# Knowledge-enhanced Multi-View Graph Neural Networks for Session-based Recommendation

Qian Chen
School of Computer Science and Technology, Huazhong University of Science and Technology, China
chenqian0201@hust.edu.cn

Zhiqiang Guo*
School of Computer Science and Technology, Huazhong University of Science and Technology, China
zhiqiangguo@hust.edu.cn

Jianjun Li*
School of Computer Science and Technology, Huazhong University of Science and Technology, China
jianjunli@hust.edu.cn

Guohui Li
School of Software Engineering, Huazhong University of Science and Technology, China
guohuili@hust.edu.cn

## ABSTRACT

Session-based recommendation (SBR) has received increasing attention to predict the next item via extracting and integrating both global and local item-item relationships. However, there still exist some deficiencies in current works when capturing these two kinds of relationships. For global item-item relationships, the global graph constructed by most SBR is a pseudo-global graph, which may cause redundant mining of sequence relationships. For local item-item relationships, conventional SBR only mines the sequence patterns while ignoring the feature patterns, which may introduce noise when learning users' interests. To address these problems, we propose a novel Knowledge-enhanced Multi-View Graph Neural Network (KMVG) by constructing three views, namely knowledge view, session view, and pairwise view. Specifically, benefiting from the rich semantic information in the knowledge graph (KG), we build a genuine global graph that is sequence-independent based on KG to mine the global item-item relationships in the knowledge view. Then, a session view is utilized to capture the contextual transitions among items as the sequence patterns of local item-item relationships, and a pairwise view is used to explore the feature commonality within a session as the feature patterns of the local item-item relationships. Extensive experiments on three real-world public datasets demonstrate the superiority of KMVG, showing that it outperforms the state-of-the-art baselines. Further analysis also reveals the effectiveness of KMVG in exploiting the item-item relationships under multiple views.

## CCS CONCEPTS

• **Information systems** → *Recommender systems.*

## KEYWORDS

Session-based Recommendation, Graph Neural Network, Knowledge Graph

**ACM Reference Format:**
Qian Chen, Zhiqiang Guo, Jianjun Li, and Guohui Li. 2023. Knowledge-enhanced Multi-View Graph Neural Networks for Session-based Recommendation. In *Proceedings of the 46th International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR '23), July 23–27, 2023, Taipei, Taiwan.* ACM, New York, NY, USA, 10 pages. https://doi.org/10.1145/3539618.3591706

## 1 INTRODUCTION

Recommender system (RS) is an effective solution to help users find desired items from a large number of options to resist the problem of *information overload*. As the most popular recommendation method, collaborative filtering (CF) typically relies on user profiles and the long-term historical interactions between users and items [10, 18] to make recommendation. However, in some real-world scenarios where user information is unavailable (e.g., the unlogged-in user or anonymous user) or the interaction information is limited, conventional CF approaches may perform unsatisfactorily. Consequently, session-based recommendation (SBR) comes into being and has attracted extensive attention recently. Each session consists of multiple interacted items in chronological order that occurred over a short continuous period, and the basic task of SBR is to predict the next item based on a given anonymous session.

Some early studies based on Markov chains [17, 19] predict the next possible item by modeling the transformation of items in a session. Under the strong independence assumption, independent combinations of past behaviors confine the prediction accuracy. Later, Recurrent Neural Networks (RNNs) have been introduced to address the SBR problem and achieved remarkable progress [5, 12, 21, 22, 25]. However, due to the well-known problem of long-distance dependencies and only modeling one-way transitions between consecutive items, these RNN-based methods are insufficient to mine the transition between items within a session to model item representations. Recently, given its strong ability in mining higher-order connectivity, Graph Neural Network (GNN) has gained much attention to represent the session by propagating information between adjacent items [1, 31, 33]. However, limited by the data sparsity problem caused by short-term behavior, only exploiting the contextual relationship within a session to improve

*Corresponding author.

*SIGIR '23, July 23–27, 2023, Taipei, Taiwan*
© 2023 Copyright held by the owner/author(s). Publication rights licensed to ACM.
ACM ISBN 978-1-4503-9408-6/23/07…$15.00
https://doi.org/10.1145/3539618.3591706

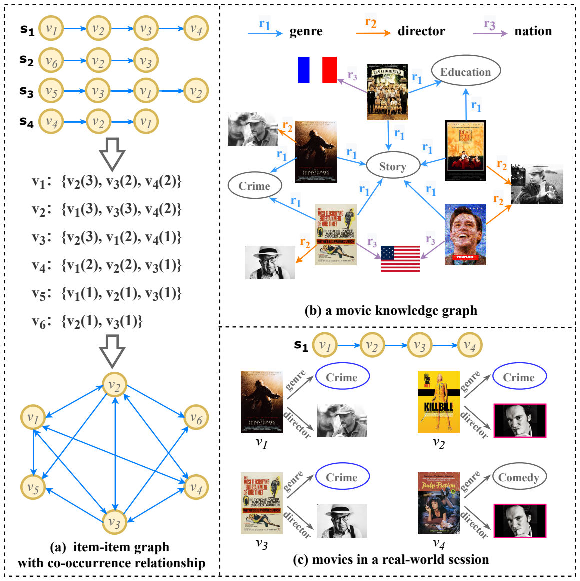

s1 v1 → v2 → v3 → v4
s2 v6 → v2 → v3
s3 v5 → v3 → v1 → v2
s4 v4 → v2 → v1
v1: {v2(3), v3(2), v4(2)}
v2: {v1(3), v3(3), v4(2)}
v3: {v2(3), v1(2), v4(1)}
v4: {v1(2), v2(2), v3(1)}
v5: {v1(1), v2(1), v3(1)}
v6: {v2(1), v3(1)}
(a) item-item graph with co-occurrence relationship
r1 genre
r2 director
r3 nation
Education
Story
Crime
(b) a movie knowledge graph
s1 v1 → v2 → v3 → v4
genre
director
Crime
Comedy
v1
v2
v3
v4
(c) movies in a real-world session

**Figure 1: Illustration of item-item relationships. (a) presents the process of constructing global item-item graph with co-occurrence relationship. (b) is a real-world movie knowledge graph. (c) shows two kinds of explicit attributions of movies within a session.**

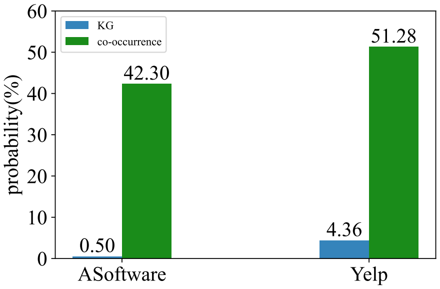


**Figure 2: The overlap probability of neighbors in knowledge/co-occurrence graph and contextual neighbors in session for ASoftware and Yelp.**

the representation ability and thereby boost model performance has encountered a bottleneck.

Recently, a growing body of studies has begun to extract cross-session item relations as a complement to intra-session item representation learning [16, 30, 34]. Specifically, these SBR methods regard cross-session item relations and intra-session item relations as global and local item-item relationships respectively, and utilize graph neural networks to model item and session representations [30]. However, these methods still have some limitations in capturing these two kinds of relationships. **Firstly**, global graphs constructed by sessions (e.g., the co-occurrence global item-item graph) are usually the first choice for these methods to model global item-item relationships. However, propagating messages on such graphs will cause the problem of redundant mining of sequence patterns and only capturing the pseudo-global item-item relationship. As an illustrative example, Figure 1(a) presents a constructed co-occurrence global item-item graph, in which the frequency of every two items appearing in the same session are counted, and the top-3 items with the highest co-occurrence frequency are selected as the neighbors of the current item. It can be observed that the coincidence degree of the item's neighbors in the co-occurrence global graph and the contextual neighbors in the session is fairly high. Actually, the proportion of coincidence reached 42.3% and 51.28% on datasets ASoftware and Yelp, respectively, according to our statistical results shown in Figure 2. Therefore, existing methods cannot make full use of global item-item relationships to improve item representation quality. **Secondly**, most SBR methods under-explore local item-item relationships. They conventionally only treat the item context in session as local item-item relationships to explore the single sequence patterns. We emphasize that there are rich relations between items in a session, and insufficient local relation mining will lead to insufficient sequence representation learning and weak performance. In particular, items in the same session generally have feature commonality. Figure 1(c) shows the attributions of three movies in a real session. Obviously, the three movies $v_1$, $v_2$, and $v_3$ explicitly have the same movie genre *crime*, while movies $v_2$ and $v_4$ have the same director. We can speculate that the occurrence of interactions in this session may depend on the common feature points of items, i.e., *crime* and *Quentin Tarantino*. Hence, simply aggregating all items in a session to model session representations lacks a keen sense of the commonality among items and may introduce irrelevant noise.

