ju, Chong Chen, Xian-Sheng Hua, and Ming Zhang. 2023. Dynamic hypergraph structure learning for traffic flow forecasting. In *Proceedings of the 2023 IEEE 39th International Conference on Data Engineering (ICDE)*. IEEE, 2303–2316.

[55] Peng Zhou, Zongqian Wu, Xiangxiang Zeng, Guoqiu Wen, Junbo Ma, and Xiaofeng Zhu. 2023. Totally dynamic hypergraph neural network. In *Proceedings of the 32nd International Joint Conference on Artificial Intelligence*, 2476–2483.

[56] Xingrui Zhuo, Shengsheng Qian, Jun Hu, Fuxin Dai, Kangyi Lin, and Gongqing Wu. 2024. Multi-hop multi-view memory transformer for session-based recommendation. *ACM Transactions on Information Systems* 42, 6 (2024), 1–28. DOI: https://doi.org/10.1145/3663760

Received 5 October 2024; revised 11 August 2025; accepted 3 October 2025