To address the aforementioned limitations, we propose a novel Knowledge-enhanced Multi-View Graph Neural Network (KMVG) to enhance item and session representations by exploiting global and local item-item relationships under three views, *knowledge view*, *session view*, and *pairwise view*. Specifically, **to address the first limitation**, we construct a global knowledge graph (KG) to capture the global item-item relations as *knowledge view*. Different from global item-item graph that defines nodes' neighbors through co-occurrence relationships, a knowledge graph contains the real semantic relationships among items and links items via the common attributes, as illustrated in Figure 1(b), which results in a relatively low degree of coincidence between the item's neighbors in KG and the contextual neighbors in the session. As an evidence, we can observe from Figure 2 that the overlap probability of neighbors in KG and contextual neighbors in sessions are 0.5% and 4.36% for datasets ASoftware and Yelp, respectively. Therefore, propagating messages on KG can avoid the redundant mining of sequence dependencies. Based on the constructed KG, we further employ a knowledge graph attention network to aggregate item semantic representations to capture ground-truth global item-item relationships in the knowledge view. **To address the second limitation**, we not only develop a graph attention network to extract item representation with sequence contextual patterns from session graph in the *session view*, but also design a simple yet effective pairwise item aggregator to extract feature commonality patterns between any two items within a session in the *pairwise view*, so as to express the current session with more fine-grained and accurate representation. Afterward, we adopt a position-aware soft-attention mechanism to fuse the item embeddings from the session view to represent the current session. Finally, the session representations from different views are further integrated for next-item prediction. The main contributions of this work can be outlined as follows:

- We emphasize the importance of global and local item-item relationships for improving the performance of SBR and construct three views to fully mine multiple item-item relationships.
- We utilize KG to extract the global item-item relationship, which can effectively alleviate both the data sparsity and redundant mining problems.
- We extract both sequence patterns and feature patterns to represent local item-item relationships in a fine-grained manner.
- Extensive experiments on three real-world datasets demonstrate that KMVG outperforms several state-of-the-art baselines, and further results also verify the effectiveness of KMVG in capturing and utilizing multiple relationships among items to improve model performance.

## 2 RELATED WORK

In this section, we review some literature highly related to our work, including conventional SBR methods, GNN-based SBR methods, and KG-based recommendation methods.

**Conventional SBR Methods.** Early methods based on Markov chains predict the next-click item through the previous clicks and treat SBR as a sequential optimization problem. Some works [17, 19] employ Markov decision processes to solve the problem. However, the strong independence assumption followed by these methods confines the prediction accuracy. To model the transition between items, RNNs have received much attention by many researchers for SBR [5, 7, 12, 21]. For example, GPU4Rec uses Gated Recurrent Unit (GRU) [2] to model the sequential behavior of items [5]. Li et al. [12] proposed NARM by employing an attention mechanism on RNNs to capture users' intent of sequential behavior. However, these methods only model the single-way transitions between consecutive items, neglecting the complex contextual transitions.

**GNN-based SBR Methods.** Due to its strong representation learning ability, GNN has been widely used in subsequent SBR methods [30, 31, 33, 34]. SR-GNN is a typical model to capture the complex contextual transition between adjacent items by applying a gated graph neural network [31]. GC-SAN uses a self-attention mechanism to learn the global dependencies between distant positions [33]. Though GNN-based methods have shown promising performance, the session graphs face a lossy encoding problem. To address this problem, Chen and Wong [1] proposed to aggregate information effectively by lossless edge-order preserving aggregation and shortcut graph attention. Pan et al. [15] proposed to use a star node to bridge items by filtering out irrelevant items. Xia et al. [32] proposed to augment session data by co-training to capture the intent of users more accurately. Zhang et al. [37] introduced the price factor into session-based recommendation and Guo et al. [3] used heterogeneous graph attention network to get the intent representations from multi-granularity levels. However, limited by the session sparsity problem caused by short-term behavior, only exploiting the relationship among items within a session is insufficient to improve the session representation. Recently, CSRM [16] is proposed to incorporate collaborative information via a memory network to enrich the representation of the current session, boosting the recommendation performance. GCE-GNN [30] treats items as the minimum granularity to learn the transition from global-level and session-level graphs. Ye et al. [34] proposed CA-TCN to explore the cross-session influence on item-level and session-level simultaneously. While encouraging, these works usually utilize relationships extracted from sessions as global representation, resulting in redundant mining of sequence correlations, which is not conducive to recommendation performance.

**KG-based Recommendation.** KG has been widely used in recommender systems to alleviate the data sparsity problem, due to its rich semantic information. CKE [35] is the first work to use structured information of knowledge base for collaborative filtering by utilizing TransR [13] as transform. Later, Huang et al. [6] proposed KSR to integrate with external memories by leveraging KG information on the sequential recommendation. KGNN-LS [24] is proposed to convert KG into user-specific graphs, by considering user preference on KG relations and label smoothness in the information aggregation phase to generate user-specific item representations. KGAT [26] combines KG with the user-item graph and applies an attentive neighborhood aggregation mechanism on a holistic graph. KGIN [28] further reveals user intents behind the interaction with KG information. KSTT [36] introduces KG into SBR with temporal transformer and MGS [11] utilizes items' properties as mirror graph and introduces a contrastive learning strategy to optimize model performance. Different from them, in this work, we introduce the knowledge graph as truth global item relations to mine the item semantic representations for better recommendation performance.

## 3 PRELIMINARIES

In this section, we first present the formal definition of the general SBR problem, and then introduce the graph structures of the three views, i.e., *knowledge-view graph*, *session-view graph*, and *pairwise-view graph*.

### 3.1 Problem Statement

Let $\mathcal{V} = \{v_1, v_2, ..., v_n\}$ represents all of the items in a recommender system. Each anonymous session consisting of a sequence of interacted items in chronological order is denoted as $\mathcal{S} = \{v_1^s, v_2^s, ..., v_l^s\}$, where $v_i^s$ denotes item $v_i$ clicked within session $\mathcal{S}$, and $l$ denotes the length of session $\mathcal{S}$. Given a session $\mathcal{S}$, the goal of SBR is to recommend the next item $v_{l+1}^s$ from $\mathcal{V}$ that is most likely to be clicked by the user of the current session $\mathcal{S}$. Therefore, we can formalize this process as,

$$p(v_{l+1}^s|\mathcal{S}) = \textsc{Predict}(\mathcal{S}, \mathcal{V}) \tag{1}$$

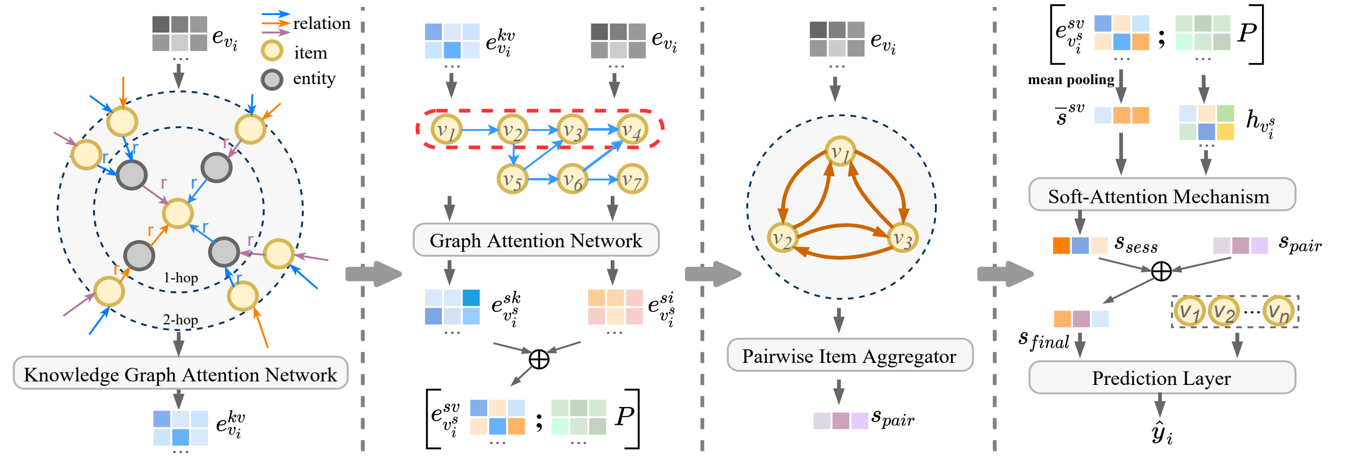
where $p(v_{l+1}^s|\mathcal{S})$ is the probability distributions of the predicted next item, $\textsc{Predict}(\cdot)$ is a prediction function to generate the probability of the next item that may be clicked by taking the session $\mathcal{S}$ and item space $\mathcal{V}$ as input.

### 3.2 Multi-View Graph Models

As mentioned earlier, we introduce three views to capture multiple relationships among items for modeling item and session representations. Next, we describe the graph structures and item dependency information of the three views in detail.

#### 3.2.1 *Knowledge-View Graph*.

We denote knowledge graph as $\mathcal{G}_{kv} = \{(h, r, t)|h, t \in \mathcal{V}_{kv}, r \in \mathcal{R}\}$, where $\mathcal{R}$ and $\mathcal{V}_{kv}$ represents the



**(a) Knowledge-View** **(b) Session-View** **(c) Pairwise-View** **(d) Ensemble and Prediction**

**Figure 3: Framework of the proposed KMVG.**

relation and the entity sets, respectively. Each triple $(h, r, t)$ means there is a relation $r$ from head entity $h$ to tail entity $t$. Noteworthy, the entity set $\mathcal{V}_{kv}$ is composed of all items $\mathcal{V}$ and its attribute entities. Without loss of generality, we use $h$ and $t$ to perform subsequent computations uniformly.

*3.2.2* ***Session-View Graph.*** We use items in a session $\mathcal{S}$ to construct a directed session graph, which is denoted as $\mathcal{G}_{sv} = (\mathcal{V}, \mathcal{E})$, where $\mathcal{E} = \{(v^s_i, v^s_j) | v^s_i, v^s_j \in \mathcal{S}, v_j \in \mathcal{A}(v_i)\}$, and $\mathcal{A}(v_i)$ denotes the next click of $v_i$. Following [30], we further define four types of edges: $r_{in}, r_{out}, r_{in-out}, r_{self}$. For edge $(v_i, v_j)$, $r_{in}$ represents there is only transition from $v_j$ to $v_i$, $r_{out}$ means transition only exists from $v_i$ to $v_j$, $r_{in-out}$ indicates there is a bidirectional transition between $v_i$ and $v_j$, and $r_{self}$ refers to a self-loop of an item.

*3.2.3* ***Pairwise-View Graph.*** Pairwise-view graph is constructed by connecting pairs of items in a session. In particular, we denote pairwise-view graph as $\mathcal{G}_{pv} = \{(v^s_i, v^s_j) | v^s_i, v^s_j \in \mathcal{S}\}$, where $v^s_i$ and $v^s_j$ are any two items in the same session.

## 4 METHODOLOGY

Figure 3 depicts the framework of our proposed KMVG, which mainly comprises four modules: 1) ***knowledge-view representation learning***, which learns knowledge-enhanced item representations by employing relation graph aggregation to incorporate each item's semantic neighbors' embeddings based on KG; 2) ***session-view representation learning***, which utilizes a graph convolutional network to learn session-level item representations by capturing the contextual transitions on the session-view graph; 3) ***pairwise-view representation learning***, which mines and aggregates the latent feature commonality of pairwise items within the current session to represent the session embedding of commonality; and 4) ***ensemble and prediction***, which combines the item and session representations learned from the above three views to predict the probability of candidate items for recommendation. We now introduce the four modules in detail.

### 4.1 Knowledge-View Representation Learning

The main task of knowledge-view representation learning is to extract the global item-item relationships from KG. An item $v_1$, in general, is involved in multiple triplets. For example, given two one-hop relationships $v_1 \xrightarrow{r_1} e_1$ and $v_1 \xrightarrow{r_2} e_2$, $e_1$ and $e_2$ are the neighbor entities that have relations $r_1$ and $r_2$ with item $v_1$, respectively. Then, $v_1$ can aggregate multiple relations of neighbor entities $e_1$ and $e_2$ to enrich its own semantics. Furthermore, the semantic information can be propagated and transformed between different items through two- (or more) hop relationships, e.g., $v_1 \xrightarrow{r_1} e_3 \xrightarrow{-r_1} v_2$.

To realize the flow and transformation of semantic information on KG, we employ a knowledge graph attention network to recursively propagate item embeddings along the high-order connectivity of KG [23]. Specifically, for each entity $h$, we denote $\mathcal{N}_h = \{(h, r, t) | (h, r, t) \in \mathcal{G}_{kv}\}$ as the set of triplets, where $h$ and $t$ are the head and tail nodes, respectively. Then, we can perform a one-order relation graph propagation on a one-hop structure to aggregate the semantic information of all tail entities,

$$\mathbf{e}_{\mathcal{N}_h} = \sum\nolimits_{(h,r,t) \in \mathcal{N}_h} \zeta(h, r, t) \cdot \mathbf{e}_t, \tag{2}$$

where $\mathbf{e}_t \in \mathbb{R}^d$ is the initial embedding of the tail node, $d$ indicates the dimension of embeddings, and $\zeta(h, r, t)$ is an attentive weight that reveals how much information will be propagated from $t$ to $h$ through relation $r$, which is implemented by a relation attention mechanism,

$$\begin{aligned} \zeta(h, r, t) &= \frac{exp(s(h, r, t))}{\sum_{(h,r',t') \in \mathcal{N}_h} exp(s(h, r', t'))}, \\ s(h, r, t) &= (\mathbf{W}_r \mathbf{e}_t)^\top tanh(\mathbf{W}_r \mathbf{e}_h + \mathbf{e}_r), \end{aligned} \tag{3}$$

where $s(h, r, t)$ denotes the attentive score that is calculated based on the distance between $\mathbf{e}_h$ and $\mathbf{e}_t$ on relation $r$, i.e., giving higher attention score and propagating more information for closer entities, and $\mathbf{W}_r \in \mathbb{R}^{d \times d}$ is a trainable weight matrix. The final attention score $\zeta(h, r, t)$ is the result of $s(h, r, t)$ normalized through a softmax function.

The one-order entity representation $\mathbf{e}_h$ can be obtained by aggregating the entity representation $\mathbf{e}_h$ and its relation representations $\mathbf{e}_{\mathcal{N}_h}$, i.e., $\mathbf{e}_h^{(1)} = f_1(\mathbf{e}_h, \mathbf{e}_{\mathcal{N}_h})$. Here, we use GCN aggregator [9] as function $f_1$. Subsequently,

$$\mathbf{e}_h^{(1)} = f_1(\mathbf{e}_h, \mathbf{e}_{\mathcal{N}_h}) = \sigma(\mathbf{W}_1(\mathbf{e}_h \oplus \mathbf{e}_{\mathcal{N}_h})), \tag{4}$$

where $\oplus$ is the element-wise addition to achieve self-loop operation, $\mathbf{W}_1 \in \mathbb{R}^{d \times d}$ is a trainable weight matrix to distill useful information for propagation, and $\sigma$ is a nonlinear transformation, set as *LeakyReLU*. To stack more layers to gather the high-order semantic information, we can recursively formulate the representation of the entity $h$. Formally, in the $l$-th layer, the high-order propagation is calculated as,

$$\mathbf{e}_h^{(l)} = f_1(\mathbf{e}_h^{(l-1)}, \mathbf{e}_{\mathcal{N}_h}^{(l-1)}), \tag{5}$$

where the semantic neighbor representations $\mathbf{e}_{\mathcal{N}_h}^{(l-1)}$ in the $(l-1)$-th layer is defined as,

$$\mathbf{e}_{\mathcal{N}_h}^{(l-1)} = \sum\nolimits_{(h,r,t)\in\mathcal{N}_h} \zeta(h,r,t)\mathbf{e}_t^{(l-1)}. \tag{6}$$

Obviously, $\mathbf{e}_t^{(l-1)}$ is the representation of entity $t$ obtained at the $(l-1)$-th layer, which memorizes the semantic information from its $(l-1)$-hop neighbors on KG. Here, to avoid overfitting, we use dropout operation [20] on representation learning, which is effective in enhancing robustness on many neural networks [27, 29].

In the knowledge view, we can catch both low-order and high-order connectivity information by using a relation attentive way. For low-order connectivities, such as $e_1 \xrightarrow{-r_1} v_1$ and $e_2 \xrightarrow{-r_2} v_1$, the information of attributes $e_1$ and $e_2$ can be aggregated into item $v_1$. For high-order connectivities, like $v_1 \xrightarrow{r_1} e_1 \xrightarrow{-r_1} v_2$, the information of item $v_1$ can be propagated to item $v_2$ and explicitly encoded into $v_2$'s embedding. Noteworthy, we extract the ground-truth relationship among items from KG as global item-item relationships to obtain the item representations with rich semantic information, which avoids the redundant mining of strong sequential correlations and the data sparsity problem simultaneously.

## 4.2 Session-View Representation Learning

The contextual relationship of items within a session is very significant for predicting the next item [12, 31]. Based on the directed session-view graph structure, the main task of session-view representation learning is to exploit the complex contextual transitions among items in a session.

Due to the different transition relationships between items, the neighbors of an item may have different importance to this item in a session. Therefore, we utilize graph attention network to obtain the output features $\mathbf{e}_{v_i^s}$ for each item $v_i^s$ in session $s$ by computing a linear weighted combination of the neighbors' features,

$$\mathbf{e}_{v_i^s}^s = \sum\nolimits_{(v_i^s, r_{ij}, v_j^s)\in\mathcal{N}_{v_i^s}} \xi(v_i^s, r_{ij}, v_j^s) \cdot \mathbf{e}_{v_j^s}, \tag{7}$$

where $v_j^s$ is a neighbor of item $v_i^s$ through edge relation $a_{r_{ij}}$ in session $s$, $\mathcal{N}_{v_i^s}$ denotes the one-order neighbors of item $v_i$ in the session-view graph, and $\xi(v_i^s, r_{ij}, v_j^s)$ is an attention score function to reduce the impact of noisy neighbors. Following [30], $\xi(v_i^s, r_{ij}, v_j^s)$ can be computed as,

$$\xi(v_i^s, r_{ij}, v_j^s) = \frac{exp(\mathbf{a}_{r_{ij}}^\top \sigma(\mathbf{e}_{v_i^s} \otimes \mathbf{e}_{v_j^s}))}{\sum_{(v_i^s, r_{ij}, v_k^s)\in\mathcal{N}_{v_i^s}} exp(\mathbf{a}_{r_{ij}}^\top \sigma(\mathbf{e}_{v_i^s} \otimes \mathbf{e}_{v_k^s}))}, \tag{8}$$

where $\otimes$ is element-wise product, $\sigma$ is *LeakyReLU* activate function, $\mathbf{a}_{r_{ij}}$ denotes a relation vector to filter important features and $r_{ij}$ indicates the relation type. Note we denote four relation vectors based on different relations in the session-view graph.

Considering that the initial embeddings of each item are semantically impoverished, we take the item representations learned from the knowledge-view graph as additional input for session-view representation learning. Then, we can obtain the feature $\mathbf{e}_{v_i^s}^{sv}$ of item $v_i^s$ by incorporating different outputs of session-view representation learning, i.e.,

$$\mathbf{e}_{v_i^s}^{sv} = f_2(\mathbf{e}_{v_i^s}^{si}, \mathbf{e}_{v_i^s}^{sk}), \tag{9}$$

where $\mathbf{e}_{v_i^s}^{si}$ and $\mathbf{e}_{v_i^s}^{sk}$ are outputs of session-view representation learning from initial embedding and knowledge embedding, respectively. Notice that $\mathbf{e}_{v_i^s}^{si}$ is the item representation of mining contextual transition, while $\mathbf{e}_{v_i^s}^{sk}$ implies the transition of semantic information in the sequence. We implement $f_2$ by using four types of combinations: sum pooling, average pooling, max pooling, and concatenation.

For next-click prediction, the contributions of items within a session are usually different. Typically, the items clicked later in the session are more representative of the current interest of the user, therefore, we consider adding reversed position information to the embedding of each item [30]. Specifically, we define the position information as a trainable position embedding matrix $\mathbf{P} = [\mathbf{p}_1, \mathbf{p}_2, \ldots, \mathbf{p}_l]$, where $\mathbf{p}_i \in \mathbb{R}^d$ is a position vector at position $i$. By integrating the reversed position embedding $\mathbf{p}_{l-i+1}$ into item embeddings $\mathbf{e}_{v_i^s}^{sv}$, we can obtain the position-aware item representation $\mathbf{h}_{v_i^s}$ as,

$$\mathbf{h}_{v_i^s} = tanh(\mathbf{W}_2[\mathbf{e}_{v_i^s}^{sv}; \mathbf{p}_{l-i+1}] + \mathbf{b}_1), \tag{10}$$

where $[;]$ denotes the concatenation operation, $\mathbf{W}_2 \in \mathbb{R}^{d \times 2d}$ and $\mathbf{b}_1 \in \mathbb{R}^d$ denote trainable weight matrix and bias, respectively. Obviously, the item representation $\mathbf{h}_{v_i^s}$ is an agglomeration of semantic information, sequence information, and position information.

## 4.3 Pairwise-View Representation Learning

As mentioned earlier, items within a session usually have feature correlations and mining such feature-level commonality is beneficial for fine-grained session representation. Therefore, commonality can be used as a supplement to contextual relationships, to represent the local item-item relationships more completely. We develop a pairwise item aggregator to obtain the feature commonality of items by calculating the cross features $\mathbf{z}_{ij}$ for every item-item pair on the pairwise-view graph. Specifically,

$$\mathbf{z}_{ij} = \mathbf{e}_{v_i^s} \otimes \mathbf{e}_{v_j^s}, \tag{11}$$

where $\mathbf{e}_{v_i^s}$ is the initial embedding of item $v_i^s$, $\otimes$ denotes the element-wise multiplication to highlight feature dimensions with similar values between two item embeddings. Then, the session representation $\mathbf{s}_{pair}$ can be calculated by performing pairwise items aggregation,

$$\mathbf{s}_{pair} = LeakyReLU\Big(\frac{1}{l}\sum_{i=1}^{l}\sum_{j=i+1}^{l}\mathbf{z}_{ij}\Big), \tag{12}$$

where $LeakyReLU$ is used to further filter the irrelevant common features. Considering the above computational process is time-consuming with a complexity of $O(l^2)$, we further simplify it as,

$$\begin{aligned}\mathbf{s}_{pair} &= LeakyReLU\left(\frac{1}{l}\sum_{i=1}^{l}\sum_{j=i+1}^{l}\mathbf{e}^{s}_{v_i}\otimes\mathbf{e}^{s}_{v_j}\right)\\ &= LeakyReLU\left(\frac{1}{2l}\left(\sum_{i=1}^{l}\sum_{j=1}^{l}\mathbf{e}^{s}_{v_i}\otimes\mathbf{e}^{s}_{v_j}-\sum_{i=1}^{l}\mathbf{e}^{s}_{v_i}\otimes\mathbf{e}^{s}_{v_i}\right)\right)\\ &= LeakyReLU\left(\frac{1}{2l}\left(\left(\sum_{i=1}^{l}\mathbf{e}^{s}_{v_i}\right)^2-\sum_{i=1}^{l}\left(\mathbf{e}^{s}_{v_i}\right)^2\right)\right),\end{aligned}$$

Consequently, the pairwise item aggregation can be computed in linear time complexity $O(l)$. Such a pairwise item aggregator can highlight the feature commonality of item-item pairs to complete local item-item relationships and improve the quality of session representation in a fine-grained pattern.

## 4.4 Ensemble and Prediction

We first fuse item representations from the session-view graph by a soft-attention mechanism,

$$\mathbf{s}_{sess} = \sum\nolimits_{i=1}^{l}\beta_i\cdot\mathbf{e}^{sv}_{v^s_i}, \tag{13}$$

where $\beta_i$ is an attention weight associated with position-aware embeddings $\mathbf{h}_{v^s_i}$, which can be calculated as:

$$\beta_i = \mathbf{q}^{\top}\sigma(\mathbf{W}_3\mathbf{h}_{v^s_i}+\mathbf{W}_4\bar{\mathbf{s}}^{sv}+\mathbf{b}_2), \tag{14}$$

where $\mathbf{W}_3$, $\mathbf{W}_4\in\mathbb{R}^{d\times d}$ and $\mathbf{q}$, $\mathbf{b}_2\in\mathbb{R}^{d}$ are trainable parameters, and $\bar{\mathbf{s}}^{sv}$ is the average item representations of the current session, i.e., $\bar{\mathbf{s}}^{sv}=\frac{1}{l}\sum_{i=1}^{l}\mathbf{e}^{sv}_{v^s_i}$. Then, we integrate two embeddings $\mathbf{s}_{sess}$ and $\mathbf{s}_{pair}$ to get the final session representation $\mathbf{s}_{final}$,

$$\mathbf{s}_{final} = f_3(\mathbf{s}_{sess},\mathbf{s}_{pair}), \tag{15}$$

We also implement $f_3$ through four types of combination on $s_{sess}$ and $s_{pair}$: sum pooling, average pooling, max pooling, and concatenation. The representation $\mathbf{s}_{final}$ is constructed by fusing multiple item-item relationships, including rich semantic information, contextual transition information, and pairwise commonality information. Finally, we compute the predicted probability $\hat{y}_i$ for each candidate item $v_i\in\mathcal{V}$ by dot product operation,

$$\hat{y}_i = \frac{exp(\mathbf{s}_{final}{}^{\top}\mathbf{e}_{v_i})}{\sum_{v_j\in\mathcal{V}}exp(\mathbf{s}_{final}{}^{\top}\mathbf{e}_{v_j})}, \tag{16}$$

where $\mathbf{e}_{v_i}$ denotes the initial embedding of item $v_i$. The loss function is defined as the cross-entropy of the prediction $\hat{y}_i$ and the ground-truth $y_i$,

$$\mathcal{L} = -\sum\nolimits_{i=1}^{n}y_i log(\hat{y}_i)+(1-y_i)log(1-\hat{y}_i), \tag{17}$$

where $y_i=1$ denotes item $v_i$ appears in next item prediction, and $y_i=0$ otherwise. The mini-batch Adam [8] is employed to optimize and update model parameters.

**Table 1: Statistics of three public datasets.**

| Datasets | | ASoftware | Yelp | Cosmetics |
|---|---|---|---|---|
| Sessions | #Clicks | 29,455 | 253,975 | 1,219,906 |
| | #Train | 9,838 | 75,904 | 162,030 |
| | #Test | 3,055 | 9,995 | 47,951 |
| | #Items | 21,664 | 27,097 | 41,374 |
| | Avg. length | 2.30 | 2.67 | 5.34 |
| KG | #Entities | 58,367 | 29,082 | 42,101 |
| | #Relations | 4 | 17 | 2 |
| | #Triplets | 214,681 | 180,263 | 65,038 |

## 4.5 Model Analysis

We now analyze the time complexity of each of the four modules in KMVG. The time cost of the knowledge view is $O(|\mathcal{V}_{kv}|d^2)$ (the complexities of the translation principle and the attention embedding propagation are both $O(|\mathcal{V}_{kv}|d^2)$). The time cost of the session view is $O(|\mathcal{E}|d)$. The cost of the pairwise view is $\sum_{s=1}^{s=S_n}O(l)$, where $S_n$ denotes the number of sessions and $l$ is the length of the session. For the final ensemble and prediction, the attention score and the inner product are conducted, so its complexity is $O(|\mathcal{E}|d+|\mathcal{V}|d)$. Overall, the training complexity of KMVG is $O(|\mathcal{V}_{kv}|d^2+|\mathcal{E}|d+|\mathcal{V}|d+\sum_{s=1}^{s=S_n}(l))$, which means the actual complexity of KMVG is on the same order of magnitude as that of the ordinary GNN-based SBR model with a single view.

# 5 EXPERIMENT

## 5.1 Experiment Setting

*5.1.1* ***Dataset.*** To evaluate the effectiveness of our proposed model, we conduct experiments on three real-world datasets: Amazon software (**ASoftware** for short), **Yelp**, and **Cosmetics**, which are publicly accessible and vary in terms of domain, size, and sparsity. Besides, these datasets have fundamental metadata or some properties to construct knowledge graphs.

**ASoftware**[1]: Amazon-review is a widely used benchmark for product recommendation [14]. We choose the subset, Amazon software, to evaluate our proposed model.

**Yelp**[2]: This dataset is adopted from the 2018 edition of the Yelp challenge. Here we view the local businesses like restaurants and bars as items. For time performance consideration, we choose the first 1, 000, 000 interactions as the dataset.

**Cosmetics**[3]: This dataset is a Kaggle competition dataset, which records user behaviors in a medium cosmetics online store. We use one month (October 2019) records and only retain the interactions with the type 'purchase' in our work.

*5.1.2* ***Session Setup.*** Cosmetics is a benchmark dataset, in which the session has been segmented. For the other two datasets, considering the real-world scenarios of the datasets (i.e., the amount of software or bars that people consume in a short time is small), we divide each 30-day interaction into a session for ASoftware and Yelp. Afterward, following previous works [1, 12, 31, 37], we filter out all sessions of length 1 and items appearing less than 5 times in all datasets. Furthermore, we generate sequences and corresponding labels by splitting the input sequence [21, 31]. For example,

[1] http://jmcauley.ucsd.edu/data/amazon/

[2] https://www.yelp.com/dataset/

[3] https://www.kaggle.com/mkechinov/ecommerce-events-history-in-cosmetics-shop

**Table 2: Performances (%) of all comparison methods on three datasets. The best and second-best results are highlighted in boldface and underlined, respectively. * denotes KMVG surpasses the second-best model using a pair t-test ($p < 0.01$).**

| Model | Asoftware | | | | Yelp | | | | Cosmetics | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | HR@20 | N@20 | HR@10 | N@10 | HR@20 | N@20 | HR@10 | N@10 | HR@20 | N@20 | HR@10 | N@10 |
| GRU4Rec | 31.07 | 17.82 | 27.23 | 17.40 | 21.62 | 8.37 | 15.84 | 7.74 | 39.25 | 17.94 | 32.23 | 17.24 |
| NARM | 25.34 | 17.27 | 21.45 | 16.28 | 25.59 | 12.99 | 18.81 | 11.28 | 43.26 | 25.05 | 35.94 | 23.19 |
| SR-GNN | 25.26 | 15.71 | 21.56 | 14.77 | 27.77 | 13.16 | 19.61 | 11.11 | 44.83 | 26.23 | 37.26 | 24.35 |
| GC-SAN | 17.45 | 9.87 | 15.25 | 9.32 | 24.56 | 11.30 | 16.64 | 9.31 | 44.08 | 25.26 | 36.46 | 23.33 |
| NISER+ | 19.07 | 12.83 | 16.59 | 12.20 | 25.86 | 13.16 | 19.08 | 11.45 | 45.72 | 26.06 | 37.85 | 24.07 |
| LESSR | 17.47 | 12.09 | 15.94 | 11.53 | 28.66 | 13.63 | 20.00 | 11.48 | 45.60 | 26.56 | 37.94 | 24.62 |
| MSGIFSR | 28.03 | 18.48 | 23.46 | 16.15 | 32.34 | 15.77 | 22.90 | 13.39 | 46.53 | 26.54 | 38.79 | 24.58 |
| CSRM | 29.60 | 19.10 | 24.68 | 17.57 | 25.22 | 12.32 | 15.89 | 9.15 | 45.88 | 26.21 | 34.88 | 22.49 |
| GCE-GNN | 32.37 | 18.05 | 27.02 | 16.71 | 28.69 | 14.12 | 20.58 | 12.00 | 46.26 | 26.81 | 38.69 | 24.89 |
| SR-GNN+KG | 26.57 | 17.23 | 22.34 | 16.15 | 28.04 | 13.64 | 20.23 | 11.69 | 44.96 | 26.31 | 37.50 | 24.42 |
| MGS | 34.08 | 18.28 | 28.67 | 16.90 | 32.49 | **16.94** | 24.40 | **14.90** | 42.73 | 24.58 | 35.12 | 22.66 |
| KMVG | **39.32**$^*$ | **20.14**$^*$ | **32.85**$^*$ | **18.57**$^*$ | **33.59**$^*$ | 16.89 | **24.71**$^*$ | 14.68 | **47.01**$^*$ | **27.24**$^*$ | **39.14**$^*$ | **25.25**$^*$ |

for an input session $\mathcal{S} = \{v_1^s, v_2^s, ..., v_l^s\}$, we generate a series of sequences and labels $(\{v_1^s\}, v_2^s)$, $(\{v_1^s, v_2^s\}, v_3^s)$,...,$(\{v_1^s, v_2^s, ..., v_{l-1}^s\}, v_l^s)$, where $\{v_1^s, v_2^s, ..., v_{l-1}^s\}$ is the sequence and $v_l^s$ indicates the next-clicked item, i.e., the label of the sequence.

5.1.3 ***Knowledge Graph Constructing.*** For ASoftware, we utilize its metadata to construct the knowledge graph. For Yelp and Cosmetics, we construct their knowledge graphs by extracting the properties in the original datasets. To be specific, there are 4 relations (e.g., category, brand) in the KG of ASoftware, 2 relations (i.e., category and brand) in Cosmetics, and 17 relations (e.g., stars, location) in Yelp. Details of these datasets are summarized in Table 1.

## 5.2 Baselines

We compare KMVG with the following four groups of baselines:
**RNN-based SBR methods:**

- **GRU4Rec** [5] is a classical RNN-based method to model the sequential pattern of items in a session by using a Gated Recurrent Unit [2].
- **NARM** [12] employs RNN with an attention mechanism to capture user's preferences.

**GNN-based SBR methods:**

- **SR-GNN** [31] employs gated graph neural networks to capture the contextual transition between items within a session.
- **GC-SAN** [33] uses a self-attention layer after GNN to integrate the contextual information as session representations.
- **NISER+** [4] utilizes dropout and $L_2$ norm to alleviate overfitting and long-tail effect.
- **LESSR** introduces shortcut graph attention and edge-order preserving aggregation layers to tackle information loss and long-range dependency problems.
- **MSGIFSR** [3] adopts a heterogeneous session graph to extract the user's multi-granularity intent for enhancing the recommendation performance.

**Cross-Session SBR methods:**

- **CSRM** [16] utilizes the memory networks to incorporate the latest $m$ sessions for a better prediction of the intent of the current session.
- **GCE-GNN** [30] treats items as the minimum granularity and exploits the item transitions from a session graph and global graph to model the session representations.

**SBR methods with KG:**

- **SR-GNN+KG** is an improved version of SR-GNN by adding KG embedding on SR-GNN in the same way as KMVG.
- **MGS** [11] introduce a mirror graph built by attributes of items that selects the most attribute-representative information for each session item to assist the session graph.

## 5.3 Parameter Setting

For each dataset, we randomly select 80% of the sessions to constitute the training set and treat the remaining as the test set. Moreover, we randomly select 10% of the sessions in the training set as the validation set to tune hyper-parameters. For a fair comparison, we employ grid search to find the best hyper-parameters of all models based on the validation set. All parameters are initialized using a Gaussian distribution with a mean of 0 and a standard deviation of 0.1. The mini-batch Adam optimizer is exerted to optimize these parameters, where the initial learning rate is set to 0.001 and will decay by 0.1 after every 3 epochs. Moreover, the embedding dimension, the batch size, and the $L_2$ penalty are set to 50, 100, and $10^{-5}$ respectively. For GNN-based models, the number of layers is searched in $\{1, 2, 3, 4\}$, and the dropout ratio is searched in $\{0.1, 0.2, \ldots, 0.9\}$. To evaluate all models, we adopt two widely used ranking-based metrics: HR@$N$ (Hit Rate) and N@$N$ (Normalized Discounted Cumulative Gain) with the truncated list $[10, 20]$.

## 5.4 Performance Comparison

The overall performance comparison of all methods is represented in Table 2. From the results, we have the following key observations:

**Table 3: Impact of different components in KMVG.**

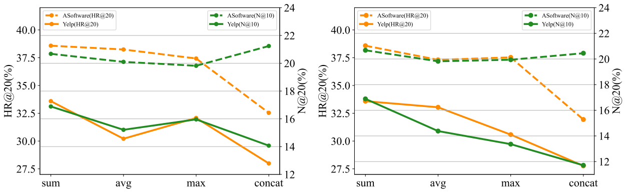
| Dataset | ASoftware | | Yelp | | Cosmetics | |
|---|---|---|---|---|---|---|
| Metrics | HR | N | HR | N | HR | N |
| **w/o KV** | 32.82 | 16.36 | 33.05 | 16.73 | 46.90 | 26.93 |
| **w/o SV** | 36.43 | 20.08 | 26.92 | 12.70 | 34.89 | 17.26 |
| **w/o PV** | 38.77 | 19.68 | 33.18 | 16.78 | 46.91 | 27.21 |
| **KMVG** | **39.32** | **20.14** | **33.59** | **16.89** | **47.01** | **27.24** |

- *GNN-based methods generally outperform RNN-based methods on long sequential datasets.* The main reason is that RNN cannot accurately capture long-distance dependencies, while GNN excels at mining the contextual features among items within a session for next-item prediction.
- *CSRM and GCE-GNN, both cross-session methods, achieve better performance than normal GNN-based methods.* For instance, GCE-GNN achieves 14.89%, 7.29% and 2.21% improvements over SR-GNN on the three datasets in terms of N@20, respectively. The results indicate the effectiveness of incorporating item transitions from the global graph. Besides, CSRM outperforms GCE-GNN on N@20 in ASoftware, possibly because the long-term memory may be beneficial on sparse ASoftware.
- *Methods of introducing KG outperform others.* In particular, SR-GNN+KG consistently outperforms SR-GNN, and MGS that utilizes the attributes of items also achieves promising performance among all baselines, which proves the effectiveness of knowledge graph in introducing real external semantic information to enhance global item-item relationships.
- *KMVG almost outperforms all baselines on all three datasets.* Specifically, KMVG outperforms the strongest baselines *w.r.t* HR@20 by 15.38%, 3.38%, and 1.03% on ASoftware, Yelp, and Cosmetics respectively, and is only inferior to MGS slightly in terms of N@20 and N@10 on Yelp. These results demonstrate exploiting multiple dependencies among items can improve the ability of session representation to facilitate recommendation. Furthermore, KMVG performs better in sparser sessions, probably because additional supplementary information (i.e., semantic information and commonality among items) becomes particularly important in the absence of sufficient transition information.

## 5.5 Ablation Studies

We further conduct ablation experiments to investigate the effect of different components in KMVG. Table 3 shows the results in terms of HR@20 and N@20 (represented as HR and N in Table 3) on three variants: **w/o KV** is a variant of KMVG by removing semantic-view representation learning and only reserves the session-view and pairwise-view. **w/o SV** is a variant of kMVG that only utilizes the additional information from knowledge-view and pairwise-view. Similarly, the variant **w/o PV** removes the pairwise-view representation learning and only retains knowledge-view and session-view in KMVG. From the results, we have the following findings:

- The performance will decrease if any of the views is removed, which suggests the effectiveness of modeling multiple item-item relationships of different levels from multiple views and the ability of KMVG to leverage those relationships.
- N@20 drops by an average of 30.37% on the three datasets after removing session-view, and the performances of **w/o SV** on Yelp and Cosmetics are the worst compared to the other two variants, showing the indispensability of contextual transition in the session-view graph.
- Compared with the session view, the knowledge view has a greater impact on ASoftware. In particular, the HR@20 on ASoftware decreased by 19.8%. The main reason is that the sparsity of ASoftware (i.e., ASoftware has the shortest average length per session among three datasets) is very tricky to mine accurate contextual sequence information from the session view, while knowledge graphs can provide additional richer semantic information to alleviate such data sparsity. Besides, the triplet sparsity in KG also influences the model performance, the triples of the knowledge graph of ASoftware are the densest in the three datasets, this is probably why **KV** has a big impact on Asoftware. Therefore, KMVG may better boost the performance of RS in scenes where the item's KG has dense triples.
- Although pairwise view (**PV**) has the least impact, the performance also decreases when it is removed, indicating that mining the commonality of items in the same session is also beneficial for further enhancing session-based recommendation.



(a) implementation of $f_2$ (b) implementation of $f_3$

**Figure 4: Impact of the different implements of $f_2$ and $f_3$.**

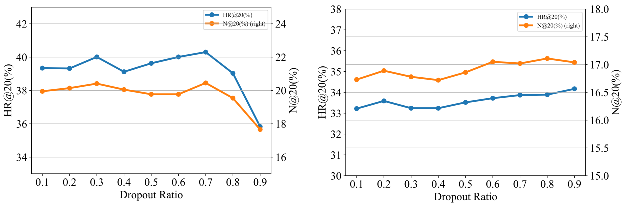
## 5.6 Depth Analysis of KMVG

*5.6.1* ***Impact of combined approach.*** We explore the impact of the combined approach of $f_2$ and $f_3$ by experimenting on ASoftware and Yelp. Specifically, we implement $f_2$ (i.e., fusing$\mathbf{e}_{v_i^s}^{si}$ and $\mathbf{e}_{v_i^s}^{sk}$) and $f_3$ (i.e., fusing $\mathbf{s}_{sess}$ and $\mathbf{s}_{pair}$) by using four strategies, sum pooling, average pooling, max pooling, and concatenation, respectively. From Figure 4, we can find that sum pooling performs the best on both $f_2$ and $f_3$ on Yelp and HR@20 of ASoftware, and concatenation performs the best on Asoftware of N@20, but performs the worst on other indicators. After a comprehensive consideration, we choose sum pooling for both $f_2$ and $f_3$.

*5.6.2* ***Impact of number of layers.*** We investigate the influence of the number of layers $L$ in the knowledge view by searching $L$ in the range of $\{1, 2, 3, 4\}$. Specifically, $L = 1$ means items only aggregate their attributes entities, while $L = 2$ implies that semantic information between items with the same attribute can be transferred to each other. Table 4 reports the results of different numbers of layers on ASoftware and Yelp, from which we can observe that $L = 2$ works the best on Yelp on both metrics, while KMVG achieves the best performance with $L = 1$ in terms of N@20 on ASoftware. The main reason may be that the knowledge-view graph of ASoftware has richer attribution entity information than Yelp, thereby only aggregating attribute information is not enough for Yelp. The results demonstrate that the semantic information transition among

**Table 4: Impact of layer numbers in knowledge-view graph.**

| Datasets | ASoftware | | Yelp | |
|---|---|---|---|---|
| Measures | HR@20 | N@20 | HR@20 | N@20 |
| Layer-1 | 37.98 | **20.23** | 32.61 | 16.44 |
| Layer-2 | 39.32 | 20.14 | **33.59** | **16.89** |
| Layer-3 | **39.68** | 19.96 | 33.17 | 16.77 |
| Layer-4 | 38.77 | 19.47 | 33.58 | 16.82 |



(a) ASoftware

(b) Yelp

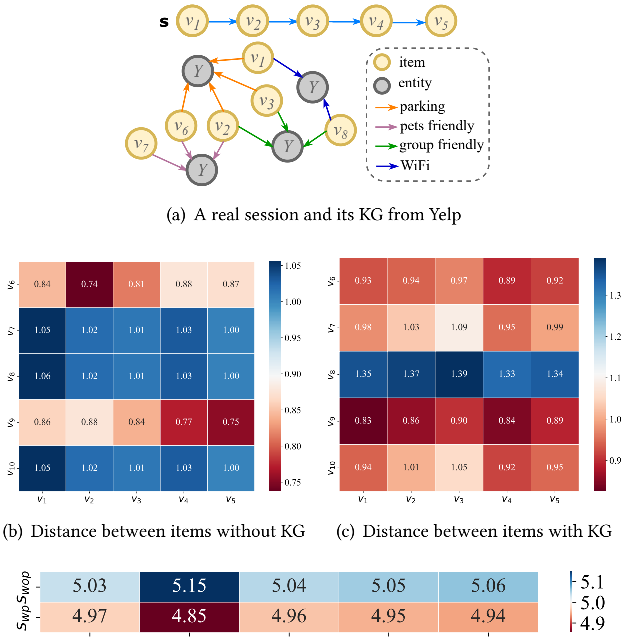
**Figure 5: Impact of dropout ratio.**

items is beneficial for improving item representation on sparse data, while aggregation of multi-hop neighbor knowledge may be noisy and negatively affect performance.

*5.6.3* ***Impact of dropout ratio.*** We employ dropout regularization techniques in the knowledge-view representation learning to improve the robustness of our model in the training process. We explore the effect of different dropout ratios in the training process by searching the dropout ratio in $\{0.1, 0.2, \ldots, 0.9\}$ on ASoftware and Yelp. As shown in Figure 5, we can observe that the performances show a tortuous rise (until it reaches the peak at dropout ratio = 0.7) and then begin to fall on ASoftware, while the performances on Yelp2018 present a tortuous upward as a whole. The reason is that there is noise in the knowledge graph. The larger the dropout ratio, the more robust the model. However, too large a dropout ratio will reduce the utilization of semantic relations in the knowledge graph, resulting in performance degradation, especially for ASoftware dataset with fewer relations in KG.

## 5.7 Case Study

To prove that KMVG can effectively use the multiple item-item relationships, we take a real session from Yelp, and its KG is shown in Figure 6 (a) (Due to space concern, we only present part of the KG), which contains four relations: whether the restaurant can park or not, whether it has WiFi, whether it is pet-friendly, and whether it is friendly to groups. Figure 6 (b) and Figure 6 (c) are heatmaps of the Euclidean distance between items in the session $(v_1, v_2, \ldots, v_5)$ and several other randomly picked items $(v_6, v_7, \ldots, v_{10})$, where Figure 6 (b) does not utilize KG and Figure 6 (c) uses information of KG. We can find that after using the information of KG, the distance between items with common relationships and entities will be closer, and otherwise farther. Specifically, Figure 6 (d) shows the Euclidean distance between items in the session and session's representation, where $s_{wop}$ indicates the representation not using commonality and $s_{wp}$ stands for the representation fusing with commonality. It can be observed that the Euclidean distance between items and $s_{wp}$ is closer than the Euclidean distance between items and $s_{wop}$.



(a) A real session and its KG from Yelp

(b) Distance between items without KG

(c) Distance between items with KG

(d) Distance between items and session representation (with pairwise-view and without pairwise-view)

**Figure 6: Distance analysis between item features with a real session and knowledge graph on Yelp.**

# 6 CONCLUSION

In this paper, we propose a novel graph-based SBR model KMVG to mine global and local item-item relationships to enhance the session representations from three views, i.e., *knowledge view*, *session view*, and *pairwise view*, in which tailor-made graph networks are adopted to mine item and session representations from the three views. Specifically, we adopt a knowledge graph attention network to capture the semantic information as global item-item relationships on the knowledge-view graph. For the session view, we take initial embedding and knowledge embedding of items as inputs to extract complex contextual transitions as sequence patterns of local item-item relationships by applying a graph attention network. Besides, we develop a pairwise item aggregator to extract commonalities between any two items of the same session as feature patterns of local item-item relationships from the pairwise-view graph. Finally, the session representations learned from multiple views are fused to perform next-item prediction. Comprehensive experiments on three public datasets demonstrate the superiority of KMVG over state-of-the-art methods. Further results validate the effectiveness of extracting and leveraging global and local item-item relationships from our proposed three views. For future work, we will try to explore if knowledge graph distillation can help reduce the impact of unrelated entity noise.

# ACKNOWLEDGMENTS

The work was partially supported by the National Natural Science Foundation of China under Grant No. 62272176. We would like to thank the anonymous reviewers for their valuable comments.

# REFERENCES

[1] Tianwen Chen and Raymond Chi-Wing Wong. 2020. Handling information loss of graph neural networks for session-based recommendation. In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. 1172–1180.

[2] Kyunghyun Cho, Bart Van Merriënboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. 2014. Learning phrase representations using RNN encoder-decoder for statistical machine translation. *arXiv preprint arXiv:1406.1078* (2014).

[3] Jiayan Guo, Yaming Yang, Xiangchen Song, Yuan Zhang, Yujing Wang, Jing Bai, and Yan Zhang. 2022. Learning Multi-granularity Consecutive User Intent Unit for Session-based Recommendation. In *Proceedings of the Fifteenth ACM International Conference on Web Search and Data Mining*. 343–352.

[4] Priyanka Gupta, Diksha Garg, Pankaj Malhotra, Lovekesh Vig, and Gautam M Shroff. 2019. NISER: normalized item and session representations with graph neural networks. *arXiv preprint arXiv:1909.04276* (2019).

[5] Balázs Hidasi, Alexandros Karatzoglou, Linas Baltrunas, and Domonkos Tikk. 2015. Session-based recommendations with recurrent neural networks. *arXiv preprint arXiv:1511.06939* (2015).

[6] Jin Huang, Wayne Xin Zhao, Hongjian Dou, Ji-Rong Wen, and Edward Y Chang. 2018. Improving sequential recommendation with knowledge-enhanced memory networks. In *The 41st International ACM SIGIR Conference on Research & Development in Information Retrieval*. 505–514.

[7] Dietmar Jannach and Malte Ludewig. 2017. When recurrent neural networks meet the neighborhood for session-based recommendation. In *Proceedings of the eleventh ACM conference on recommender systems*. 306–310.

[8] Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980* (2014).

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

[10] Yehuda Koren, Robert Bell, and Chris Volinsky. 2009. Matrix factorization techniques for recommender systems. *Computer* 42, 8 (2009), 30–37.

[11] Siqi Lai, Erli Meng, Fan Zhang, Chenliang Li, Bin Wang, and Aixin Sun. 2022. An Attribute-Driven Mirror Graph Network for Session-Based Recommendation. In *Proceedings of the 45th International ACM SIGIR Conference on Research and Development in Information Retrieval* (Madrid, Spain) *(SIGIR '22)*. Association for Computing Machinery, 1674–1683.

[12] 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.

[13] Yankai Lin, Zhiyuan Liu, Maosong Sun, Yang Liu, and Xuan Zhu. 2015. Learning entity and relation embeddings for knowledge graph completion. In *Twenty-ninth AAAI conference on artificial intelligence*.

[14] Jianmo Ni, Jiacheng Li, and Julian McAuley. 2019. Justifying recommendations using distantly-labeled reviews and fine-grained aspects. In *Proceedings of EMNLP-IJCNLP*. 188–197.

[15] 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.

[16] 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.

[17] 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.

[18] B. Sarwar, G. Karypis, J. Konstan, and J. Riedl. 2001. Item-Based Collaborative Filtering Recommendation Algorithms. In *10th International World Wide Web Conference*.

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

[20] Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. 2014. Dropout: a simple way to prevent neural networks from overfitting. *The journal of machine learning research* 15, 1 (2014), 1929–1958.

[21] 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.

[22] Trinh Xuan Tuan and Tu Minh Phuong. 2017. 3D convolutional networks for session-based recommendation with content features. In *Proceedings of the eleventh ACM conference on recommender systems*. 138–146.

[23] Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Lio, and Yoshua Bengio. 2017. Graph attention networks. *arXiv preprint arXiv:1710.10903* (2017).

[24] Hongwei Wang, Fuzheng Zhang, Mengdi Zhang, Jure Leskovec, Miao Zhao, Wenjie Li, and Zhongyuan Wang. 2019. Knowledge-aware graph neural networks with label smoothness regularization for recommender systems. In *Proceedings of the 25th ACM SIGKDD international conference on knowledge discovery & data mining*. 968–977.

[25] Meirui Wang, Pengjie Ren, Lei Mei, Zhumin Chen, Jun Ma, and Maarten de Rijke. 2019. A collaborative session-based recommendation approach with parallel memory modules. In *Proceedings of the 42nd international ACM SIGIR conference on research and development in information retrieval*. 345–354.

[26] Xiang Wang, Xiangnan He, Yixin Cao, Meng Liu, and Tat-Seng Chua. 2019. Kgat: Knowledge graph attention network for recommendation. In *Proceedings of the 25th ACM SIGKDD international conference on knowledge discovery & data mining*. 950–958.

[27] 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.

[28] Xiang Wang, Tinglin Huang, Dingxian Wang, Yancheng Yuan, Zhenguang Liu, Xiangnan He, and Tat-Seng Chua. 2021. Learning intents behind interactions with knowledge graph for recommendation. In *Proceedings of the Web Conference 2021*. 878–887.

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

[30] 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.

[31] 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.

[32] Xin Xia, Hongzhi Yin, Junliang Yu, Yingxia Shao, and Lizhen Cui. 2021. Self-Supervised Graph Co-Training for Session-based Recommendation. In *Conference on Information and Knowledge Management*.

[33] Chengfeng Xu, Pengpeng Zhao, Yanchi Liu, Victor S Sheng, Jiajie Xu, Fuzhen Zhuang, Junhua Fang, and Xiaofang Zhou. 2019. Graph Contextualized Self-Attention Network for Session-based Recommendation.. In *IJCAI*, Vol. 19. 3940–3946.

[34] Rui Ye, Qing Zhang, and Hengliang Luo. 2020. Cross-Session Aware Temporal Convolutional Network for Session-based Recommendation. In *2020 International Conference on Data Mining Workshops (ICDMW)*. IEEE, 220–226.

[35] Fuzheng Zhang, Nicholas Jing Yuan, Defu Lian, Xing Xie, and Wei-Ying Ma. 2016. Collaborative knowledge base embedding for recommender systems. In *Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining*. 353–362.

[36] R. Zhang, Y. Gu, X. Shen, and H. Su. 2021. Knowledge-enhanced Session-based Recommendation with Temporal Transformer. (2021).

[37] 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. *arXiv preprint arXiv:2205.04181* (2022).